CC1 - Handheld

Coincidence Counter

USER'S MANUAL

Rev. 1.02

www.qubitekk.com

Contents

Principle of Operation

1.1 Overview

The CC1 is a low-cost, handheld coincidence counter intended for use with single photon counting experiments. The unit can detect and count TTL electrical pulses with nanosecond resolution and a maximum count value of 2,097,152 pulses per channel. Pulses that occur simultaneously on Channel 1 and Channel 2, within a specified “coincidence window,” are treated as “coincident” pulses and are measured and displayed by the unit (max coincidence count equal to 2,097,152).

An LCD display on the front of the unit allows easy viewing of all measured single and coincidence counts. This display, in addition to the push buttons on the front of the unit, provides a simple interface for displaying results and setting up measurements. A USB connection on the side of the unit provides power to the system while also allowing for optional serial control of the device by a computer.

The CC1 comes pre-programmed with all of the functionality described in this manual; however, it is capable of much more. The unit contains both a re-programmable microprocessor (for handling the user interface and serial communications) and a re-programmable FPGA (for handling high-speed pulse counting). Both of these devices can be programmed by the user to implement a wide range of features.

In the Appendix section, the user will find the necessary documentation for programming the CC1 for custom applications. Whether it is implementing a time histogram recorder, expanding/modifying the existing features of the CC1, or developing a custom device for use in implementing new cryptographic protocols, the CC1 is a flexible platform that will greatly simplify and speed your development effort.

1.2 What is Included

The CC1 Handheld Coincidence Counter ships with the following materials:

a) 2m USB cable (USB-B)
b) CC1 Coincidence Counter
c) Rabbit programming cable (2mm)
d) Retractable USB cable (USB-mini)
e) Development software and programming file
f) USB-to-AC power module

The various components (with their respective labels) are shown in Figure 1 on the following page.

Figure 1. Components included with CC1 Coincidence Counter.

1.3 Powering the Unit and Connecting Inputs

The CC1 is powered through its USB port. The USB cable provided with the system should be plugged into the side of the unit as shown in Figure 2.

Figure 2. Power over USB connection.

The other end of the USB cable can be plugged into a computer USB port or plugged into the included USB-to-AC power connector, as shown in Figure 3.

Figure 3. USB-to-AC Adapter.

Once powered, the CC1 will display a startup screen for five seconds on the unit's blacklit LCD. The unit's "Count Screen" will then be displayed as shown in Figure 4.

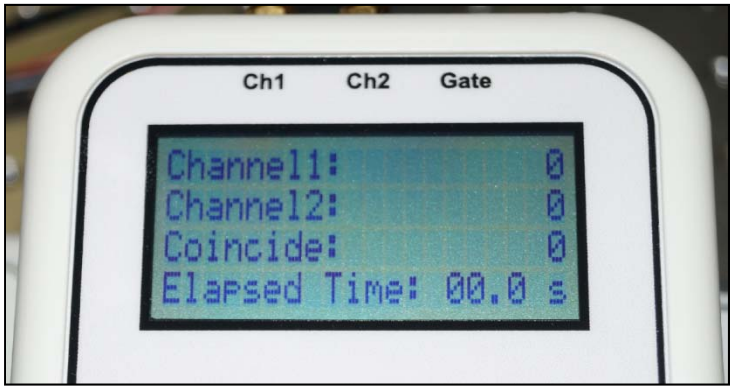


Figure 4. Count Screen on startup.

To connect signals to the CC1, three SMA connector jacks are available at the top of the unit. To detect coincident pulses, Channels 1 and 2 must both be connected to a pulse signal. If the counting of single and coincidence pulses should be gated (see next section for more details), then the third channel ("GATE") should be connected to the associated gate pulse.

1.4 Navigating the Menus

The CC1 has six buttons on its front face. The buttons, along with signal connectors and peripheral ports, are shown in Figure 4.

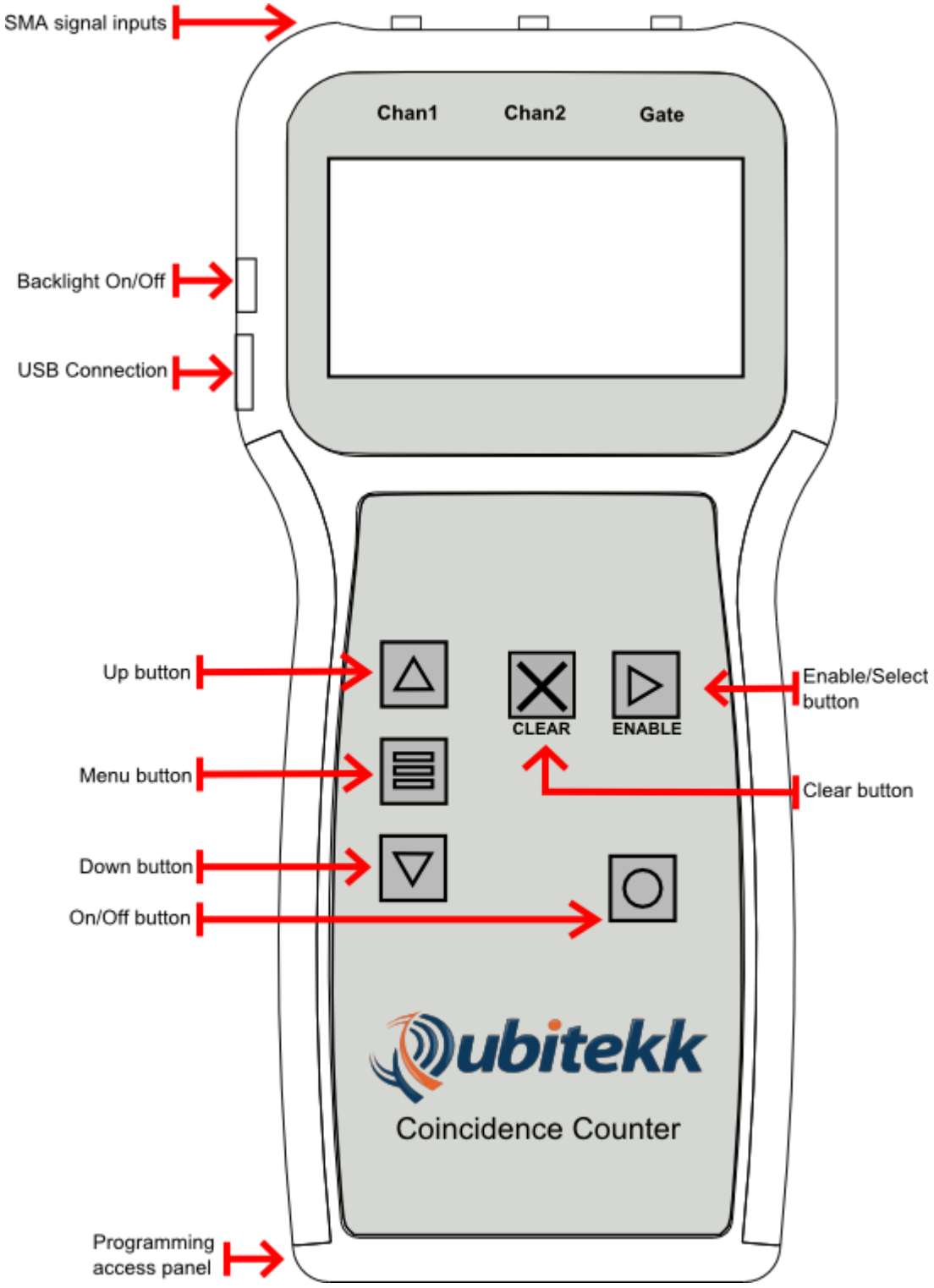


Figure 4. CC1 Buttons and Ports.

To configure the CC1 for a measurement, press the "menu" button. The CC1's Menu Screen will be displayed as shown in Figure 5.

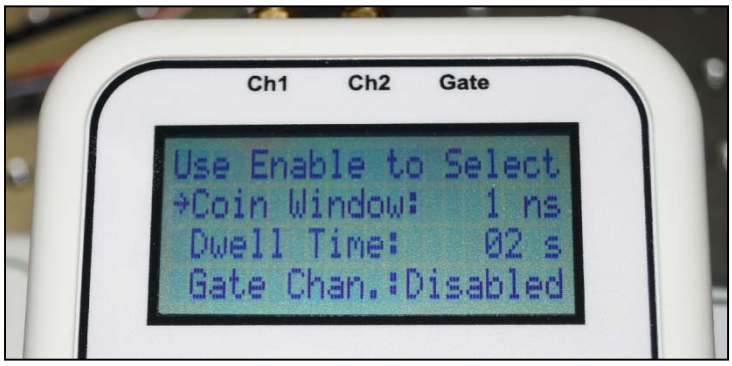


Figure 5. Menu Screen.

Pressing the "menu" button again will return the user back to the Count Screen.

The Menu Screen contains six settings that can be changed by the user. These settings will be covered in the following section. The various settings can be viewed by scrolling up or down in the menu using the "UP" and "DOWN" arrow buttons.

When a setting needs to be changed, the setting can be selected for modification by pressing the "ENABLE" button. The setting value will blink, indicating that it is ready for modification. Again, the UP" and "DOWN" arrow buttons can be pressed to change the setting value. Once complete, press the "ENABLE" button again to continue scrolling through the menu.

1.5 Setting Up a Measurement

Prior to any measurement, the measurement parameters for the CC1 should be configured. The next section of the manual will describe each parameter and discuss their impact on the measurement. The default settings for the CC1 (upon power up) are shown in Table 1 below:

Setting	Value
Coincidence Window	1 ns
Dwell Time	2 s
Gate Chan.	Disabled
Subtract Acc.?	No
Trigger	Start/Stop
CH1 Delay	0 ns
Firmware Ver	2.001

1.6 Making a Measurement

Once all measurement settings have been adjusted, the “menu” button should be pressed to return to the Count Screen. With the Count Screen displayed, press the “ON/OFF” button to start or end a count.

Once the “ON/OFF” button has been pressed, the details of the count measurement will be displayed on the LCD as shown in Figure 6.

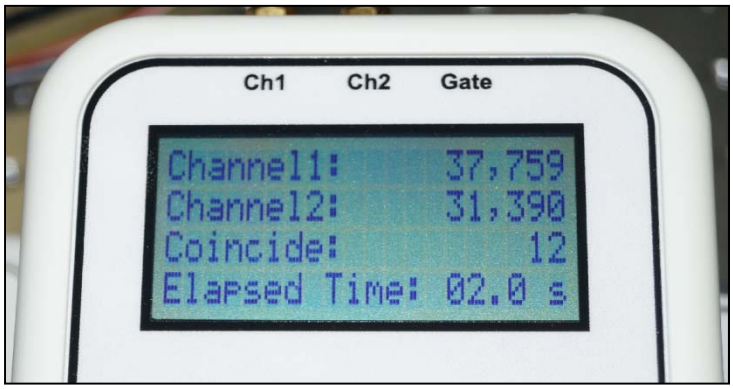


Figure 6. Count Screen after measurement.

Once the measurement is complete, the counts will remain on the screen until a new count is initiated.

NOTE: If the unit is being triggered in “Continuous” mode, the counts will only remain on the screen for a short period of time before resetting to zero and beginning the next count.

Settings

2.1 Description of Counting Process

Pulses detected on Channel 1, Channel 2, and the optional Gate channel will be counted as coincident pulses based on a number of settings made prior to the measurements. In general, the simplest measurement of coincidence is as shown in Figure 7:

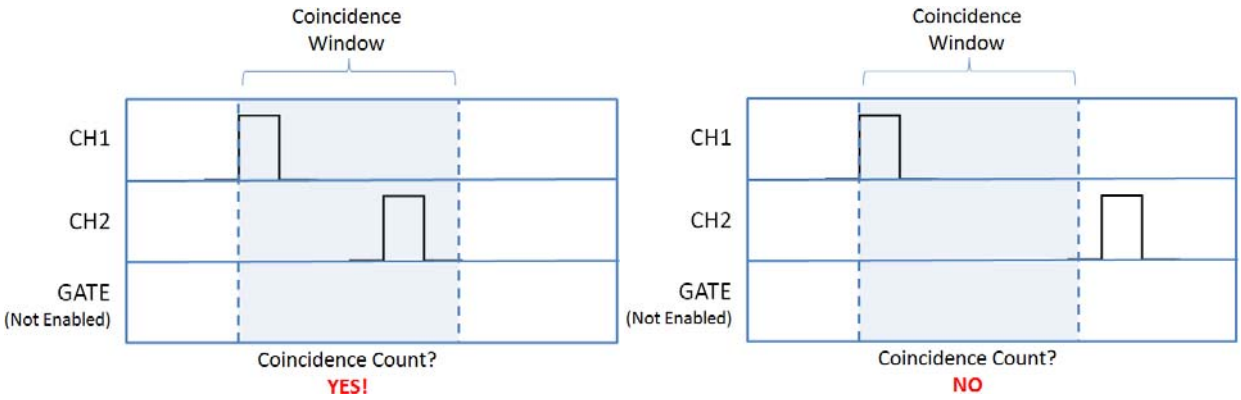


Figure 7. Simple coincidence measurement described.

Two pulses will be counted by the CC1 to be coincident if:

a. The time between the rising edges of the two pulses is less than the coincidence window

If the GATE signal is enabled, then the two pulses will only be counted by the CC1 as coincident if the following two conditions are satisfied:

a. The time between the rising edges of the two pulses is less than the coincidence window
b. The GATE signal is enabled during both rising edges

Figure 8 represents a coincidence measurement with the GATE signal enabled.

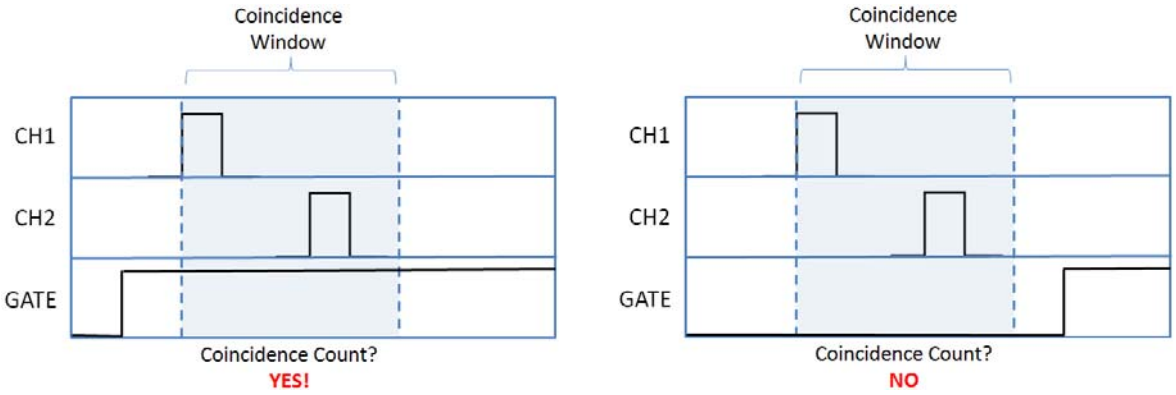


Figure 8. Coincidence measurement with a GATE signal.

With this understanding in place, the remainder of this section will look at how aspects of this measurement are specified in the CC1.

2.2 Coin Window

If pulses are simultaneously detected within some short time window, then they are said to be coincident. The CC1 refers to this time window as the Coincidence Window, or “Coin Window” for short. Technically, the Coin Window is the maximum allowed time between the rising edge of a pulse on Channel 1 and a pulse on Channel 2 before two pulses are no longer considered to be coincident.

The Coin Window can be set on the CC1 to be any integer value between 1 and 8 nanoseconds.

2.3 Dwell Time

The Dwell Time is the amount of time to accumulate pulses. For example, if the Dwell Time is 20 seconds, then all single pulses detected on Channel 1 during a 20 second measurement period would be totaled and displayed. This would also be true for Channel 2 and the coincident channel.

The Dwell Time can be set on the CC1 to values ranging between 0.1 second and infinity. When a measurement is initiated, the CC1 will accumulate counts for an amount of time equal to the Dwell Time. If the Dwell Time is infinity, the CC1 will accumulate counts indefinitely or until the ON/OFF button is pressed and the count process stopped.

2.4 Gate Chan.

The Gate Chan. setting determines whether or not the GATE channel performs any function. If the Gate Chan. value is set to “ENABLED,” single and coincidence counts will only be detected when the GATE signal is high. If the Gate Chan. value is set to “DISABLED,” the GATE signal will be ignored.

2.5 Subtract Acc.?

In some experiments, the CC1 may be used to measure coincidence counts to identify correlations between signals. Coincidence counts can often indicate the presence of some shared source of photon generation or optical phenomena. However, coincidence counts can also be the results of random chance.

Statistically, there is some probability that a random stream of pulses on Channel 1 will occasionally produce a pulse that is coincident with another random stream of pulses on Channel 2. This statistical likelihood is called the “accidentals” and it can be calculated using the following formula:

$$\text{Accidentals} = \frac{2(\text{Coincidence Window (sec)})(\text{Channel \#1 Singles})(\text{Channel \#2 Singles})}{\text{dwell time (sec)}}$$

Because accidentals can sometimes obscure the true correlations being sought in an experiment, it can be advantageous to have these accidentals automatically calculated and subtracted from the measured coincidence value.

The Subtract Acc.? setting allows the user to turn this calculation on or off. If the Subtract Acc.? value is set to “YES,” then the estimated accidentals will be calculated and subtracted from all coincidence measurements. If the Subtract Acc.? value is set to “NO,” then the full coincidence counts (with accidentals included) will be reported.

2.6 Trigger

The Trigger setting allows the user to select how a measurement will be started and stopped. If the Trigger setting is set to “Continuous,” then the measurements will start once the ON/OFF button is pressed. Once a count is initiated, the unit will accumulate counts for the set Dwell Time. Once the Dwell Time has been reached, the total counts accumulated will be shown on the counter’s display for a short period of time. Then the unit’s counters will automatically be reset and a new count initiated. This cycle will continue over and over until the ON/OFF button is again pressed to halt continuous triggering.

If the Trigger setting is set to “Start/Stop” then a new count will be initiated only when the ON/OFF button is pressed. Once a count is started, the count will continue until either the dwell time is reached or the ON/OFF button is pressed again to stop the counting cycle.

2.7 CH1 Delay

The signal coming into the Channel 1 connector can be delayed by adjusting the “CH1 Delay” setting. Adding a delay to a channel can be useful when searching for coincident events in signals that may be delayed. For example, if the cable from one photon detector is one meter long while the cable from a second photon detector is two meters long, then there will be a transmission delay of roughly 5 ns that must be subtracted to detect coincident events.

The CH1 delay setting allows delays of 0, 2, 4, 6, 8, 10, 12, and 14 ns to be programmed into the CC1.

2.8 Firmware Ver.

The firmware version is a four digit identifier of the CC1’s firmware release version. This setting cannot be changed by the user.

Serial Interface

3.1 Serial Interface Setup

The CC1 has an emulated COM port for serial communication, and uses SCPI-compliant commands. To use the serial interface, use the included USB-B 2m cable to connect the CC1 to a computer USB port. The USB cable allows the CC1 to both receive power and send data over the USB connection.

To communicate with the CC1, download a serial terminal emulator program (such as RealTerm or PuTTY on Windows) or send serial commands through your own custom program. The serial interface settings should be configured with the following settings:

Setting	Value
Baud Rate	19200 bps
Data bits	8
Parity	None
Stop Bits	1
Flow Control	None

3.2 Commands

The following serial commands can be used to control/interface the CC1 unit with a computer or microcontroller. If a command is not recognized, the CC1 will reply with “Unknown Command.”

Command	Description
COUN:C1? (:C2?)(:CO?)	Returns the current value on the counter, either Channel 1 (C1), Channel 2 (C2), or Coincidences (CO)
:CLEA	Sends the clear signal to the counters
:DELA N	Set the delay value (in nanoseconds) on Channel 1. N may be 0, 2, 4, 6, 8, 10, 12, or 14ns.
:DELA?	Returns the delay value on Channel 1 in nanoseconds
:DWEL N	Sets the dwell time to N seconds. N must be equal to or greater than 0.1 and cannot be more than 5 characters long (including the decimal point).
DWEL?	Returns the dwell time in milliseconds
FIRM?	Returns the firmware release identifier
:GATE N	Sets whether or not the GATE channel is enabled: 0 = No, 1 = Yes
GATE?	Returns whether or not the GATE channel is enabled: 0 = No, 1 = Yes
:SUBT N	Sets whether or not to subtract accidentals from coincidence rate: 0 = No, 1 = Yes
SUBT?	Returns value indicating if accidentals are being subtracted from coincidences: 0 = No, 1 = Yes
:TRIG N	Sets the trigger value: 0 = Continuous, 1 = Start/Stop

Command	Description
TRIG?	Returns the value of the trigger setting: 0 = Continuous, 1 = Start/Stop
:WIND N	Sets the coincidence window length to N nanoseconds. N must be between 1 and 8.
WIND?	Returns the coincidence window length in nanoseconds

3.3 Sample Serial Program

The following sample program (written in Visual Basic 6.0) provides a very simple example of how serial commands can be used through third-party programs to control the CC1:

```
MSComm3.PortOpen = True                          ‘Open COM Port

'-- Prepare Coincidence Counter for Measurements --
MSComm3.Output = ":GATE 0" & Chr$(13)            ‘Disable GATE Channel
Sleep 300                                        'Wait 0.3 seconds for command to transfer
MSComm3.Output = ":SUBT 0" & Chr$(13)            'Turn off Subtract Accidentals option
Sleep 300                                        'Wait 0.3 seconds for command to transfer
MSComm3.Output = ":TRIG 1" & Chr$(13)            'Set Trigger Option to Start/Stop
Sleep 300                                        'Wait 0.3 seconds for command to transfer
MSComm3.Output = ":DWEL 20000" & Chr$(13)        'Set Dwell Time to 20 seconds
Sleep 300                                        'Wait 0.3 seconds for command to transfer
MSComm3.Output = ":DELA 2" & Chr$(13)            'Set Delay Time on Channel 1 to 2ns
Sleep 300                                        'Wait 0.3 seconds for command to transfer
MSComm3.Output = ":WIND 3" & Chr$(13)            ‘Set Coincidence Window to 3 ns
Sleep 300                                        'Wait 0.3 seconds for command to transfer

'-- Take Measurement --
MSComm3.Output = ":COUN:ON" & Chr$(13)

dt = Now                                         ' Wait for Dwell Time (plus 1 second)
Do While DateDiff("s", dt, Now) < 21
    DoEvents()
    Sleep 50                                     'Put app to sleep in small increments
Loop

'--Transfer Measurements --
MSComm3.Output = "COUN:C1?" & Chr$(13)           ‘Query Channel 1 Counts
Sleep 300                                        'Wait 0.3 seconds for command to transfer
Textbox1.text = MSComm3.Input                    ‘Display counts in a textbox
MSComm3.Output = "COUN:C2?" & Chr$(13)           ‘Query Channel 2 Counts
Sleep 300                                        'Wait 0.3 seconds for command to transfer
Textbox2.text = MSComm3.Input                    ‘Display counts in a textbox
MSComm3.Output = "COUN:CO?" & Chr$(13)           ‘Query Coincidence Counts
Sleep 300                                        'Wait 0.3 seconds for command to transfer
Textbox3.text = MSComm3.Input                    ‘Display counts in a textbox

MSComm3.PortOpen = False                         ‘Close COM Port
```

Customizing the CC1

4.1 Overview

The CC1 is an open-source platform that was developed to give as much flexibility to users as possible. This means that the internal firmware that operates the device can be changed by the user, if desired. Inside the CC1, two programmable electronic components are responsible for most of the functionality of the CC1. They include:

- A Rabbit RCM3400 Microprocessor
- An Altera Cyclone IV FPGA

The location of these components inside the CC1 is shown below:

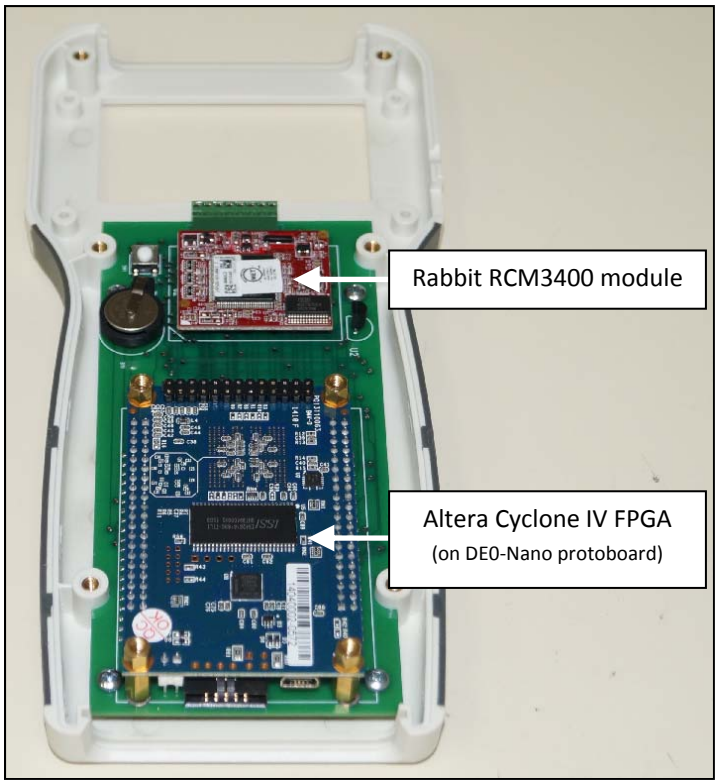


Figure 9. Internal view of CC1 electronics.

The RCM3400 microprocessor is used to implement the user interface (i.e. LCD display, menus, serial communicatons, etc.) as well as some very simple calculations (i.e. accidentals). The Altera FPGA is used for making high speed measurements (i.e. the coincidence counting).

These two devices can be independently programmed to realize new functionality for a wide range of applications. For example, the CC1 device could be reprogrammed to timestamp every photon detection event and create a histogram of the arrival times. Or two CC1s could be re-programmed to act as a four photon coincidence counter (using the GATE channel as a

communication signal between the two CC1s). Or the CC1 could be used with other external devices and serial data to implement a part - or all - of a QKD communication protocol.

Because the electrical components, connections, and user interface are already in place (and described in Appendix A), the CC1 can serve as a flexible platform for interrogating signals from single photon detectors and performing associated operations.

4.1.1 The Rabbit RCM3400 Microprocessor

The Rabbit RCM3400 is a microprocessor module consisting of a simple processor integrated with other general purpose peripheral logic. The device is programmed using a language proprietary to Rabbit Semiconductor called Dynamic C. Dynamic C is very similar to ANSI C, with extra functions for interacting with the digital I/O, analog inputs, and PWM outputs on the RCM3400 module.

To program in Dynamic C, an integrated development environment (Dynamic C 9.62) and programmer utility (DC 9.62 RFU) must be installed on your computer. A single installation program (DynamicC_9.62_WebFull.exe) - available on the flash drive that came with your unit as well as from Qubitekk's website (www.qubitekk.com) – will install both programs. A programming cable for uploading new firmware to your CC1 is also required and has been included with your unit (Rabbit Programming Cable - 2mm)

A great resource for learning Dynamic C and how to program the RCM3400 is the book entitled, "Embedded Systems Design using the Rabbit 3000 Microprocessor" by Kamal Hyder. Although this book was published in 2004, it still contains very relevant and useful examples of how to program and understand the operation of the RCM3400. In addition, manuals for the RCM3400 and Dynamic are available on the flash drive that came with your unit and Qubitekk's website.

4.1.2 The Altera Cyclone IV FPGA

The Altera Cyclone IV chip is a field programmable gate array (FPGA) that can operate at frequencies in excess of 1GHz. The chip is particularly well suited to detecting and timestamping LVTTL pulses (as would be emitted by a single photon counting module). The chip can be programmed using Altera's Quartus II Web Edition (version 13+) – which is free and available at both Altera's website (www.altera.com) and Qubitekk's website.

In the CC1, the Altera FPGA is actually part of an off-the-shelf prototyping board (the Terasic DE0-Nano). This prototyping board has additional functionality that may be used in future versions of the CC1 product. A manual for the Terasic DE0-Nano is available on the flash drive that came with your unit as well as Qubitekk's website.

4.1.3 Default Firmware

The CC1 comes pre-programmed for use as a general purpose coincidence counter. However, the Dynamic C source code and the FPGA Quartus project files (including verilog modules,

block diagrams, pin assignments, board support files and board schematic files) are all included on the flash drive that came with your unit.

If your device should ever need to be returned to its original operating firmware, the source code and the programming files to do this are available on the flash drive. In addition, newer versions of the CC1's firmware are posted online at Qubitekk's website and are freely available to all.

The default firmware can be a useful place to start when exploring how to modify your CC1 unit. Both the source code for the RCM3400 microprocessor and the Cyclone IV FPGA are available and well commented. Be sure to check the Qubitekk website frequently for the latest versions of the default firmware.

4.2 Uploading Custom Firmware

4.2.1 Accessing the Programming Ports

To upload new firmware to your CC1, it is necessary to first remove the cover at the bottom of the CC1. To do so, first remove the six Phillips screws on the back of the CC1 that hold the two halves of the CC1 instrument case together. With these screws removed, the two halves of the enclosure can be gently pulled apart (pull each half straight away from the other for easiest disassembly).

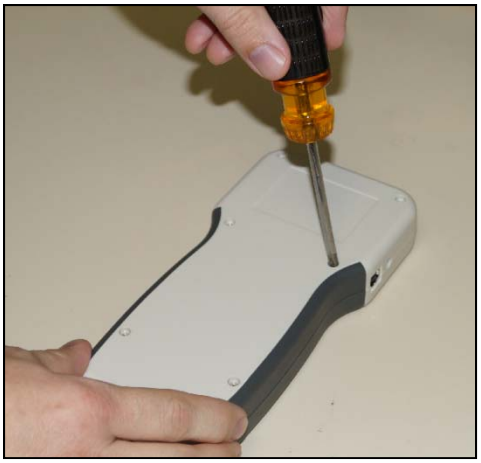


Figure 10. Removal of screws to disassemble CC1.

With the two halves separated, it is now possible to access and remove the bottom cover. The disassembled CC1 should look as shown in Figure 11. Note how the two halves of the CC1 are connected through a 2x7 pin header. When reassembling the CC1, be careful to realign these pins properly.

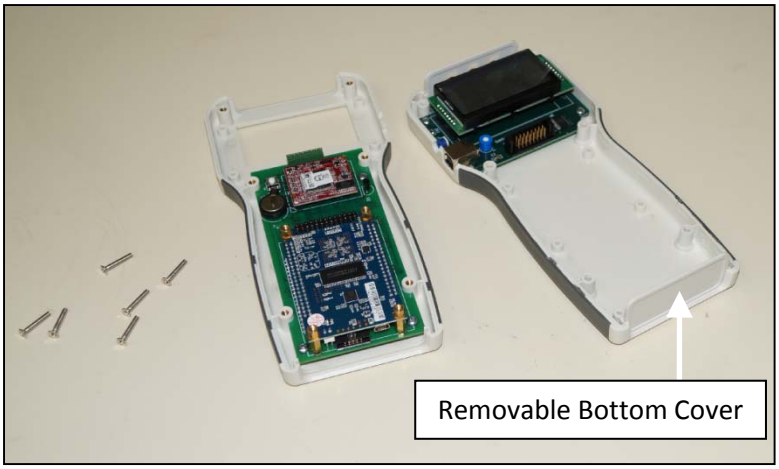


Figure 11. Disassembled CC1

With the bottom cover removed, the programming ports for the RCM3400 and the Cyclone IV FPGA should be visible and accessible. Using the reverse order, the unit should be reassembled without the bottom cover. The bottom of the CC1 will look as shown in Figure 12.

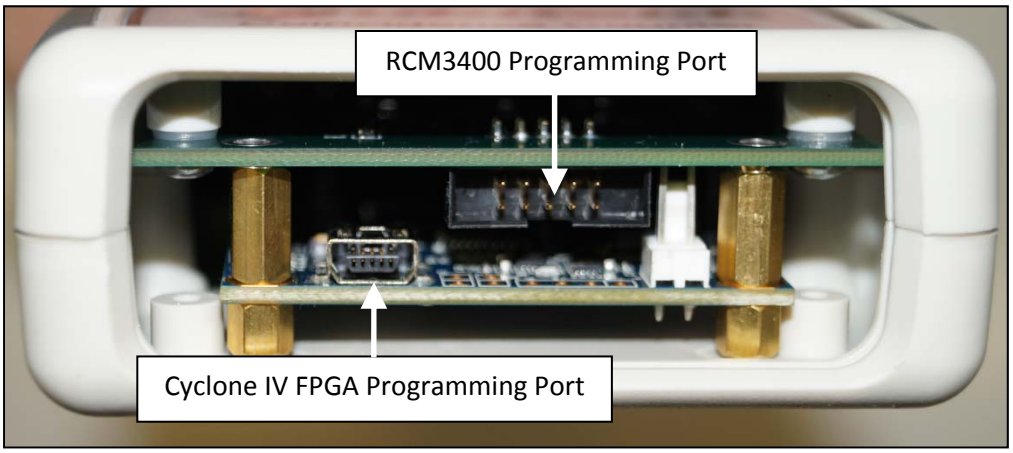


Figure 12. Programming ports exposed.

4.2.2 Programming the Rabbit RCM3400

To program the Rabbit RCM3400, a *.bin programming file must first be created in the Dynamic C 9.62 development environment. To create a *.bin file from Dynamic C source code, select “Compile -> Compile to .bin file -> Compile to Flash” from the Dynamic C 9.62 program. The location of this menu option is shown in Figure 13.

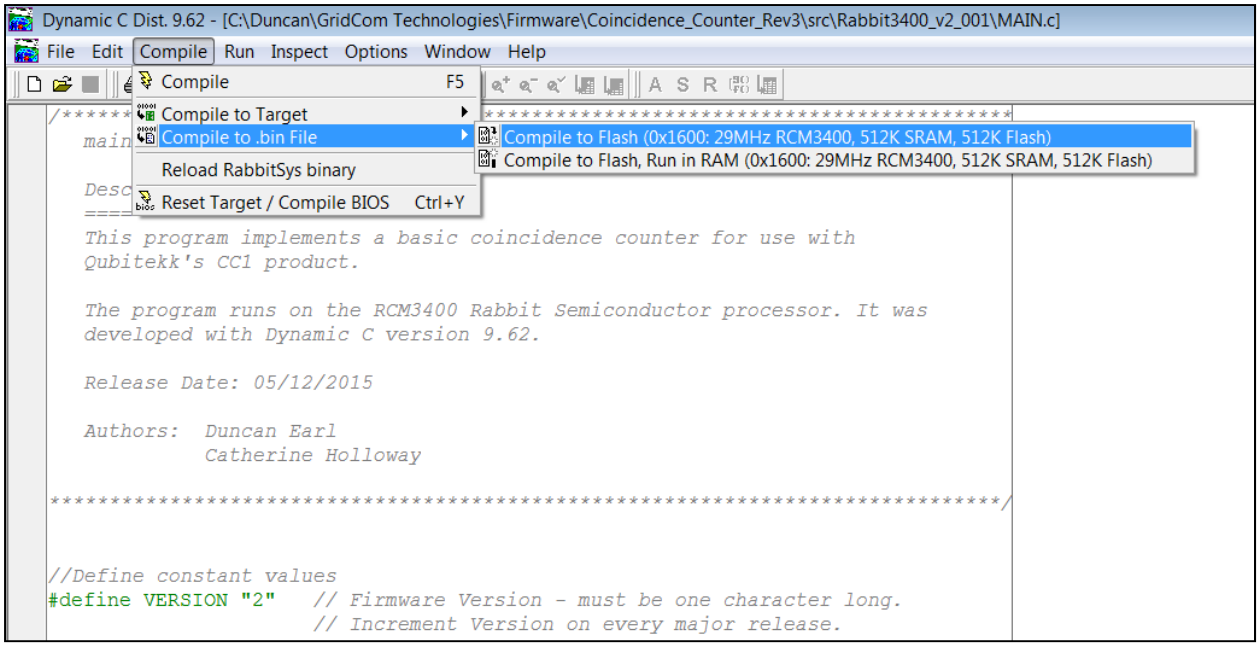


Figure 13. Creating a programming file for the RCM3400.

With the programming file (.bin) created, plug the USB end of the included Rabbit Programming Cable (2mm) into your computer's USB port. The other end of the programming cable contains two 10-pin headers. The header labeled "PROG" should be connected to the Rabbit Programming Port on the base of your CC1 unit as shown in Figure 14 (note orientation of red wire):

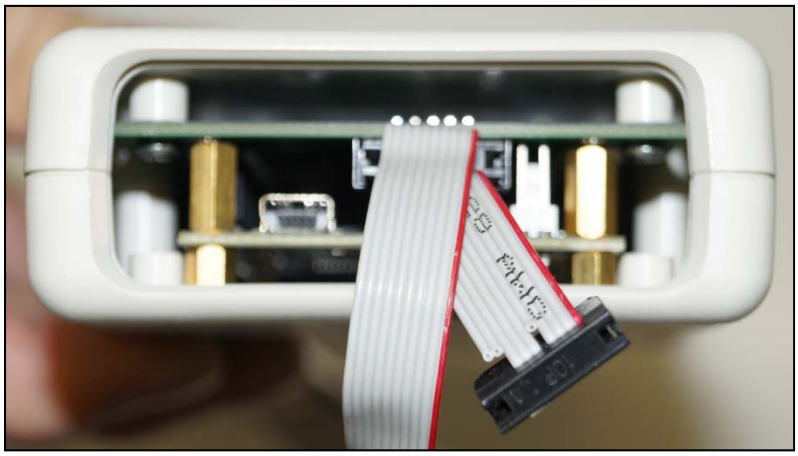


Figure 14. Creating a programming file for the RCM3400.

With the Rabbit Programming Cable in place, connect the CC1's USB cable to your computer - or to the provided USB power supply - to power up the CC1. Open the DC RFU program that was installed on your computer. The DC RFU is a programming utility that will use the programming cable to upload any new firmware files (.bin) to the CC1's RCM3400.

Select “File -> Load Flash Image…” from the menu, as shown in Figure 15. Once the proper file is selected, the uploading of the firmware should begin immediately.

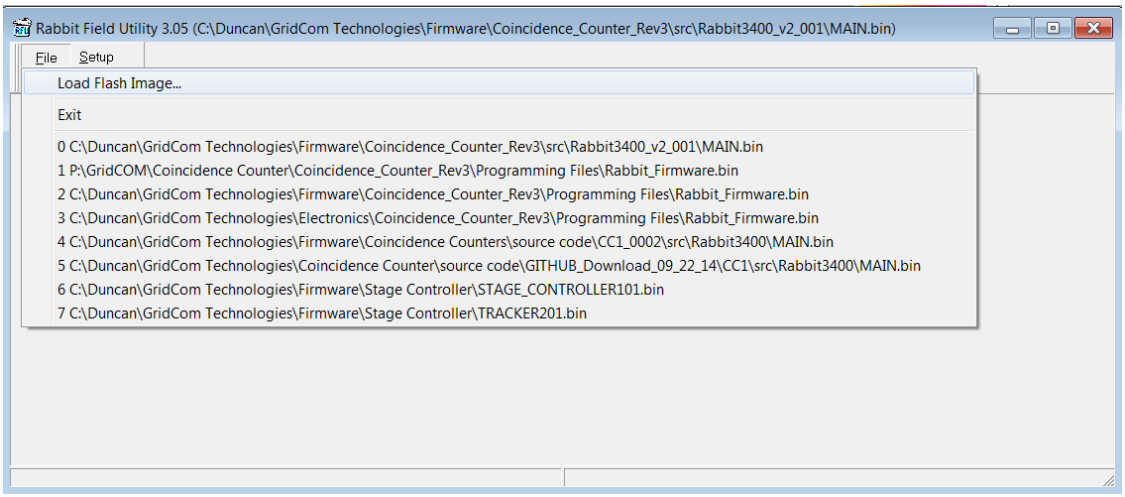


Figure 15. Creating a programming file for the RCM3400.

On some systems, the port location of the Rabbit Programming Cable may not be recognized automatically. If this happens, you will get an error that says “Could not open serial port.” If this occurs, you will need to select the “Setup -> Communications…” item in the DC RFU program menu. This will open up a communications screen where you can select the proper COM port that identifies your Rabbit Programming Cable. An example of this screen (with the proper COM port settings) is shown in Figure 16:

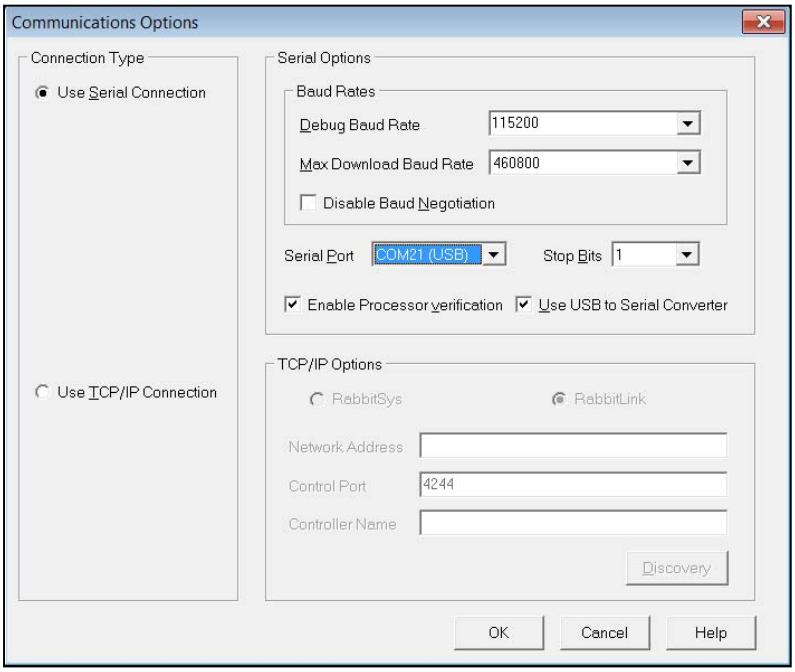


Figure 16. Communication settings for the Rabbit Programming Cable.

4.2.3 Programming the Cyclone IV FPGA

To program the Altera Cyclone IV FPGA - and have that program persist in memory after power cycling the device - a *.jic programming file must be created in the Quartus II development environment. However, to create a *.jic file, a compiled Quartus II design must first be produced. How to develop and compile a Quartus II design file is covered extensively on Altera's website and in various resources on the web. Any working design ultimately gets compiled and results in an SRAM Object File (.sof) in the design's working directory.

To create the *.jic file, the design's *.sof file can be converted by selecting the menu item "File -> Convert Programming Files…" in Quartus II. This will open the "Convert Programming File" window which allows various parameters to be provided by the user to convert a .sof file to a .jic file. In this window, the Programming File Type should be set to .jic and the configuration device should be set to EPCS64. Select the "Flash Loader" in the "File/Data area" and click the "Add Device" button to the right. The EP4CE22 device should be selected from the pop-up window. Select the "SOF Data" in the "File/Data area" and click the "Add File" button to the right. The recently compiled *.sof file for the design should be selected from the file dialog box.

When finished, the "Convert Programming File" window should look as shown in Figure 17. Click the "Generate" button to convert the .sof file to a .jic file.

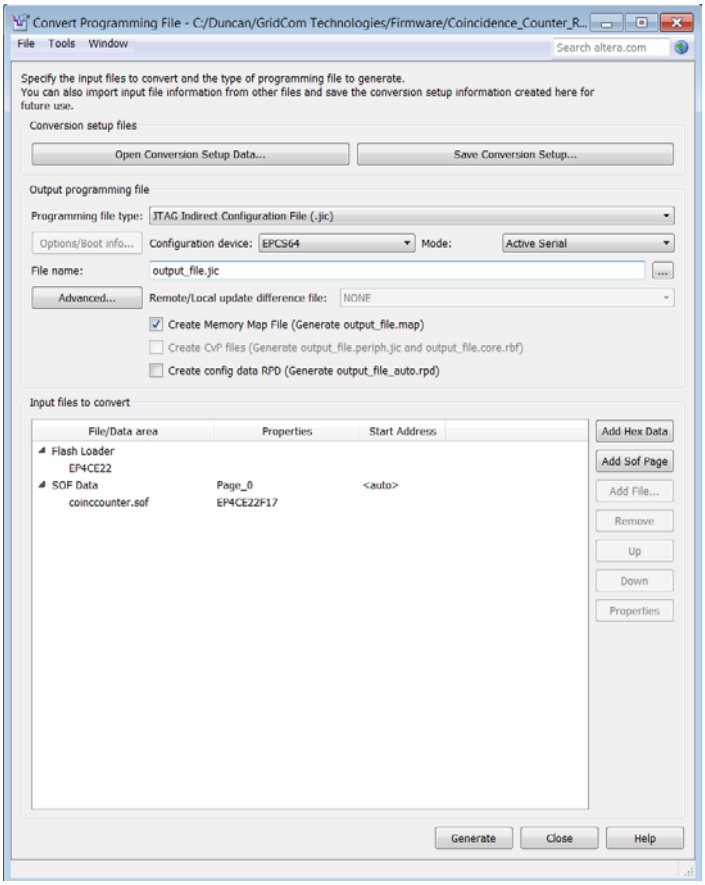


Figure 17. Convert Programming File window parameters.

To program the FPGA, connect the retractable USB cable (USB-mini) from your computer to the CC1 (this USB cable will provide all necessary power to the CC1 during programming). Select the “Tools -> Programmer” menu option in Quartus II to open the Altera Programmer window. Click the “Add File” button to add the *.jic file that was just created and select the USB-Blaster by clicking the “Hardware Setup” button. The Programmer setup window should look as shown in Figure 18.

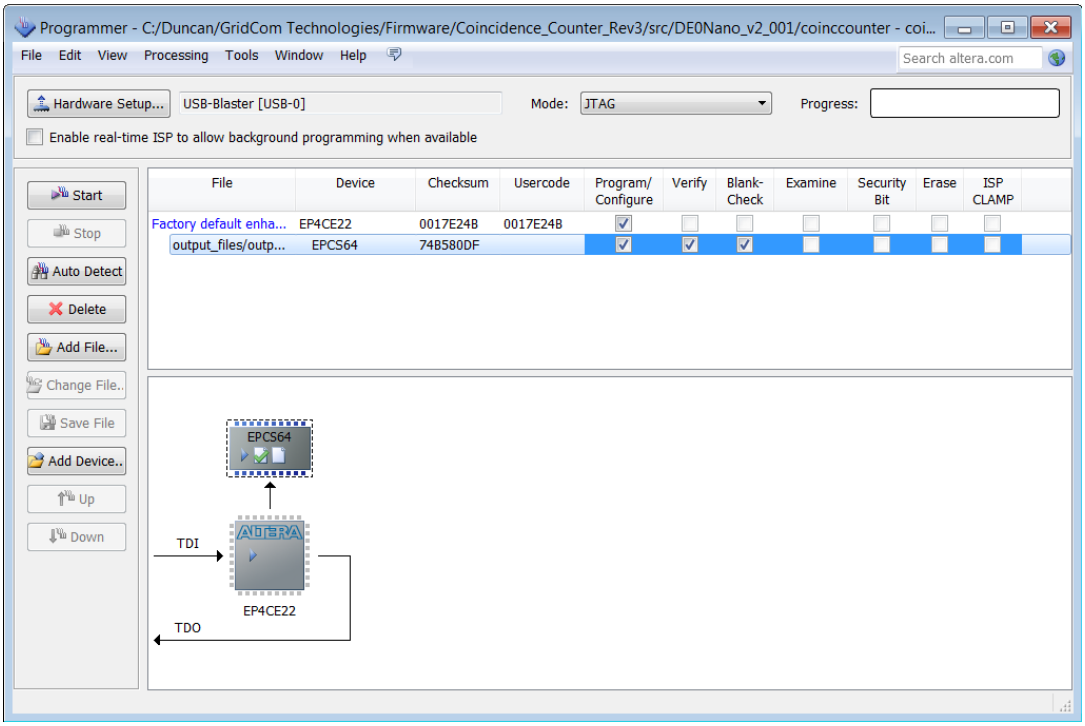


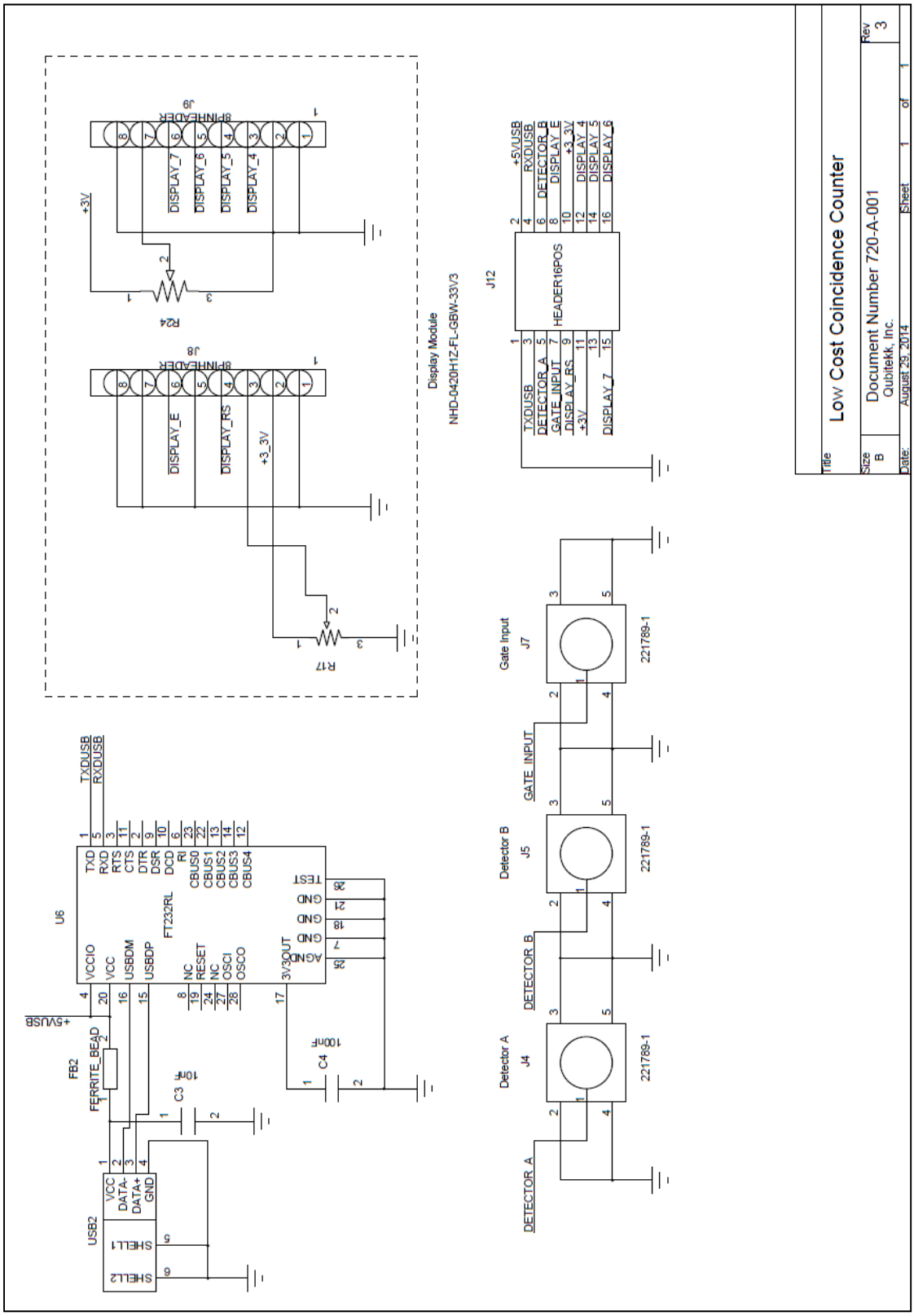
Figure 18. Programmer settings.

Click “Start” to program the FPGA.

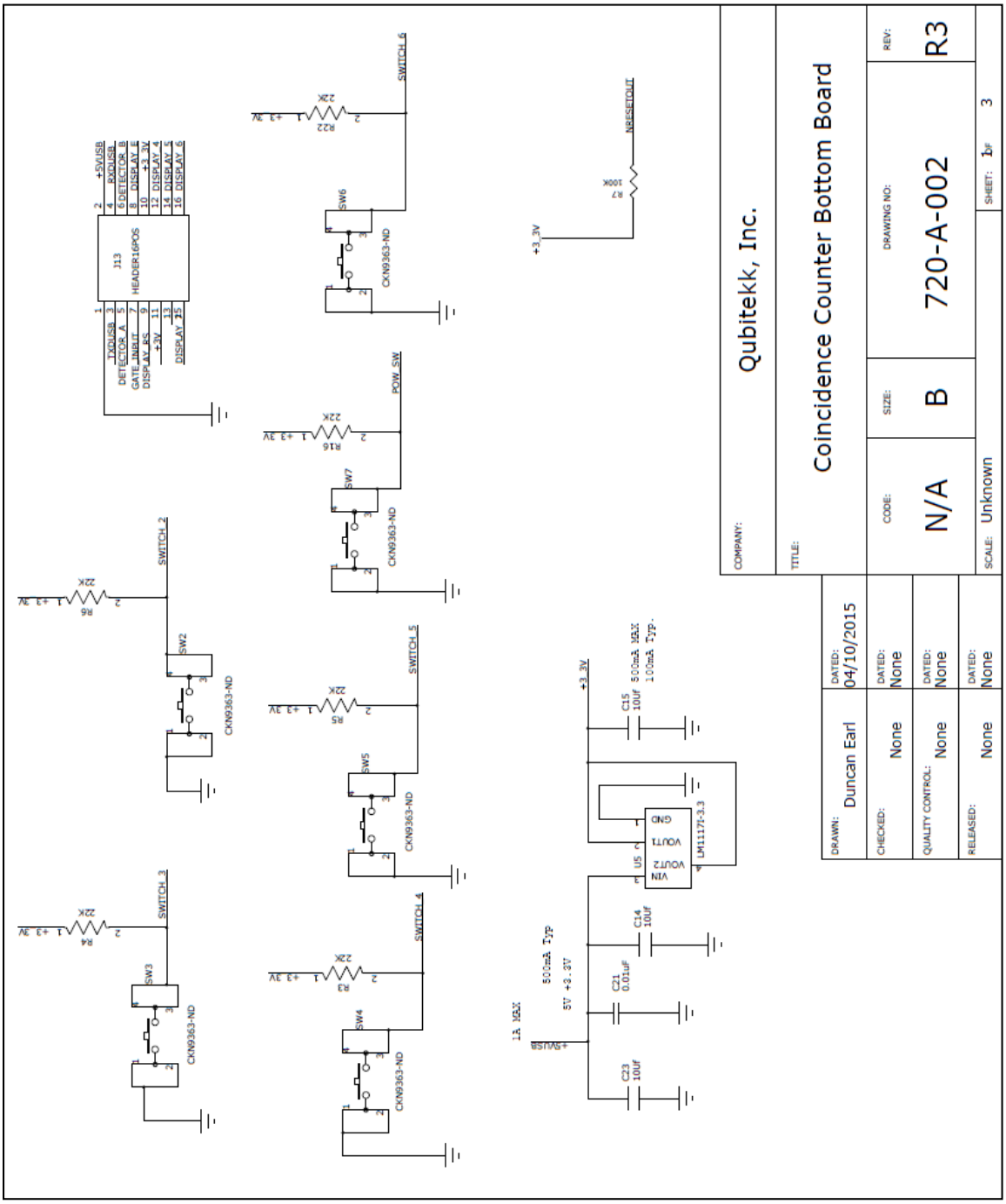
APPENDIX A

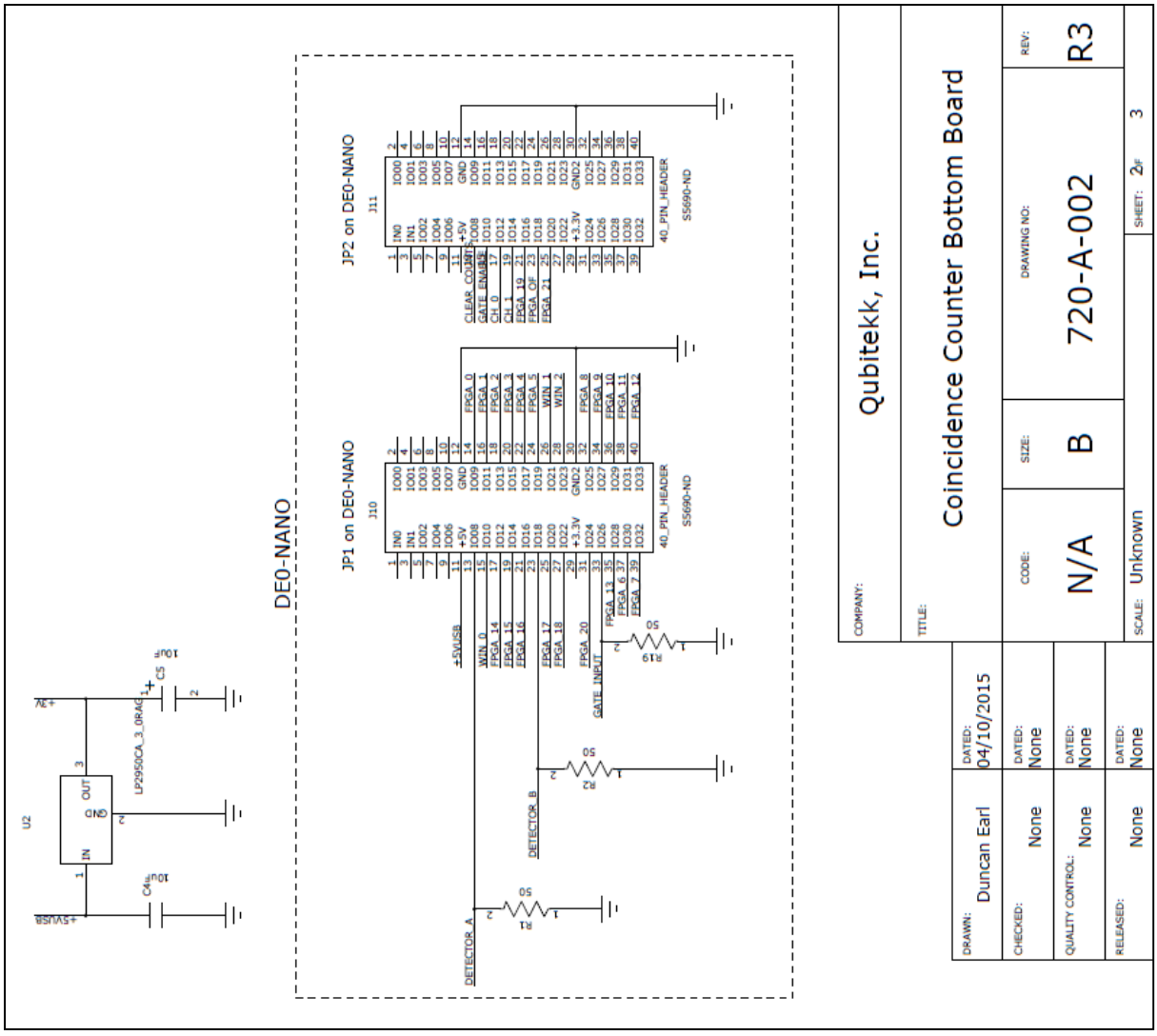
Electrical Schematics and Connections

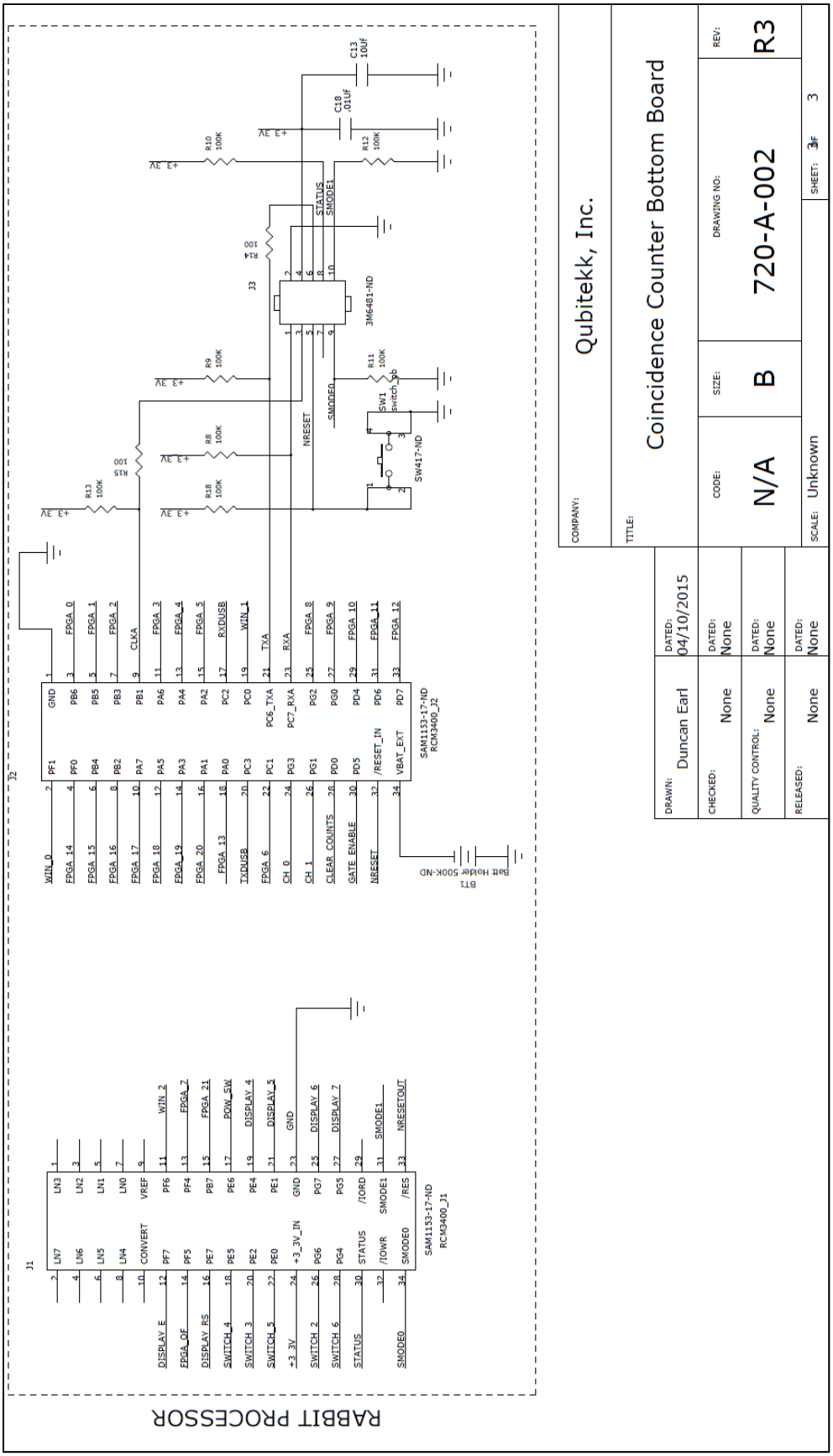
A.1 Top Board (LCD Display)



A.2 Bottom Board (FPGA and Microprocessor)







A.3 Pinout Table

The pinout table for the RCM3400 is shown below:

Header	Pin Number	Pin Name	Description
J1	1	LN3	Not Connected
	2	LN7	Not Connected
	3	LN2	Not Connected
	4	LN6	Not Connected
	5	LN1	Not Connected
	6	LN5	Not Connected
	7	LN0	Not Connected
	8	LN8	Not Connected
	9	VREF	Not Connected
	10	CONVERT	Not Connected
	11	PF6	Coincidence Window or Delay (Bit 2)
	12	PF7	Bus to LCD Display (Line E)
	13	PF4	Bus to FPGA Counter (Bit 7)
	14	PF5	Bus to FPGA Counter (Overflow Flag)
	15	PB7	Bus to FPGA Counter (Bit 21)
	16	PE7	Bus to LCD Display (Line RS)
	17	PE6	Start/Stop Button on User Interface
	18	PE5	Down Button on User Interface
	19	PE4	Bus to LCD Display (Line 4)
	20	PE2	Clear Button on User Interface
	21	PE1	Bus to LCD Display (Line 5)
	22	PE0	Menu Button on User Interface
	23	GND	Ground
	24	+3_3V_IN	+3.3V Input (Power)
	25	PG7	Bus to LCD Display (Line 6)
	26	PG6	Enable/Select Button on User Interface
	27	PG5	Bus to LCD Display (Line 7)
	28	PG4	Up Button on User Interface
	29	/IORD	Not Connected
	30	STATUS	Reserved for firmware upload
	31	SMODE1	Reserved for firmware upload
	32	/IOWR	Not Connected
	33	/RES	Reserved for firmware upload
	34	SMODE0	Reserved for firmware upload
J2	1	GND	Ground
	2	PF1	Coincidence Window or Delay (Bit 0)
	3	PB6	Bus to FPGA Counter (Bit 0)
	4	PF0	Bus to FPGA Counter (Bit 14)
	5	PB5	Bus to FPGA Counter (Bit 1)
	6	PB4	Bus to FPGA Counter (Bit 15)
	7	PB3	Bus to FPGA Counter (Bit 2)
	8	PB2	Bus to FPGA Counter (Bit 16)
	9	PB1	Reserved for firmware upload
	10	PA7	Bus to FPGA Counter (Bit 17)

J2	11	PA6	Bus to FPGA Counter (Bit 3)
	12	PA5	Bus to FPGA Counter (Bit 18)
	13	PA4	Bus to FPGA Counter (Bit 4)
	14	PA3	Bus to FPGA Counter (Bit 19)
	15	PA2	Bus to FPGA Counter (Bit 5)
	16	PA1	Bus to FPGA Counter (Bit 20)
	17	PC2	Serial communication with USB chip (Rx)
	18	PA0	Bus to FPGA Counter (Bit 13)
	19	PC0	Coincidence Window or Delay (Bit 1)
	20	PC3	Serial communication with USB chip (Tx)
	21	PC6_TXA	Reserved for firmware upload
	22	PC1	Bus to FPGA Counter (Bit 6)
	23	PC7_RXA	Reserved for firmware upload
	24	PG3	Channel Select for FPGA Counter (Bit 0)
	25	PG2	Bus to FPGA Counter (Bit 8)
	26	PG1	Channel Select for FPGA Counter (Bit 1)
	27	PG0	Bus to FPGA Counter (Bit 9)
	28	PD0	Clear FPGA Counter Bit
	29	PD4	Bus to FPGA Counter (Bit 10)
	30	PD5	Enable GATE Signal Flag for FPGA
	31	PD6	Bus to FPGA Counter (Bit 11)
	32	/RESET_IN	Reserved for firmware upload
	33	PD7	Bus to FPGA Counter (Bit 12)
	34	VBAT_EXT	Reserved for Battery Backup

The pinout table for the Cyclone IV FPGA (and DE0-Nano) is as shown below:

Header	Pin Number	Pin Name	Description
JP1	1	IN0	Not Connected
	2	IO00	Not Connected
	3	IN1	Not Connected
	4	IO01	Not Connected
	5	IO02	Not Connected
	6	IO03	Not Connected
	7	IO04	Not Connected
	8	IO05	Not Connected
	9	IO06	Not Connected
	10	IO07	Not Connected
	11	+5V	+5V Power from USB connector
	12	GND	Ground
	13	IO08	Channel A Signal In
	14	IO09	FPGA Counter Output to RCM3400 (Bit 0)
	15	IO10	Coincidence Window or Delay (Bit 0)
	16	IO11	FPGA Counter Output to RCM3400 (Bit 1)
	17	IO12	FPGA Counter Output to RCM3400 (Bit 14)
	18	IO13	FPGA Counter Output to RCM3400 (Bit 2)
	19	IO14	FPGA Counter Output to RCM3400 (Bit 15)

JP1	20	IO15	FPGA Counter Output to RCM3400 (Bit 3)
	21	IO16	FPGA Counter Output to RCM3400 (Bit 16)
	22	IO17	FPGA Counter Output to RCM3400 (Bit 4)
	23	IO18	Channel B Signal In
	24	IO19	FPGA Counter Output to RCM3400 (Bit 5)
	25	IO20	FPGA Counter Output to RCM3400 (Bit 17)
	26	IO21	Coincidence Window or Delay (Bit 1)
	27	IO22	FPGA Counter Output to RCM3400 (Bit 18)
	28	IO23	Coincidence Window or Delay (Bit 2)
	29	+3.3V	Not Connected
	30	GND2	Ground
	31	IO24	FPGA Counter Output to RCM3400 (Bit 20)
	32	IO25	FPGA Counter Output to RCM3400 (Bit 8)
	33	IO26	Channel GATE Signal In
	34	IO27	FPGA Counter Output to RCM3400 (Bit 9)
	35	IO28	FPGA Counter Output to RCM3400 (Bit 13)
	36	IO29	FPGA Counter Output to RCM3400 (Bit 10)
	37	IO30	FPGA Counter Output to RCM3400 (Bit 6)
	38	IO31	FPGA Counter Output to RCM3400 (Bit 11)
	39	IO32	FPGA Counter Output to RCM3400 (Bit 7)
	40	IO33	FPGA Counter Output to RCM3400 (Bit 12)
JP2	1	IN0	Not Connected
	2	IO00	Not Connected
	3	IN1	Not Connected
	4	IO01	Not Connected
	5	IO02	Not Connected
	6	IO03	Not Connected
	7	IO04	Not Connected
	8	IO05	Not Connected
	9	IO06	Not Connected
	10	IO07	Not Connected
	11	+5V	Not Connected
	12	GND	Ground
	13	IO08	Clear FPGA Counter Bit
	14	IO09	Not Connected
	15	IO10	Enable GATE Signal Flag for FPGA
	16	IO11	Not Connected
	17	IO12	Channel Select for FPGA Counter (Bit 0)
	18	IO13	Not Connected
	19	IO14	Channel Select for FPGA Counter (Bit 1)
	20	IO15	Not Connected
	21	IO16	FPGA Counter Output to RCM3400 (Bit 19)
	22	IO17	Not Connected
	23	IO18	FPGA Counter Output to RCM3400 (Overflow Bit)
	24	IO19	Not Connected
	25	IO20	FPGA Counter Output to RCM3400 (Bit 21)
	26	IO21	Not Connected
	27	IO22	Not Connected
	28	IO23	Not Connected

JP2	29	+3.3V	Not Connected
	30	GND2	Ground
	31	IO24	Not Connected
	32	IO25	Not Connected
	33	IO26	Not Connected
	34	IO27	Not Connected
	35	IO28	Not Connected
	36	IO29	Not Connected
	37	IO30	Not Connected
	38	IO31	Not Connected
	39	IO32	Not Connected
	40	IO33	Not Connected