CoCoDEV

6809 development board

design by Dave Philipsen
(12-29-2020 revision)

ROUGH DRAFT
for review

This document is a work in progress and subject to change. Please email me with any questions, corrections, etc.

dave@davebiz.com

CONTENTS

INTRODUCTION

CoCoDEV is an FPGA development board based upon the 6809 CPU with 512K SRAM, MMU, VGA video output, real time clock, wifi, serial port, and SD card. The board bears some functional resemblance to a Tandy (Radio Shack) Color Computer 3 in that it does contain some internal registers that mimic the MMU, the video base address can be manipulated, and IRQs can be generated from the vertical sync of the video and enabled/disabled. This functionality is what allows it to a version of NitrOS9 Level 2 which is just slightly modified from the CoCo 3 version.

The basic objective of the board is to provide a platform where the user has the ability to more easily interface to external devices. A number of I/O pins and a generous prototyping area with over 1,000 pads placed on a .1" grid is provided. Thus, the user can easily interface to LEDs, relays, stepper motors, servo motors, I^2C devices, etc. Additionally, signals can be brought out to an optional terminal block on the rear of the board for easy access.

The heart of the board is the Waveshare EP2C5 FPGA breakout board. The board is programmed with the CPU core (MC6809) and the logic for the various elements including VGA, keyboard, memory management unit (MMU), UARTs, and I/O.

Power is supplied by a regulated 12vdc switching power supply with capability to deliver approximately 18 watts. This voltage is regulated down to 5vdc and 3.3vdc and the voltages are available on specific pins to the user.

QUICK START

Starting OS9 without Disk BASIC

CoCoDEV is shipped with the Tandy Color Computer ROMs. If the ROMs are uninstalled, CoCoDEV will simply boot into a ROM monitor upon power-up. Once the ROMS are added it will boot into Disk BASIC on power-up. Since NitrOS9 is a freely available operating system based upon the original OS9 operating system, it is also included with CoCoDEV on an SD card without charge.

When you power-up the CoCoDEV board your options will be a basic 6809 monitor, the NitrOS9 operating system, or Disk BASIC.

Ok, let's get on to powering up CoCoDEV for the first time! Here's what you'll need:

1) CoCoDEV (obviously)
2) microSD card with NitrOS9 installed (included)
3) 12vdc power supply (included)
4) VGA monitor and connection cable(not included)
5) USB keyboard (not included but available from Dave Philipsen)

Prepare a work area for your CoCoDEV system which will include space for the VGA monitor, keyboard, and CoCoDEV itself. Make sure that the included NitrOS9 micro SD card is plugged into the micro SD socket located on the front panel of CoCoDEV (this is a push-push type SD socket). You will need access to two electrical outlets; one for the CoCoDEV power supply and the other (presumably) for your VGA monitor. Plug in the monitor and the CoCoDEV power supply. Next plug in your USB keyboard to the USB connector on the back of CoCoDEV.

It is important to remember that even though this is a USB connector the underlying system is not a USB port. It is a PS/2 port using a USB connector. Many inexpensive USB keyboards fall back automatically to the PS/2 standard. If you do not have such a keyboard, you may use a PS/2 keyboard with an adapter that adapts the mini-DIN connector to a USB connector.

Using the VGA connection cable, plug one end into your VGA monitor and the other end into the 15-pin high density D-sub connector on the back of CoCoDEV. Next, plug the power supply into the power connector on the back of CoCoDEV. As soon as the VGA monitor synchronizes to the VGA signal from CoCoDEV you should see text on the screen which says something like this:

Disk Extended Color BASIC 1.1
Copyright (c) 1982 byTandy
Under license from Microsoft

CoCoDEV FW 1.0 HW 1.0
12/29/20 13:00

OK

_

This is a modified version of the Color Computer Disk Extended Color BASIC. Unmodified versions of the Color BASIC, Extended Color BASIC, and Disk BASIC ROMs are present in CoCoDEV. These ROMs are copied to RAM when CoCoDEV is originally booted. At that time, a 'patch' program runs and patches parts of these ROMs (now running in RAM) for the particularities of the CoCoDEV hardware. Notably, the patches include changes for the 80 column display, addition of a TIME$ variable to read the RTC from BASIC, and accessing the micro SD card instead of real floppy drives.

There are many articles available on the internet that will familiarize you with the commands available in Disk Extended Color BASIC. Just remember that the CoCoDEV has a slightly modified version of BASIC and most commands for manipulation of graphics modes, etc. should not be used since those modes are not supported by CoCoDEV.

COMPONENTS

512K SRAM

A single 512kB SRAM chip with fast, 10ns access is interfaced to the FPGA and thus to the CPU and video cores. Both the CPU and the video system access the memory directly without wait states. The CPU has access to the memory when the E clock is low and the video system accesses the memory when that clock is high.

Since the CPU can only address up to 64kB of memory, a memory management unit is employed to subdivide the 512kB SRAM into 64 blocks of 8K each. The address space of the CPU is divided into 8 'slots' of 8K each. Thus, the MMU can place any of the 64 blocks into any of the CPU slots. In fact, the same block could occupy more than one of the CPU slots. Therefore, the CPU can access the entire 512kB of memory by manipulating the MMU registers and swapping in and out various blocks of memory as needed.

Memory management unit

The MMU is controlled by several I/O registers. A close examination of this system reveals that it is very closely related to the MMU of the TRS-80 Color Computer. The following chart explains the pertinent registers and their I/O addresses:

$FF90 Initialization Register 0 - INIT0

Bit 7	Not used	
Bit 6	MMUEN	1 = MMU enabled
Bit 5	IEN	1 = IRQ enabled
Bit 4	Not used	Reserved for future FIRQ enable
Bit 3	MC3	1 = RAM at FExx is constant (secondary vectors)
Bits 2-0	Not used	

$FF91 Initialization Register 1 – INIT1

Bit 7-1	Unused	
Bit 0	TR	MMU task select 1 = enable FFA8-FFAF MMU registers 0 = enable FFA0-FFA7 MMU registers

$FFA0-FFA7 MMU bank registers (task zero)

FFA0	Bank at $0000-1FFF
FFA1	Bank at $2000-3FFF
FFA2	Bank at $4000-5FFF
FFA3	Bank at $6000-7FFF
FFA4	Bank at $8000-9FFF
FFA5	Bank at $A000-BFFF
FFA6	Bank at $C000-DFFF
FFA7	Bank at $E000-FEFF (or $E000-FDFF if secondary vectors enabled)

$FFA8-FFAF MMU bank registers (task one)

FFA8	Bank at $0000-1FFF
FFA9	Bank at $2000-3FFF
FFAA	Bank at $4000-5FFF
FFAB	Bank at $6000-7FFF
FFAC	Bank at $8000-9FFF
FFAD	Bank at $A000-BFFF
FFAE	Bank at $C000-DFFF
FFAF	Bank at $E000-FEFF (or $E000-FDFF if secondary vectors are enabled)

FFDE	ROM mode. Any write switches ROM ($F000 - FFFF) into memory map.
FFDF	RAM mode. Any write selects all-RAM mode.

MMU registers $FFA0-FFA7 are enabled when the task bit ($FF91) is clear. MMU registers $FFA8-FFAF are enabled when the task bit is set. These registers allocate blocks of 8K into the CPU's 64k address space. Valid bank ranges are $00-3F.

Additionally, when the MC3 bit ($FF90) is set, the RAM at $FE00-FEFF is constant in the last MMU block (either FFAF or FFA7). When enabled, it is always mapped from block number $3F.

4MB Flash memory

A 4MB serial (SPI) flash memory is employed. The registers and I/O addresses of the SPI controller are shown below:

$FFB4 SPI command register

Bits 7-0	When written to, this register can be used to send commands to the SPI controller and to send data to the controller. When read, this register is used to read data from the SPI interface.

$FFB5 SPI control/status register

Bits 7-0	When written to with any value a terminate command is sent to the SPI controller. When read, bit 7 contains the status, 0 = Ready, 1 = Not ready

The following memory map shows which data is stored at various locations in the flash chip:

$000000 – 001FFF Boot code (8k)
$002000 – 003FFF Color BASIC (8k)
$004000 – 005FFF Extended BASIC (8k)
$006000 – 007FFF Disk BASIC (8k)
$008000 – 009FFF Patch code (8k)
$00A000 – 00FFFF Reserved (24k)
$010000 – 01FFFF Primary OS9 failsafe boot (64k)
$020000 – 02FFFF Secondary OS9 failsafe boot (64k)
$030000 – 03FFFF Alternate operating system (64k)
$040000 – 07FFFF Read only OS9 storage (accessed as device /fl0 - 256k)
$080000 – 0FFFFF BASIC program storage (accessed via FLOAD, FSAVE, FDIR)
$100000 – 3FFFFF Unused (3,076k)

For specific information on how to control the flash chip via the SPI interface see the datasheet for the Winbond W25Q32 chip

Real time clock

A real time clock module contains the Maxim DS3231 chip as well as a 1K EEPROM. Both chips are accessed via the I²C bus. The registers and I/O addresses of the I²C controller are shown below:

$FF80 I²C control/status register

Bit 7	Write: 0 = Write data, 1 = Read data (control register)
Bit 0	Write: 0 = Disable interface, 1 = Enable interface (control register) Read: 0 = Not busy, 1 = Busy (status register)

$FF81 I²C read/write register

Bits 7-0	Read/write data to interface

$FF82 I²C slave address register

Bits 7-0	Read/write slave address

VGA interface

The VGA interface consists of a high-density female DE-15 connector. The pinout is identical to the industry standard VGA pinout. Therefore, this interface should drive any VGA standard monitor. The pixel clock frequency is 25 MHz which is slightly slower than the standard 25.175 MHz but still accepted by all monitors. The video display has a horizontal resolution of 640 pixels and 480 vertical lines non-interlaced (progressive scan) at a 60 Hz frame rate. The interface has the ability to drive the intensities of the primary colors such that a total of 256 colors may be produced. This is achieved by the 3-3-2 method with 3 bits of color depth for RED, 3 bits for GREEN, and 2 bits for BLUE.

Currently, the only display mode used is the 80 column x 30 row text mode. Each character cell is defined by two bytes of display memory. The first byte contains the character code and the second byte contains the attribute. The character code maps to a font table known as code page 437 which is the character set of the original IBM PC. This font is an 8x16 pixels-per-character font that is stored in the controller's non-volatile memory. The font contains various ASCII characters, numerals, upper and lower case letters, various international characters, line drawing characters, and other symbols. The second byte contains attributes that affect how the character is displayed. The table below describes the bits:

Text mode attribute byte

Bit 7	Flash (1 = flash, 0 = no flash)
Bit 6	Underline (1 = underline, 0 = no underline)
Bits 5-3	Three foreground color bits (R, G, B)
Bits 2-0	Three background color bits (R, G, B)

The following registers are used to establish the base address of the video display. The base address may be located anywhere within the physical 512 kB memory map from $00000 to 7FFFF:

$FF9D Vertical offset register MSB

Bits 7-0	Y15-Y8	MSB start of video in RAM (video location * 2048)

$FF9E Vertical offset register LSB

Bits 7-0	Y7-Y0	LSB start of video in RAM (video location * 8)

Y15-Y0 are used to set the video to start at any location in the 512 kB range in steps of 8 bytes.

A possible future enhancement would be a graphics mode supporting the 640 x 480 resolution with 256 colors.

Keyboard

They keyboard connector is a standard USB-A female connector that will accommodate many USB keyboards. However, the interface itself does not use the USB communication standard. Instead, it relies upon the fact that many USB keyboards automatically fall back to the IBM PS/2 standard. So with the proper adapter the interface will support standard PS/2 and by extension the older IBM AT keyboards as well.

The keyboard controller converts many of the keys to their equivalent ASCII key codes to be presented to the keyboard data register. Other keys are assigned values so that they can be used by user programs. Additionally, the status of the ALT, SHIFT, CTRL, and Windows keys may be read according to the registers below:

$FFD0 Keyboard status register

Bits 7-0	Read only: 0 = no data available

$FFD1 Keyboard data register

Bits 7-0	Read only: Keyboard data (ASCII & others)

$FFD2 Keyboard aux keys register

Bit 7	Not used
Bit 6	Right Alt (1 = down, 0 = up)
Bit 5	Left Alt (1 = down, 0 = up)
Bit 4	Right Ctrl (1 = down, 0 = up)
Bit 3	Left Ctrl (1 = down, 0 = up)
Bit 2	Right Shift (1 = down, 0 = up)
Bit 1	Left Shift (1 = down, 0 = up)
Bit 0	Caps Lock (1 = down, 0 = up)

Wifi

An inexpensive ESP8266-01 WiFi module is used to provide access to the internet or internal networks. It is connected to the 2nd serial port. The module is flashed with modified software (Zimodem by Bo Zimmerman). The current release is version 3.4.5 Dave's Zimodem Version. Zimodem is provided as an open source offering from Bo Zimmerman (http://www.zimmers.net).

Serial port

The serial ports are modified versions of the MC6850 UART. Because of the addition of a baud rate generator and a 2048-byte receive FIFO buffer, I refer to these serial ports as “Super 6850s”. The UART require four I/O addresses. For normal (mode 0) operation, the base address when written to is the control register and when read from is the status register. The base address + 1 when written to is the transmit data register and when read from is the receive data register. The next two registers at base address + 2 and base address + 3 are read-only registers that contain the MSB and LSB respectively of an 11-bit number which indicates how full the receive FIFO is. A second mode of operation (mode 1) allows access to the baud rate generator without using additional I/O addresses. The UART can accept a baud rate addend that is stored MSB to the base address and LSB to base address + 1. To place the UART in mode 1 to accept the baud rate addend, write $02 to the control register. The next two writes to the UART must be in this order: **1**) Write the MSB of the addend to base address, **2**) write the LSB of the addend to base address + 1. After the last write to base address + 1 the UART returns to normal operation (mode 0).

The UART always operates in 8-N-1. That is 8 data bits, no parity, and 1 stop bit. No handshaking is implemented. Currently, interrupts are not enabled on the UART which means that it operates in a 'polled only' fashion. Since the UART has such a large receive buffer, terminal programs can easily be written even in high level languages like BASIC which do not require interrupts to be used.

Base address $FF68 is the on board LVTTL (3.3v low voltage transistor transistor logic) serial port found on pin header J5. Base address $FF6C is the WiFi module plugged into the header location M1.

$FF68/FF6C (Write) UART control register

Bit 7	Receive interrupt enable (1 = enable, 0 = disable)
Bit 6	Not used
Bit 5	Not used
Bit 4	Not used
Bit 3	Not used
Bit 2	Not used
Bit 1 - 0	Reset / mode control 10 = set baud rate, 11 = reset UART

$FF68/FF6C (Read) UART status register

Bit 7	Interrupt request
Bit 6	Not used
Bit 5	Not used
Bit 4	Not used

Bit 3	Not used
Bit 2	Not used
Bit 1	Transmit data register empty (1 = empty, 0 = full)
Bit 0	Received data register full (1 = full, 0 = empty)

$FF69/FF6D (Read/Write) UART data register

Bit 7-0	Data

Baud rate addend

Bit rate	Addend (in hex)	Actual bit rate	Error
921,600	4B80	921631	.003%
750,000	3D71	750,017	.002%
500,000	28F6	500,011	.002%
460,800	25C0	460,816	.002%
250,000	147B	250,006	.003%
230,400	12E0	230,408	.003%
115,200	0970	115,204	.003%
57,600	04B8	57,601	.003%
38,400	0325	38,385	-.040%
19,200	0193	19,217	.080%
9,600	00C9	9,584	-.170%
4,800	0065	4,816	.330%
2,400	0032	2,384	-.660%
1,200	0019	1,192	-.660%
600	000D	620	3.310%

Note: Do not attempt to set the baud rate of the UART below 600 or above 921,600.

Inputs / Ouputs

CoCoDEV has a total of 24 input/output pins. The majority of these are configurable as either inputs or outputs through a synthesized MC6821 PIA interface. It is important to remember that all of them are 3.3v LVTTL logic. A 3.3v LVTTL logic output pin will drive the inputs of 5v TTL logic but **the outputs of 5v TTL logic will cause irreparable damage to 3.3v TTL logic inputs!** Therefore, use great caution when interfacing to 5v TTL logic. Level shifters may be needed depending in your application. I/O pins PA0 – PA6 are clearly marked on pin header J4. PA7 is the pin labeled LED3 since LED3 of the FPGA board is connected to this pin. I/O pins PB0 – PB5 are also clearly marked on the same pin header. Pins PB6 and PB7 correspond to LED1 and LED2 respectively since LED2 and LED3 on the FPGA board are connected to these pins.

$FF10 (Read/Write) Peripheral Register A / Data Direction Register A

	When CRA bit 2 = 0	When CRA bit 2 = 1
Bit 7	PA7 (LED3)	Data direction for PA7 – 0 = input, 1 = output
Bit 6	PA6	Data direction for PA6 – 0 = input, 1 = output
Bit 5	PA5	Data direction for PA5 – 0 = input, 1 = output
Bit 4	PA4	Data direction for PA4 – 0 = input, 1 = output
Bit 3	PA3	Data direction for PA3 – 0 = input, 1 = output
Bit 2	PA2	Data direction for PA2 – 0 = input, 1 = output
Bit 1	PA1	Data direction for PA1 – 0 = input, 1 = output
Bit 0	PA0	Data direction for PA0 – 0 = input, 1 = output

$FF11 (Read/Write) Control Register A (CRA)

Bit 7	IRQA1 flag
Bit 6	IRQA2 flag
Bit 5 - 3	CA2 control
Bit 2	Data direction register access (A)
Bit 1 - 0	CA1 control

$FF12 (Read/Write) Peripheral Register B / Data Direction Register B

	When CRB bit 2 = 0	When CRB bit 2 = 1
Bit 7	PB7 (LED2)	Data direction for PB7 – 0 = input, 1 = output
Bit 6	PB6 (LED1)	Data direction for PB6 – 0 = input, 1 = output
Bit 5	PB5	Data direction for PB5 – 0 = input, 1 = output
Bit 4	PB4	Data direction for PB4 – 0 = input, 1 = output
Bit 3	PB3	Data direction for PB3 – 0 = input, 1 = output
Bit 2	PB2	Data direction for PB2 – 0 = input, 1 = output
Bit 1	PB1	Data direction for PB1 – 0 = input, 1 = output
Bit 0	PB0	Data direction for PB0 – 0 = input, 1 = output

$FF13 (Read/Write) Control Register A (CRB)

Bit 7	IRQB1 flag
Bit 6	IRQB2 flag
Bit 5 - 3	CB2 control
Bit 2	Data direction register access (B)
Bit 1 - 0	CB1 control

Micro SD card

CoCoDEV ships with a 4GB micro SD card that has NitrOS9 installed on it. The following technical information will help you in understanding its layout:

The first partition is the primary NitrOS9 partition starting at logical sector $000000. The descriptor for this device (/dd) indicates that it contains $7100 cylinders (tracks) and there are $0012 sectors assigned to each track. This partition is approximately 128MB in size and contains 520,704 sectors arranged in 1-sector clusters. Under this partition, all data is stored in 512-byte sectors and the NitrOS9 operating system uses a de-blocking scheme so that the data appears as 256-byte sectors.

The second partition is the primary partition for BASIC and starts at logical sector $xxxxxx. This partition contains 1,000 virtual floppy drives for use by Disk BASIC. Under this partition, a de-blocking scheme is not used. All data is stored in the first 256 bytes of each 512-byte sector. Therefore, the remaining 256 bytes of the sector is simply wasted.

The third partition is the secondary NitrOS9 partition starting at logical sector $0D9DC0. The descriptor for this device (/sd1) indicates that it contains $C000 cylinders (tracks) and there are $00FF sectors assigned to each track with the exception of track 0 which contains $0012 sectors. This partition is approximately 3GB in size and contains 12,533,523 sectors arranged in 32-sector clusters.

Currently the /sd2 partition is not defined. Do not attempt to use partition /sd2 or format it.

Interrupts

The CPU may be interrupted from the video (vertical sync). The register at $FF92 is used to enable standard IRQ interrupts from this source.

$FF92 Interrupt request enable register - IRQENR

Bit 7-6	Not used	
Bit 5	TMR	1 = Enable timer IRQ (not implemented)
Bit 4	HBORD	1 = Enable horizontal border IRQ (not implemented)
Bit 3	VBORD	1 = Enable vertical border IRQ
Bit 2	SERIAL1	1 = Enable UART1 IRQ (not implemented)
Bit 1	SERIAL2	1 = Enable UART2 (WiFi module) IRQ (not implemented)
Bit 0	Not used	

$FF93 Fast interrupt request enable register – FIRQENR (not implemented)

Bit 7-6	Not used	
Bit 5	TMR	1 = Enable timer FIRQ (not implemented)
Bit 4	HBORD	1 = Enable horizontal border FIRQ (not implemented)
Bit 3	VBORD	1 = Enable vertical border FIRQ
Bit 2	SERIAL1	1 = Enable UART1 FIRQ (not implemented)
Bit 1	SERIAL2	1 = Enable UART2 (WiFi module) FIRQ (not implemented)
Bit 0	Not used	

Prototyping area

A fairly large prototyping area of approximately 4.2” x 2.2” is provided with over 1,000 gold-plated through holes. **The important thing to remember in regard to prototyping with the CoCoDEV board is that all I/O including the serial ports and I²C port is at low-voltage TTL levels of 0 – 3.3 vdc. Most circuitry using 5v TTL will accept 3.3v signals at their inputs without problems. However, applying a 5v TTL signal to any pin configured as an input on CoCoDEV without some sort of level translation will result in damage!**

The provided 12v power supply provides the board with approximately 18 watts of power. The core of CoCoDEV uses less than 2 watts leaving approximately 16 watts for user projects.

NitrOS9 Level 2

A version of the popular NitrOS9 operating system designed for the MC6809 will run on CoCoDEV. This version is based upon the CoCo 3 6809 version of the operating system with minimal modifications. The board is designed to boot the system from either flash memory or SD card. As shipped, the CoCoDEV board has NitrOS9 already installed on the flash chip and the micro SD card. Using the 'DOS' command from BASIC will boot NitrOS9 (Note: This command simply loads and executes the BOOT.BIN file that is present on drive 0).

The system contains a failsafe system to restore OS9 to the SD card by means of a backup operating system stored in the flash memory. However, at this time no user program exists to access the backup system. An alternate machine language program will be provided soon that will allow you to boot CoCoDEV from the flash memory. Booting from the flash chip will allow formatting of the SD card and installing files there to allow the system to boot from the SD card. A minimum size of 4GB is recommended for the SD card as the first 300MB are reserved for use by Disk Extended Color BASIC. The first NitrOS9 partition on the SD card starts at logical sector $9A4C0 and continues through sector $D9DBF. The size of the partition is about 128 MB and the clusters are one sector each. The second NitrOS9 partition starts at logical sector $D9DC0 and continues through sector $8D1CC9. The size of the partition is about 4 GB and the clusters are 32 sectors each.

The version of NitrOS9 for the CoCoDEV uses a modified version of Brett Gordon's excellent boot loader to boot the system. This eliminates the need for a special process to generate a “boot track” loader. As long as the OS9Boot file is located in the root directory of the first SD card partition, the system will be bootable. The boot loader program is contained within the system ROM (monitor) and as a file named BOOT.BIN on the first virtual disk drive (0).

Note that the only window type supported by CoCoDEV is the 80-column 30-row text window. Graphics modes, 40-column windows, and 32-column windows are not supported. The default startup file under NitrOS9 initializes an alternate window that may be accessed with the F12 key. The default boot includes descriptors for windows W1-W7 in addition to the default Term window.

For the most part, documentation found in *NitrOS-9 Technical Reference, NitrOS-9 Operating System User's Guide,* and *The NitrOS-9 Level 2 Windowing System* apply.

CoCoDEV Specifications

Board size:	7.185” x 5.05”
Power supply:	12vdc @ 1.5A
Keyboard:	PS/2 keyboard using USB-A female connector
VGA:	640x480 standard resolution with 25 MHz dot clock
WiFi:	ESP8266 module using Zimodem firmware
Realtime clock:	DS3231 module with battery backup
Non-volatile memory:	4 kB EEPROM (24C32) for parameter storage
Flash memory:	4 MB flash memory (Winbond 25Q32)
Input / output:	Two 8-bit ports configurable as 6821 PIA
Removable media:	micro SD card slot with supplied 4GB card
Header connectors:	Two 22-pin single-row male headers (.1” spacing), One 10-pin single-row male header (.1” spacing)
Optional connectors:	Power supply terminal block (5.00mm spacing), I/O terminal block – 10 terminals (5.00mm spacing)
Power consumption:	Base system 12vdc @ 150ma (secondary) 120vac @ 23ma (primary)