CoCo3FPGA

Users Guide for
Version 1.1

Legal Stuff

Please report bugs to the author, but before you do so, please make sure that this is not a derivative work and that you have the latest version of this file.

Introduction

CoCo3FPGA is a Verilog/VHDL implementation in a programmable FPGA of a Tandy / Radio Shack Color Computer 3 (CoCo3). It gives all the functionality of the original CoCo3 within a few limits. These limitations will be discussed later in this document. Including the standard features of the original CoCo3, CoCo3FPGA includes many features that were add-ons to the original. These features include 1 Meg of RAM, Multi-Pak, Deluxe RS-232 Pak, Orchestra-90 CC Stereo Music Synthesizer, and Floppy / Mass Storage Interface.

CoCo3FPGA version 1.10 is implemented into a Digilent Inc. Spartan 3 Starter Board with the 1000K gate upgrade. At the time of this release, the 1000K gate FPGA is the standard. This board has almost everything needed to implement the COCO3 in an FPGA. An additional analog add on board is required if Deluxe RS-232 Cartridge, Joysticks, and Speaker are needed. But this additional hardware is not needed for basic CoCo3 functionality. The design files are included for this add-on board.

Processor

The CoCo3 came with a Motorola 6809 processor. The default speed was 0.89 MHz and it had software programmable hardware that doubled this speed to 1.78 MHz, the CPU rate bit. CoCo3FPGA also runs at the default 0.89 MHz. and has the same hardware rate bit. But the actual speed that CoCo3FPGA runs while at the faster speed is a manual switch setting. The Digilent Starter board includes 8 slide switches; switches 0 and 1 set the speed for the fast speed setting.

Switches		Speed
1	0	
OFF	OFF	1.78 MHz
OFF	ON	4.17 MHz
ON	OFF	12.5 MHz
ON	ON	25.0 MHz*

*Because of the way video memory reads are interlaced with CPU accesses, a 25 MHz CPU clock cycle is skipped every 8 cycles. The effective CPU speed is 25 * 7/8 = 21.875 MHz.

The 6809 logic core, CPU09, has been borrowed from the System09 on opencores.com and was written by John Kent. It is not cycle accurate to the Motorola 6809. The crude measurements taken from me and helpers on the web show this CPU to be approximately 15% faster than the original 6809. But not all instructions are completed faster than the 6809. This is a benefit for most applications, but a hindrance for others. If the application was written with tight timing requirements, then it will probably fail under CoCo3FPGA. An example of this failure is Sock Master's Boink bouncing ball demo. The Boink demo uses the "mul" instruction to waste CPU clock cycles for timing purposes. The "mul"

instruction is one of the examples where the 6809 completes the instruction faster than the CPU09. A special switch has been added to CoCo3FPGA especially for the Boink demo program. By setting switches 0 and 1 to the 1.78 MHz setting and turning on switch 7, the CPU speed will be adjusted where the Boink demo runs. It is not perfect, but not too bad.

RAM / ROM / Multi-Pak Interface

The original CoCo3 came equipped with 128K of Dynamic RAM. There was an upgrade available for 512K. Several third party vendors sold upgrades to 1Meg or even greater. CoCo3FPGA comes with a full 1M of Static RAM. No additional upgrades are available at the present time.

The original CoCo3 came with 32K of built in ROM and a single slot to plug in additional Program Paks. An optional Multi-Pak Interface (MPI) upgraded the system to a total of 4 Program Paks. CoCo3FPGA also has the original 32K built in ROM. This ROM is implemented in the block RAM inside the Xilinx Spartan 3 FPGA and is loaded at power up. No user interaction is needed for the CoCo3FPGA to boot into EXTENDED COLOR BASIC 2.0.

The MPI functionality is also included. The FPGA does not have enough block RAMs to implement the Program Pak ROMs in the 4 slot MPI. One additional ROM, slot 3, is emulated using 256K of the extra 512K Static RAM that EXTENDED COLOR BASIC does not recognize. See the memory map in the next section for the locations of the additional ROM.

Slot 1 of the MPI holds the ROM from the RS232 interface. The analog board is required to use the RS232 functionality. This ROM is implemented using block RAM from the Spartan 3 FPGA.

Slot 4 holds the disk controller. The ROM for this slot is also implemented using block RAM in the Spartan 3 FPGA. Just like BASIC, the Disk controller ROM is loaded at power up and no user interaction is needed to boot into DISK EXTENDED COLOR BASIC 2.1.

Slot 2 holds a utility to load the ROM for slot 3. It uses the same interface to retrieve the data from the PC server as the floppy disk. See the Floppy Interface and PC Software sections of this document for more information. The utility program is implemented using block RAM in the Spartan 3 FPGA. Using the utility in slot 2 of the MPI, a ROM can be loaded into the unused RAM and enabled as slot 3. The utility will read ROMs up to 256K using a bank switch approach. The original CoCo and CoCo2 was only capable of up to 16K of external ROM addressed from $C000-$FEFF. Many larger ROMs were released that switched in 16K banks of ROM by writing the bank number into the Spare Chip Select addresses, $FF40-$FF5F. CoCo3FPGA allows this same type of bank switching. Because the CoCo3 has the capabilities of having 32k ROMs and excludes the Vector page, $FE00-$FEFF, CoCo3FPGA mirrors the 16K bank addresses into the other

16k, $8000-$BFFF, when the ROM size is set to 32k for Slot 3. The utility relies on this mirror to write the entire 16k. The utility loads all 256K of RAM based ROM, in 16k pages, regardless of the original ROM size. This requires some additional time when loading a smaller ROM, but greatly simplifies the loader utility / protocol. The PC Server program is written to mirror smaller ROMs throughout the 256K RAM based ROM. When the ROM loading is finished, the utility instructs the user to switch the MPI switches into the slot 3 position and hit RESET to start the execution process. This part of the CoCo3FPGA has not been extensively tested. Only a few ROMs were tested. ROMs that seem to work include:

ROM	Tandy Part #	Size
Tandy Diagnostics	26-3019	2K
Galactic Attack	26-3066	4K
Orchestra-90 CC*	26-3143	8K

*Floppy reads and writes do not work.

Other ROMs do not work and no additional effort has been done to figure out why. ROMs are not my main interest, but I would like to work with someone to help figure out why others do not work. Please register working and non-working ROMs at the Yahoo CoCo3FPGA group.

The MPI has a four position selection switch on the front of the unit. CoCo3 FPGA emulates this functionality by using two more of the Starter board slide switches. Switch 2 and switch 3 are decoded to these slot switch positions.

Switches		Slot
3	2	
OFF	OFF	1
OFF	ON	2
ON	OFF	3
ON	ON	4

Two new features are added to the CoCo3FPGA. The first feature allows any ROM to be written. The block RAMs in the Spartan 3 FPGA that are used to hold the ROMs can be re-written. But CoCo3FPGA write protects these to protect their contents. This protection is extended to the RAM used to hold the ROM for Slot 3. This protection can be overridden to make modifications. The ROM loaded utility uses this method to load the slot 3 ROM. To override the write protect, bits 2 and 3 of the MPI byte, $FF7F, are used. Bits 0 and 1 hold the slot number for the Spare Chip Select. Bits 4 and 5 hold the slot number for the Chip Select. By writing a 1 and 0 into bits 2 and 3 and then writing a 0 and 1 into bits 2 and 3, all ROMs will be unprotected. Any other write to this location will write protect the ROMs. Since bits 0, 1, 4, and 5 are written at the same time as bits 2

and 3, make sure they are programmed with the proper slot code. The slot code is the same for both sets of bits

Slot	Code
1	00
2	01
3	10
4	11

The second new feature added to CoCo3FPGA is the ability to turn off the interrupt signal from the slots. By turning on Switch 4, the interrupts for all slots are disabled. Normally the interrupt is enabled. There is interrupt sources for the all four slots.

Slot	Source
1	6551 UART IRQ
2	Auto start clock
3	Auto start clock
4	6850 UART and 6551 UART IRQ

When the slot selector switch is set to slot 4, both UART interrupts can be used. This is intended to be utilized under NitrOS-9.

Memory Map

The memory map for the lower 512K is exactly the same as the original CoCo3. There are various documents available that detail the usage of this part of the memory map. All that will be covered here is the differences in the original CoCo3 and CoCo3FPGA.

The MMU registers located at $FFA0 to $FFAF are extended by one bit to allow the use of the upper 512K. Only the lower six bits of the registers could be read or written. With CoCo3FPGA, seven bits are valid for all reads and writes.

The register located at $FF9B is used to switch the video addresses from one 512K bank to the other. This complies with the Disto 2 Meg upgrade and the standard for NitrOS-9. Only the lower bit is utilized for this bank switching.

There is a real time clock mapped at addresses $FFC0 to $FFC7. There is no support for time under DISK BASIC. See the NitrOS-9 section for complete details of this interface.

Under EXTENDED COLOR BASIC only the lower 512k of memory is used. The 256K section of RAM located at $C0000 to $FFFFF is used as ROM for slot 3 of the MPI. The remaining 256K of RAM, $80000 to $BFFFF, is unused for BASIC.

NitrOS-9 uses the full 1M of memory.

Keyboard

The CoCo3 included a 57 key keyboard. CoCo3FPGA uses a PS/2 keyboard that emulates the original 57 key keyboard. There is no additional software needed and all the original CoCo3 programs will work without modification. The keyboard layout is not the same between the original keyboard and the PS/2 keyboard. CoCo3FPGA translates the PS/2 key layout to the CoCo3 layout. So when you push a [shift] 8 on the PS/2 keyboard, CoCo3FPGA will display a "*" on the display. Pushing the [shift] 8 key on a CoCo3 will display a "(" on the screen.

NitrOS-9 does a few things differently with the keyboard. This will be discussed later in this document.

Buttons and Switches

The Digilent Starter board includes 8 slide switches and 4 buttons. The slide switches are explained elsewhere in this document, but included here for completeness. The first two slide switches, switch 0 and 1, are used to set the CPU clock speed when the software rate bit is set. See the CPU section for switch settings.

The next two slide switches set the default MPI switch setting. See the RAM / ROM / MPI section for switch settings.

Switch 4 is used to disable all interrupts generated from the MPI slots. Switch 5 is used to swap Joysticks 0 and 1 with Joysticks 2 and 3. Switch 6 is used to set the speed of the access to the disk and should be left in the off position. The final switch, Switch 7, is the special CPU speed switch for the Boink demo program.

Each of the four push buttons have different functions. The first button, Button 0, is used to show the CoCo3 Easter Egg. Because of the way the keyboard works, the special keys needed to show the Easter Egg do not work at reset. This button simulates the special keys. If you push this button it appears to the CoCo3FPGA that both the Control and ALT buttons are pushed.

Button 1 is used to change the display of the 8 LEDs. See the LED section for detailed information on the button.

Button 2 is used to pause the CPU. This button can cause the system to be unstable and should not be used extensively.

Button 3 is the system reset button. It is just like pushing the reset button on the rear of the CoCo3.

Video

The Digilent Starter board includes a DB-15 Video connector for use with a VGA monitor. The standard video modes of the CoCo3 are displayed as a 640 x 480 60 Hz VGA video mode. The 32 and 40 character text and similar graphics modes use 2 pixel wide dots. The 80 character text and similar graphics modes use 1 pixel wide dots. Every CoCo3 video mode scan line is represented by two scan lines on the VGA monitor. The VGA 640x480 60 Hz video mode timing is similar to but not exactly the same as the original CoCo3 timing. The pixel clock rate is 25 MHz (40 nS). And each scan line is 800 pixels across including the times for horizontal blanking and sync. This means that the horizontal sync pulse is 32 µS apart. The standard CoCo3 sync is 63.5 µS apart. But because each CoCo3 scan line is two lines on the VGA screen, every other sync pulse is invisible to the interrupt and timing circuitry. This gives an effective sync rate of 64 µS.

The vertical time is also similar but not identical. Each frame consists of 521 scan lines including the vertical blanking and sync times. This gives a vertical sync pulse every 16.672 mS. The standard CoCo3 has a vertical sync time of 16.667 mS.

The Digilent Spartan 3 Starter board only supports single bit per color video. The CoCo3 supports two bit per color video. The starter board is easily modified to allow for two colors. The original 270 Ohm resistors, R57, R58, and R59, are replaced with 400 Ohm resistors. Three additional 800 Ohm resistors need to be added to the design. The first resistor is attached to connector A1 pin 4 and wired to VGA connector pin 3. A second resistor is attached to connector A1 pin 21 and wired VGA connector pin 2. A third resistor is attached to connector A1 pin 22 and wired to VGA connector pin 1. A picture of this modification is included in the CoCo3FPGA distribution. A compile in the FPGA option is set to support the extra bits. If this compile option is commented out, the system will try to implement the extra colors with varying bit widths. It is not as good as but better than nothing. See the implementation section for more details.

The CoCo3 composite video is not supported with CoCo3FPGA.

Sound / Orchestra-90 CC Stereo Music Synthesizer

CoCo3FPGA includes the capabilities to drive either powered stereo speakers or stereo headphones. There are several sources for sound in the CoCo3. CoCo3FPGA only implements three of these functions. The sounds generated internally to the CoCo3, single bit sound and 6-bit sound, are mono sources. These two sounds are placed on each channel of the stereo output. The Orchastra-90 is a stereo source and is output that way. The ROM for the Orchastra-90 is not automatically loaded into the Multi-Pak interface. This ROM is included in this distribution of CoCo3FPGA and can be loaded dynamically. See the section on loading ROMs for slot 3 of the Multi-Pak interface for instructions on loading this ROM.

The hardware needed to drive a speaker / headphones are located on the Analog board. If you do not wish to add this analog board, the system will function normally, but the Sound / Orchestra-90 CC cannot be heard.

Serial Ports

The Digilent Starter board includes a DB-9 connector for use as a serial port interface. The RS-232 logic level translators are also included on the board. This serial port only includes the serial transmit and serial receive signals. No other handshaking signals are available. This connector is used for a different feature in the CoCo3FPGA and will be discussed later in this document.

The Digilent Starter board also has a 2 pin header for another serial port. This port also includes the RS-232 logic level translators. This port does not include any handshaking signals either. The transmit pin is connected to the bit bang serial port logic in the CoCo3FPGA. Likewise, the receive pin is connected to the bit bang receive circuitry of the CoCo3FPGA. The carrier detect signal is permanently wired for TRUE.

Slot 1 holds the hardware and software for the Deluxe RS-232 Program Pak, 26-2226. To use this serial port, the analog board is required. It has the additional DB-9 connector and the RS-232 logic level translators. This port includes RTS / CTS handshaking.

Floppy / Mass Storage Interface

The original floppy hardware is not emulated on the CoCo3FPGA. Instead the floppy interface is replaced with a serial port that communicates with a file server. This communications port was adapted from Terence J. Boldt's Serial Virtual Drive (SVD) for the Apple IIe at apple2.boldt.ca. This is similar to the DriveWire by cloud9tech.com except that the serial communications is handled by a UART compatible with the Motorola MC6850 ACIA. Because the port does not operate like the disk controller, the address is also not the same as the disk controller. The ACIA is located at addresses $FF50-FF52 and communicates at a speed of 115200 bps. The Tandy / Radio Shack DISK BASIC version 1.1 has been modified to work with the serial port. The modifications allow reading and writing over the ACIA to disk images stored on the server machine. Up to 4 disks are available under DISK BASIC. There is no DISK BASIC support for drives larger than the standard floppy drive:

35 tracks * 18 sectors/track * 256 bytes/sector = 161280 bytes

The disk images contain only the 256 byte sectors data from the disks. No headers or format information is needed. An image that can be used for DISK BASIC is included with the CoCo3FPGA distribution. It contains some files that are readily available on the internet.

Two additional disk images, disk 5 and disk 6, are available with other software. This other software includes NitrOS-9. These disk images are intended to be used as Hard

Disk Drives under NitrOS-9. There is one BASIC program on the DOS disk, MODBOOT, that allows DISK BASIC to boot a different DOS from any of the drives. The **DOS** command under DISK BASIC was modified to read from disk 5 and track 68. This is a special feature that can be utilized under NitrOS-9.

The ACIA uses three addresses. The 6850 command / status byte is located at address $FF50. The read / write data port is located at address $FF51. The other address is used for error detection. In the event that a communications error is encountered, meaning DISK BASIC continuously waits for a character that will never arrive; a timer is used to send a NMI to the CPU that aborts the sector read and write. This timer is set by writing anything other than 0 into port address $FF52. The length in seconds of the timer is (255-XXX)/60, where XXX is the value written into the timer register. The standard timeout value is 128. This gives a time out of approximately 2 seconds. DISK BASIC was written to allow a failed read / write to be retried 5 times before returning an error to the user.

PC Server

The server program is included in the CoCo3FPGA distribution. This program was written as a 32bit Windows console mode program. No other OS is supported. If no parameters are given on the command line, the program will print out the help message:

```
Apple][e / CoCo3 Serial Drive (c)2008 Gary L. Becker
Based on Apple2VirtualDrive (c)2001 Terence J. Boldt
Note:  This program requires a null modem connection
       to an Apple][e/CoCo3 FPGA at 115200 bps.

USAGE:  SERVER COMx [TYPE[:SIZE] FILENAME]
                        TYPE
                         /A*              Apple Hard Disk
                         /R**             CoCo Slot 3 ROM
                         /0               CoCo3 Floppy Disk 0
                         /1               CoCo3 Floppy Disk 1
                         /2               CoCo3 Floppy Disk 2
                         /3               CoCo3 Floppy Disk 3
                         /4               CoCo3 Hard Disk 0
                         /5               CoCo3 Hard Disk 1
                         /N:XXXXXX        XXXXX = Hex value of disk size in sectors

 *   Size is ignored for Apple Hard Disk Drives\n");
 ** Size is autodetected for CoCo Slot 3 ROM\n");
```

The first parameter for the com port is required. It can be any Windows compatible communications port. But, I have not had luck with USB based ports. The other parameters will depend on the configuration you want to run. When I am running DISK BASIC, I run the command line:

SERVER COM1 /0:0276 DOS1.DSK

This gives me a drive for the default disk 1 with DISK BASIC. The disk size is limited to 18 sectors per track, a single side, and 35 tracks. When I run NitrOS-9, I use the command:

SERVER COM1 /4:7E000 NITROS9.OS9

This sets up one disk for /d0 with 18 sectors per track, 2 sides, and 14,336 tracks.

The disk size is optional, but highly recommended. Only the drives you intend to use are required.

NitrOS-9

There are several features that only have support under NitrOS-9. The first feature is hard drive support. Using the same interface as the floppy, a much larger drive image can be used. The CoCo3FPGA includes an image that is just under 128 Mbytes. The hard drive support is setup as drive 4 and 5. Each hard drive can be the maximum allowed by NitrOS-9, but there are some programs that will fail with drives larger than 128 Mbytes, the maximum for 1 sector clusters. NitrOS-9 can also be setup to read smaller images from drives 0-3. These images are limited to 16 Mbytes. The included OS9 image will boot with /d0 and /d1, the 128 Mbyte drives 4-5. The default drive, /dd, points to drive /d0. There is also one smaller drive, /f0, that is sized to match a 3.5 inch 720K floppy image. This drive can be used to read disk images from single side / double side 5.25 inch floppies also.

The image files are only as large as the highest numbered sector written. The server opens all files in update mode and will increase the file size as needed. The server does not know anything about tracks or sides. It deals only with absolute sector numbers. Drives 0-3 can be sized up to $10000 sectors (67,108,864 bytes); and drives 4-5 can be up to $1000000 sectors (17,179,869,184 bytes).

The booting process of NitrOS-9 required two different “drivers” for this interface. The boot_svd4 included on the OS9 disk, boots the system from drive 4. Another file, boot_svd0, allows booting from the smaller floppy sized image at drive 0. These files are located in the NitrOS9/MODULES/BOOTTRACK directory. The second driver file is located in the NitrOS9/MODULES/RBF directory. This is the real NitrOS-9 driver, svddsk.dr. There are several device descriptor files in this same directory

Filename	Drive
dd0svd.dd	/dd for drive 0, 720K
dd4svd.dd	/dd for drive 4, 128M
d0svd.dd	/d0 for drive 4, 128M
d1svd.dd	/d1 for drive 5, 128M
f0svd.dd	/f0 for drive 0, 720K

The second feature is a real time clock. This clock uses the CLOCK2_JVEMU module. The port addresses are:

Port Address	Value

$FFC0	Century
$FFC1	Year
$FFC2	Month
$FFC3	Day
$FFC4	Day of Week
$FFC5	Hour
$FFC6	Minute
$FFC7	Second

The hardware implementation for this clock is not complete. The ability to keep track of all changes up to the Century takes too much logic. Instead only the logic to update seconds, minutes, hours, and days is implemented. The days do not have limit correction, so you could have a day 32 in a month or an eighth day of the week. Instead a method for updating the clock has been devised. Every time LSN0 is read from either drive 4 or 5 along with the sector, the PC will send the current time. Both the boot_svd and svddsk.dd files are written to receive this time and update the real time clock. This sector is read a lot, so the time will never be too far off. But it does depend on the current time being correct on the file server.

The standard NitrOS-9 make boot script, mb located in the NitrOS9/SCRIPTS directory, will not work with hard drives. The format command asks an additional question that is not asked for floppies. Instead, the format command will need to be entered manually. When the format asks “Both PHYSICAL and LOGICAL format?” the correct answer is no. The Physical format takes a long time and does not read nor write to the drive. I do not recommend saying yes to the question “Physical Verify desired?” either. This takes an enormous amount of time and does not change anything in the server image file. The updated script, buildhd located in the same directory, will create the new bootable drive. This script utilizes the gary.bl boot list file, located in the NitrOS9/BOOTLISTS directory. This file can be edited to modify the boot disk as desired. The svdgen module located in the CMDS directory is a based on os9gen and has been modified to allow it to work with some hard disk drives. The limitation is the LSN of the last sector of the boot track needs to be less than sector 6144. Since the device descriptors for all the SVD disks use 18 sectors per track and double track density, os9gen will write 18 sectors starting with sector 1224. Other number of sectors per track can be used, but the DISK BASIC will need to be modified to allow booting from another absolute sector. It was easier to use 18, the same value as floppy drives, because the DOS command in DISK BASIC assumes 18.

As with DECB, the keyboard follows the PS/2 keyboard layout. But because NitrOS-9 uses some additional symbols, some keys operate differently when the control key is also pushed. See additional information in the Getting Started with NitrOS-9 manual. The PS/2 has these key available. CoCo3FPGA will force the special control + key combination when possible.

Implementing the Design

The CoCo3FPGA was designed using the Digilent Spartan 3 Starter board with a 1000K gate FPGA. If the design is to be implemented on a different board, the code will need to be ported to the new board. The block RAM implementation is proprietary to Xilinx and will need to be modified to use a FPGA from another vendor. All the code necessary to recreate the design is included in the distribution. Also included in the design is the user constraint file, COCO3FPGA.UCF. This file sets the FPGA pin outs for the Digilent Spartan 3 Starter board.

Known Bugs and Limitations

The CPU core is not cycle accurate. The core runs some Opcodes in fewer cycles than the standard 6809. Other instructions run with more cycles than the standard 809. This bug shows up in routines that count CPU cycles for timing purposes. They may not run correctly with CoCo3FPGA.

The VGA screen timing is not exactly the same as the original CoCo3. Every effort has been made to ensure the timing is as close as possible. This big will show up in programs that depend on the screen timing.

The serial communications was written with no data integrity included, and the floppy / hard disk "server" does not require files to be their maximum size. These issues show up as two separate bugs. The first bug is the file size can grow to a very large, if the disk size is not included in the server command line. Because the maximum file size could be 17179869184 bytes, if one of the sector addresses gets received incorrectly, the file size can greatly be increased. The second issue shows up during system disk copy operations. The "server" requires additional time to increase the size of the file. CoCo3FPGA does not account for this additional time, and can fail operations where very large amounts of data are copied from one drive to another. Slowing down the CPU speed of CoCo3FPGA will help, but will not completely solve the issue.

The Orchestra-90 CC card has the capabilities to load and save files from a floppy. Because CoCo3FPGA does not implement the floppy hardware and the routines for reading and writing floppies are in the ROM, the feature to read and write floppies does not work. It would require a hack of the ROM to allow it to completely work with the CoCo3FPGA interface.

Any new bugs should be reported to the CoCo3FPGA groups at Yahoo Groups. In this group, you can find the latest information and newest version of CoCo3FPGA.