



UnoMax CPLD Development System User Manual

EARTH PEOPLE TECHNOLOGY, Inc

UNOMAX CPLD DEVELOPMENT SYSTEM User Manual

The UnoMax CPLD development system provides an innovative method of developing and debugging programmable logic code. It also provides a high speed data transfer mechanism between user code and a host PC. The UnoMax CPLD development system provides a convenient, user-friendly work flow by connecting seamlessly with Intel/Altera's Quartus Prime Lite software. The user will develop the code in the Quartus environment on a Windows Personal Computer. The programmable logic code is then synthesized and loaded into the CPLD using only the Quartus Programmer tool and a standard USB-C cable. The Active Host SDK provides a highly configurable communications interface between the UnoMax and host PC. The board connects transparently with the Active Transfer Library in the CPLD code. This Active Host/Active Transfer combination eliminates the complexity of designing a USB communication system. The UnoMax CPLD development system is a unique combination of hardware and software.

Circuit designs, software and documentation are copyright © 2023, Earth People Technology, Inc

Microsoft and Windows are both registered trademarks of Microsoft Corporation. Altera is a trademark of the Altera Corporation. All other trademarks referenced herein are the property of their respective owners and no trademark rights to the same are claimed.

<http://www.earthpeopletechnology.com/>



UnoMax CPLD Development System User Manual

Table of Contents

1 Introduction and General Description.....4

1.1 Test Driving the Active Host Test Application.....5

1.2 EPT-5M57-AP-U3 14

1.2.1 High Speed USB Communications 17

1.2.2 Serial USB Communications 18

1.2.3 Host PC Connection..... 19

1.2.4 PMOD Connectors 19

1.2.5 Inputs and Outputs 21

1.2.6 User LEDs..... 23

1.2.7 User Push Buttons 24

1.2.8 Slide Switches 25

1.2.9 Analog Inputs 27

1.2.10 JTAG..... 27

1.3 Active Host EndTerms 27

1.4 Active Transfer EndTerms 28

2 EPT Drivers..... 29

2.1 USB Driver..... 29

2.2 JTAG DLL Insert to Quartus Prime Lite..... 32

2.2.1 Installing Quartus 32

2.2.2 Downloading Quartus 34

2.2.3 Quartus Installer 39

2.2.4 Adding the EPT_Blaster to Quartus Prime 47

2.3 Active Host Application DLL 48

3 Active Transfer Library..... 53

3.1 EPT Active Transfer System Overview 53

3.2 Active Transfer Library 54

3.2.1 Active Trigger EndTerm 58

3.2.2 Active Transfer EndTerm 62

3.2.3 Active Block EndTerm 64

3.3 Timing Diagram for Active Transfer EndTerms..... 67

3.3.1 Active Trigger EndTerm Timing 67

3.3.2 Active Transfer EndTerm Timing..... 67

3.3.3 Active Block EndTerm Timing..... 68

4 Compiling, Synthesizing, and Programming CPLD..... 69

4.1 Setting up the Project and Compiling..... 69

4.1.1 Selecting Pins and Synthesizing..... 79

4.2 Programming the CPLD..... 88

5 Active Host Application..... 95

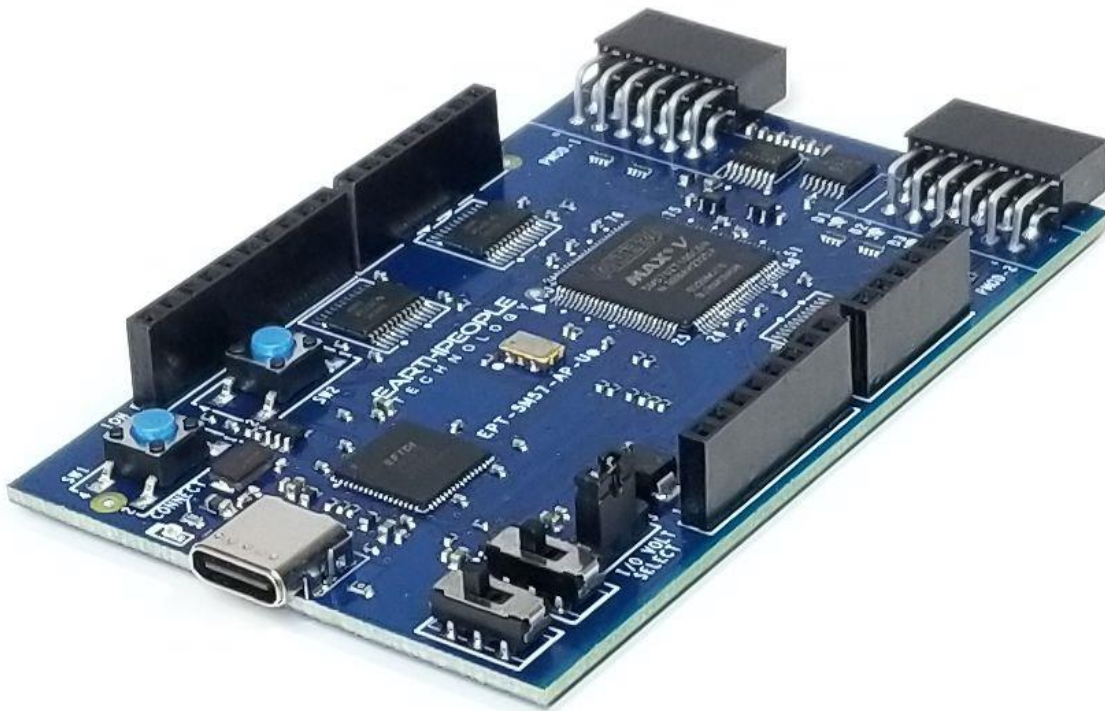


UnoMax CPLD Development System User Manual

5.1	Trigger EndTerm	96
5.2	Transfer(Byte) EndTerm	96
5.3	Block EndTerm	97
5.4	Active Host DLL	97
5.4.1	Active Host Open Device.....	98
5.4.2	Active Host Triggers	101
5.4.3	Active Host Byte Transfers	103
5.4.4	Active Host Block Transfers	105
5.5	Creating a Project	108
5.5.1	Setting up the C# Express Environment x64 bit	110
5.6	Assembling Files into the Project.....	117
5.6.1	Changing Project Name	117
5.6.2	Add Files to Project.....	119
5.6.3	Adding Controls to the Project.....	121
5.6.4	Building the Project	126
5.6.5	Testing the Project.....	127
6	Using the Analog to Digital Converter	131
6.1	Register Descriptions	132
6.2	Conversion Register	133
6.3	Setup Register.....	134
6.4	Averaging Register.....	135
6.5	Reset Register.....	136
7	The UnoProLyzer Application.....	137
7.1	Accessing the UnoProLyzer Application	138
8	APPENDIX I.....	149

1 Introduction and General Description

The Earth People Technology UnoMax CPLD development system hardware consists of a High Speed USB chip, a CPLD and a four channel ADC. The USB interface provides both JTAG programming for the CPLD and a High Speed communications path. The CPLD is an Intel/Altera MAXV chip providing 570 Logic Cells. The ADC is a four channel 300KSample/sec data converter. The software consists of the Active Host SDK for the PC. The firmware includes the Active Transfer Library which is used in the CPLD to provide advanced functions for control and data transfer to/from a PC connection.



The UnoMax Development System allows users to write HDL code (either Verilog or VHDL) that will implement any digital logic circuit. The user's HDL code is compiled and synthesized and packaged into a programming file. The programming file is programmed into the CPLD using the JTAG channel of the USB to Serial chip, the FT2232H. The Active Host SDK contains a dll which maintains device connection, polling, writes and includes a unique receive mechanism that automatically transfers

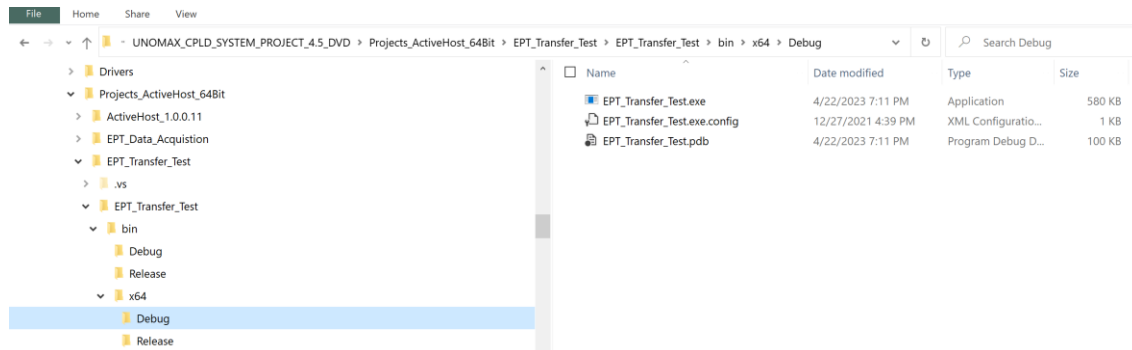


UnoMax CPLD Development System User Manual

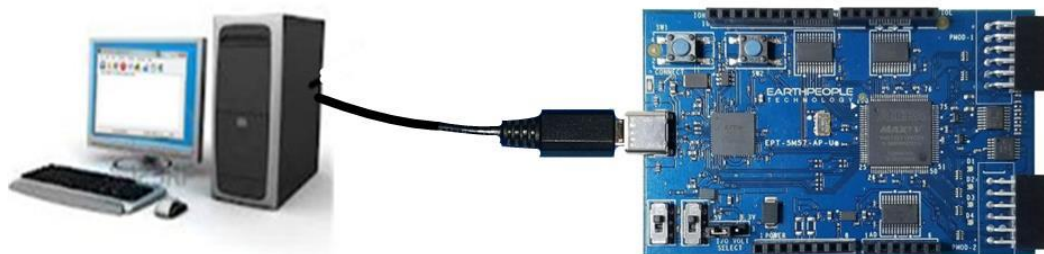
data from UnoMax when data is ready. It also alerts the user code when the dll has stored the transfer and the data is available to the software GUI (graphical user interface). Users do not need to interface with the USB Host Driver or any Windows drivers. They need only to include the Active Host dll in their projects. The Active Transfer Libraries must be included in the CPLD project to take advantage of the configurability of the Active Host SDK. All of the drivers, libraries, and project source code are available at www.earthpeopletechnology.com.

1.1 Test Driving the Active Host Test Application

The UnoMax board comes pre-loaded with the EPT_Transfer_Test HDL project in the CPLD. This project allows the user to test out the functions of the Active Host API and the board hardware.

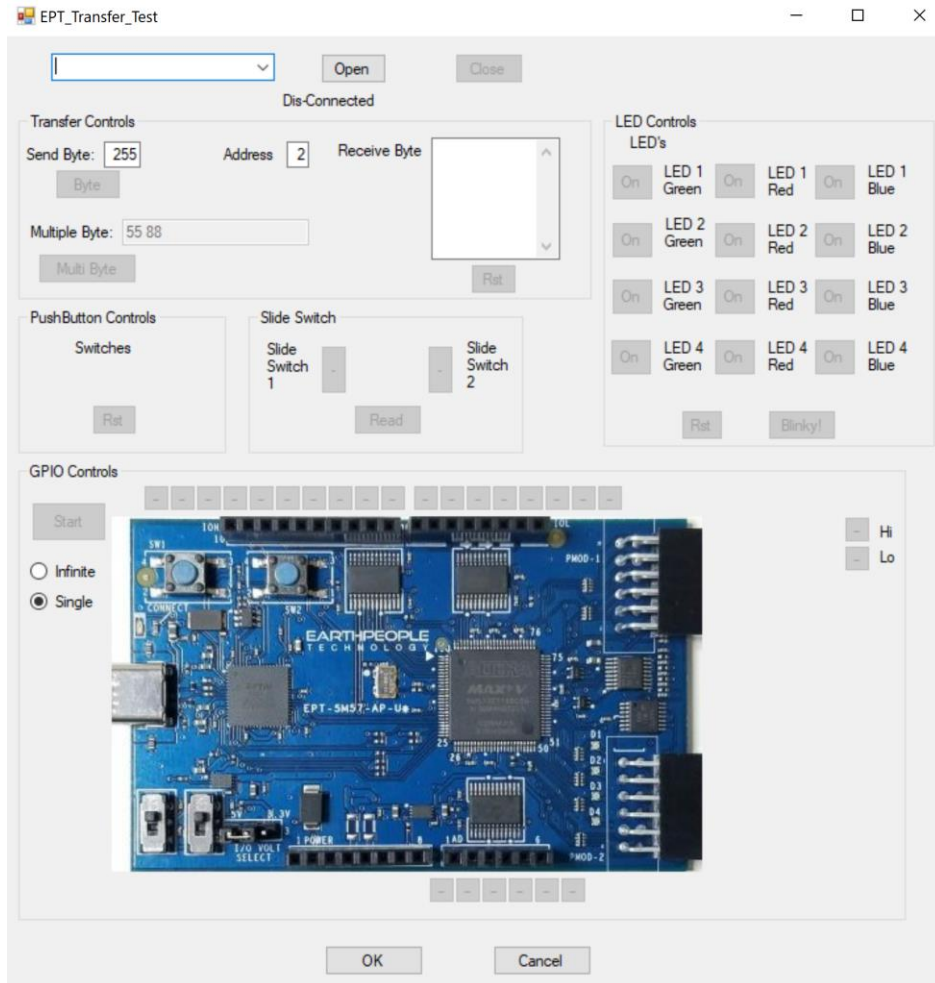


To test drive the application, connect the UnoMax to the Windows PC using Type A to Micro B USB cable. Load the driver for the board. See the section EPT Drivers for instructions on loading the EPT-5M57-AP-U3 driver.



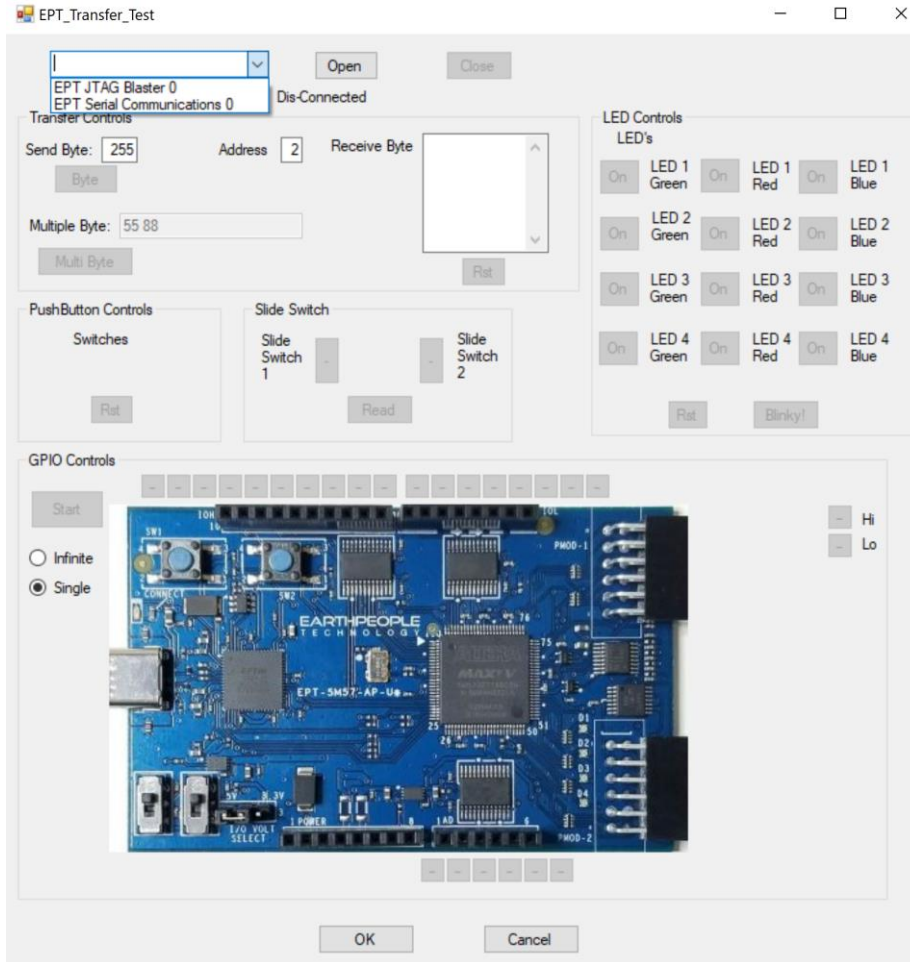
UnoMax CPLD Development System User Manual

Next, open a Windows Explorer browser. Browse to the `Projects_ActiveHost_64Bit\EPT_Transfer_Test\EPT_Transfer_Test\bin\X64\Debug\` folder on the `UNOMAX_CPLD_SYSTEM_PROJECT_4.x_DVD`. Double click on the `EPT_Transfer_Text.exe`. The application should load with a Windows form.



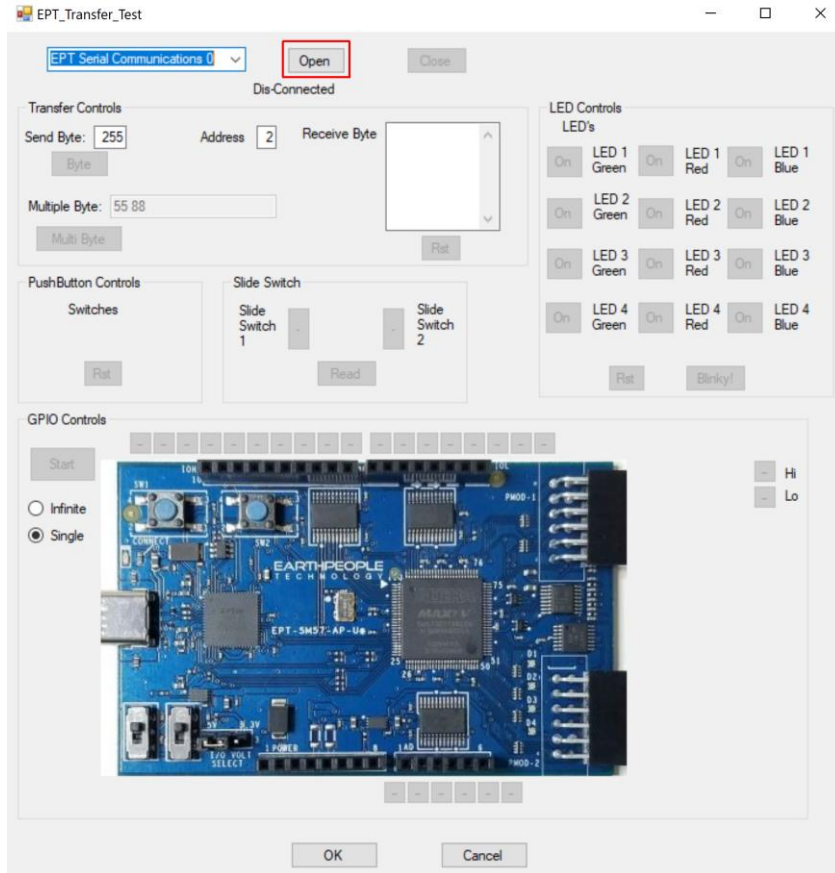
With the application loaded, select the USB-CPLD board from the dropdown combo box and click on the “Open” button.

UnoMax CPLD Development System User Manual



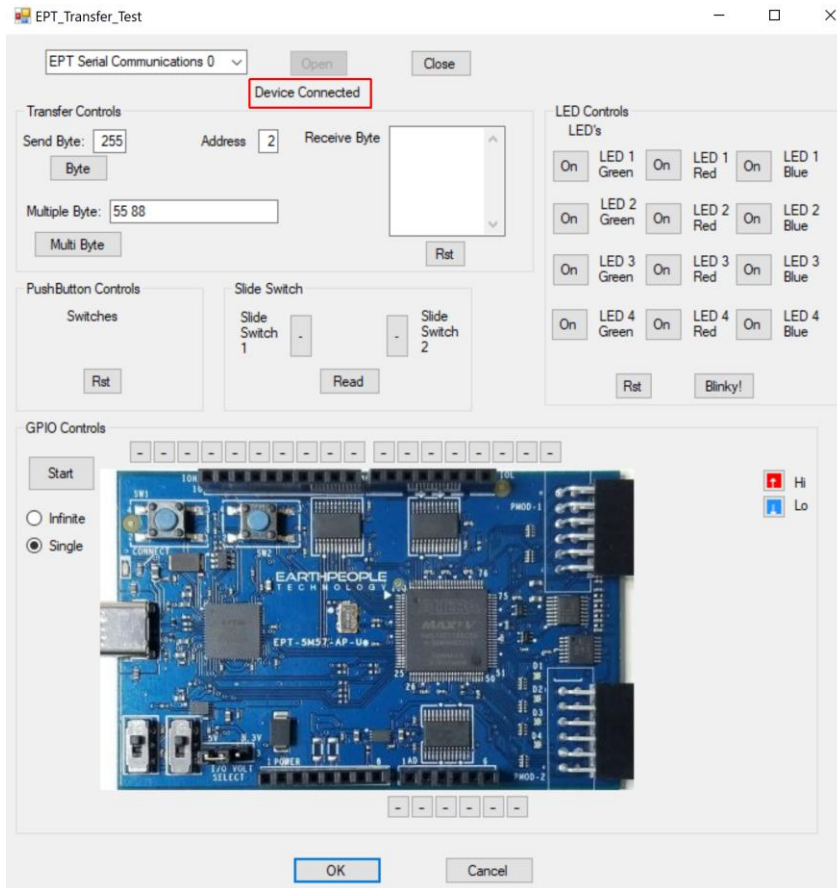
Next, click on the Open button.

UnoMax CPLD Development System User Manual



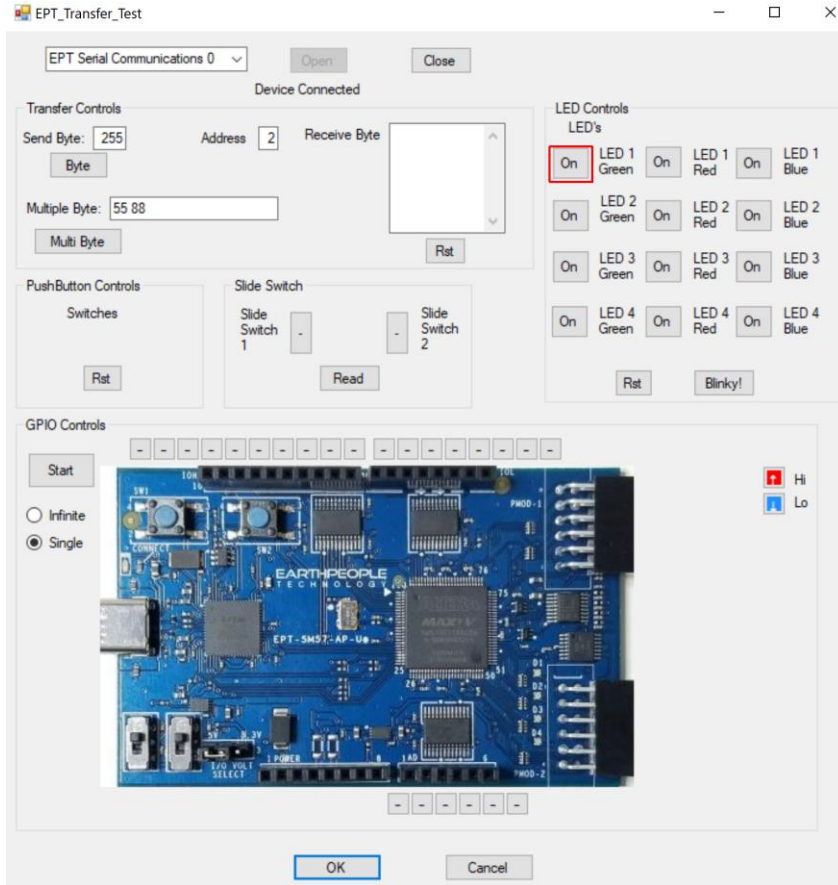
The “Device Connected” label should be seen to indicate the Windows App is now connected to the UnoMax.

UnoMax CPLD Development System User Manual



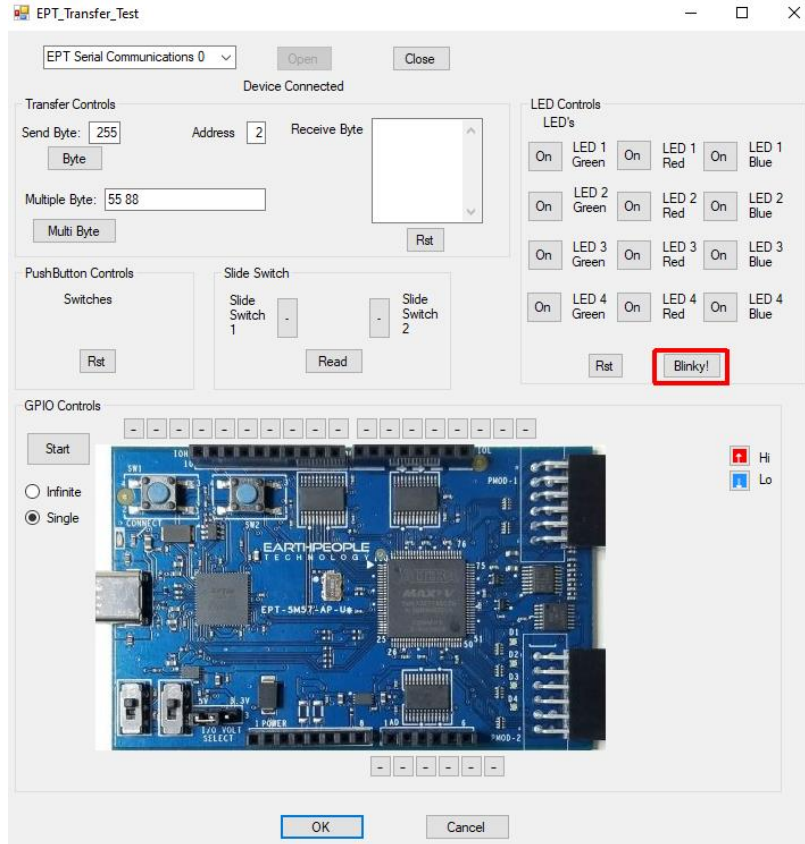
Click on one of the LED buttons in the middle of the window. The corresponding LED on the UnoMax board should light up.

UnoMax CPLD Development System User Manual



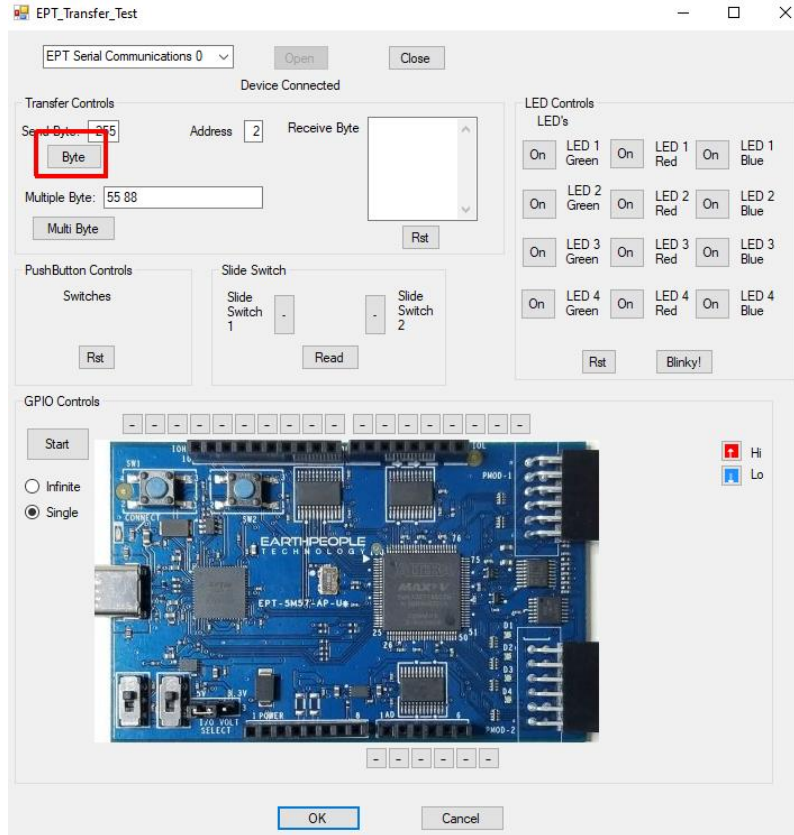
Click on the Blinky button for a light show.

UnoMax CPLD Development System User Manual

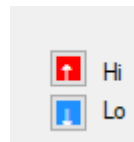


To exercise the Single Byte Transfer EndTerm, click the “Byte” button in the Transfer Controls group. Type in several numbers separated by a space and less 256 into the Multiple Byte textbox. Then hit the Multi Byte button. The numbers appear in the Receive Byte textbox.

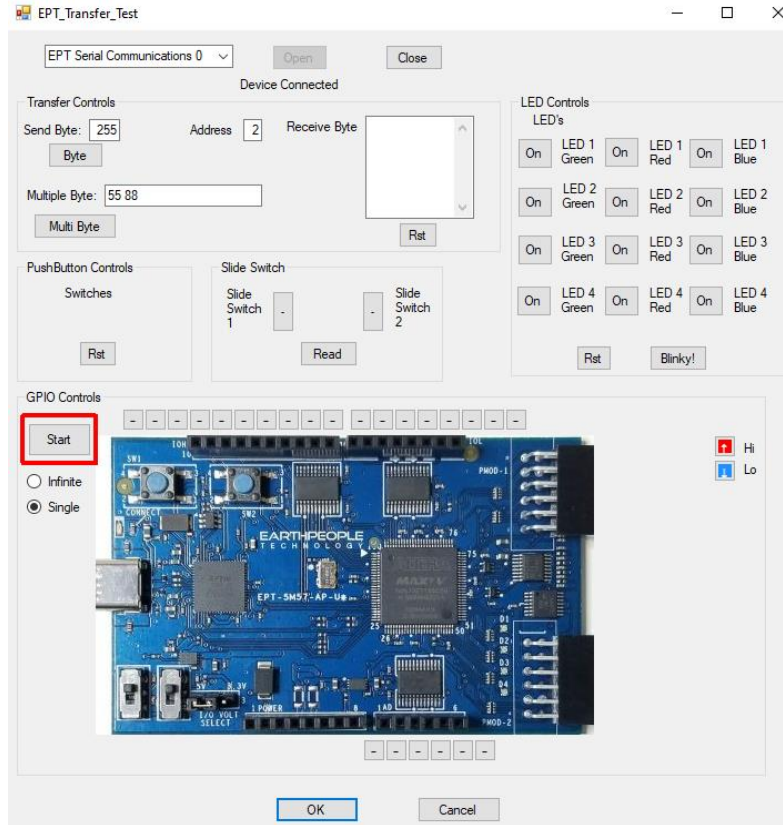
UnoMax CPLD Development System User Manual



To exercise the Block Transfer EndTerm, click the “Start” button in the GPIO Controls group. The UnoMax will sample the state of each Input pin of the CPLD that is connected to a board edge connector. The Transfer Test Window will display the results of each pin, Hi or Lo

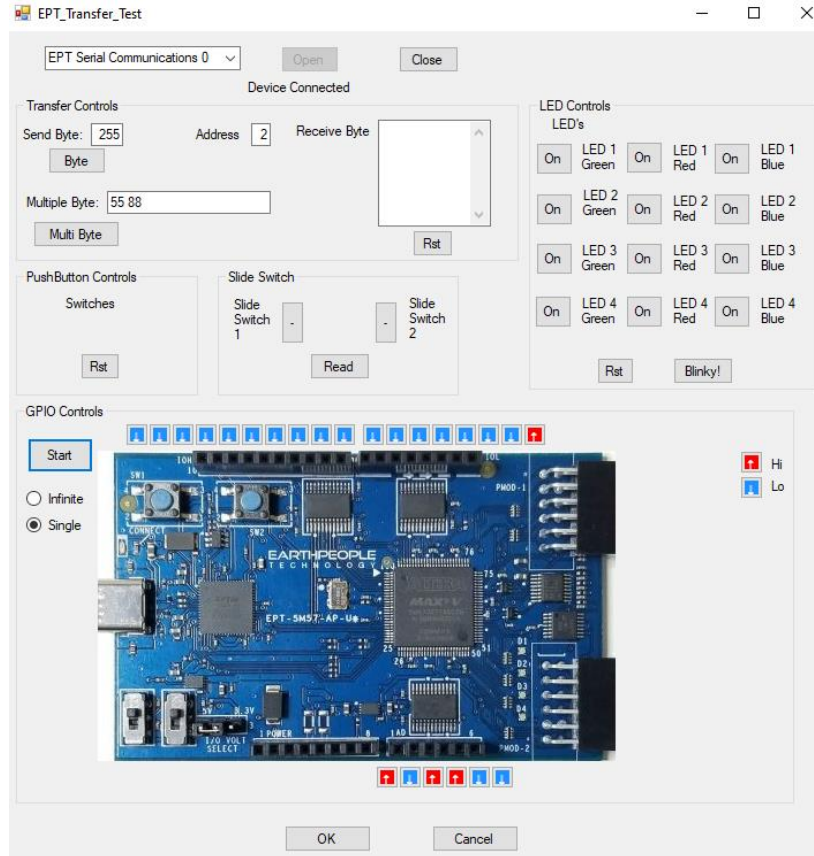


UnoMax CPLD Development System User Manual



The results of each pin are displayed next to the image of the UnoMax in separate buttons.

UnoMax CPLD Development System User Manual



Press the PCB switches on the UnoMax to view the Switch Controls in action.

1.2 EPT-5M57-AP-U3

The UnoMax board (EPT-5M57-AP-U3) is equipped with an Intel/Altera 5M570 CPLD; which is programmed using the Quartus Prime software. The CPLD has 570 Logic Elements which is equivalent to 440 Macrocells. An on board 66 MHz oscillator is used by the EPT Active Transfer Library to provide data transfer rates of up to 0.1 Mega Bytes per second. A four channel ADC with 300KSamples/Second sample rate and dedicated 6 pin header. Twenty Four I/O's from the CPLD are attached to three 8 bit transceivers to provide 3.3V/5 Volt compatible I/O's. These 74LVC245 bidirectional voltage translator/bus transceivers are controlled by one enable and direction bit per transceiver. This means the direction of the individual bits of each transceiver cannot be



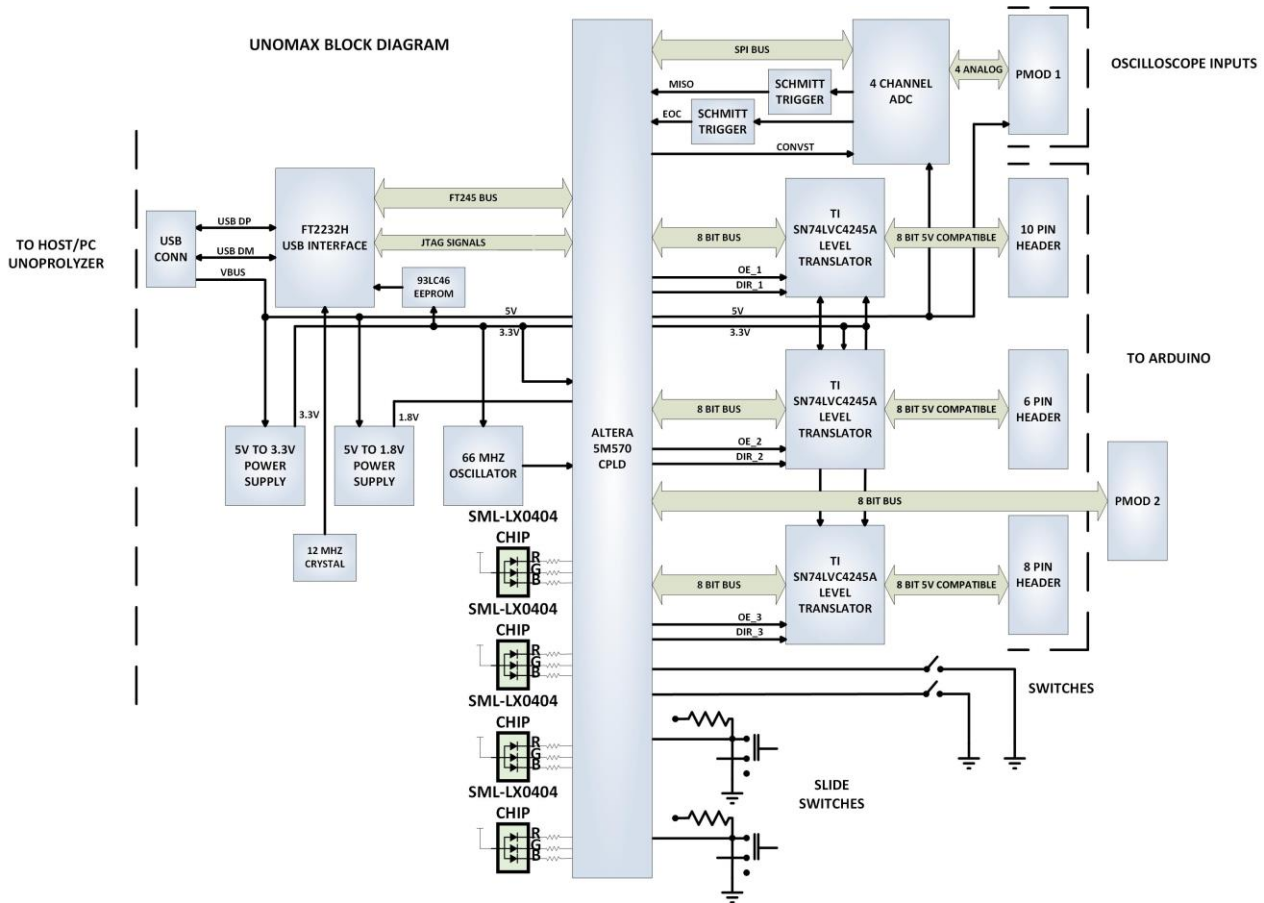
UnoMax CPLD Development System User Manual

selected; the direction is selected for all eight bits per transceiver. There are four RGB LED's and two Push Buttons that are controllable by the user code. The hardware features are as follows.

- Intel/Altera 5M570 CPLD with 440 Macrocells
- 4 Channel ADC 300KSamples/Second
- 66 MHz oscillator for driving USB data transfers and users code
- Three bidirectional voltage translator/bus transceivers
- 24 user Input/Outputs available as three 8 bit ports
- Ports have jumper selectable 3.3V/5 Volt Input/Output
- Four RGB LED's accessible by the user
- Two PCB switches accessible by the user
- Two Slide Switches accessible by the user
- Two PMOD Connetors

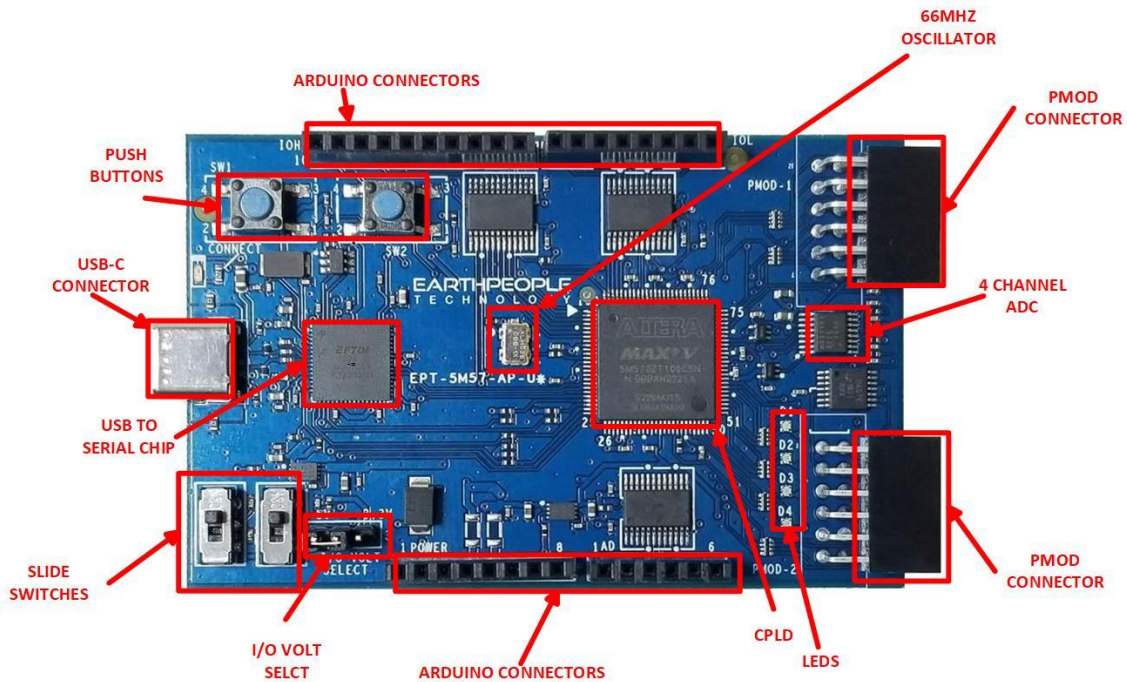
UnoMax

UnoMax CPLD Development System User Manual



UnoMax CPLD Development System User Manual

UnoMax



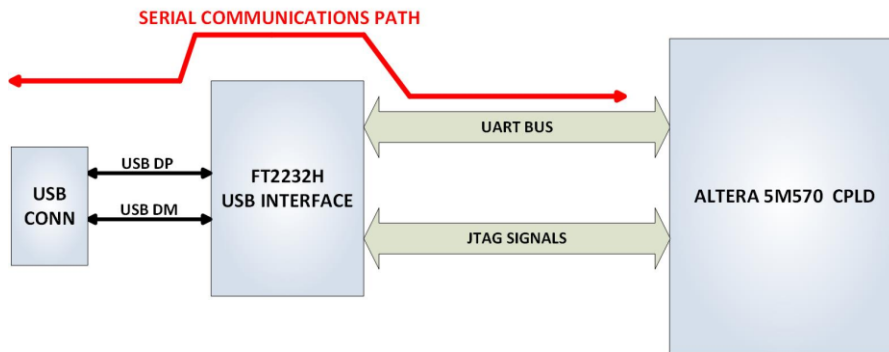
1.2.1 High Speed USB Communications

The UnoMax CPLD Development system connects an FT2232H Dual High Speed USB (480 Mbits/sec) chip to the CPLD. The CPLD uses a dedicated channel on the FT2232H for high speed transfers to the PC. Using the EPT Active Transfer Library, sustained speeds of 8 Mbytes/sec can be achieved. The transfers are bi-directional.

The FT2232H chip provides a means of data conversion from USB to serial/ parallel data and serial/parallel to USB for data being sent from the CPLD to the PC. Channel A is configured as a JTAG bus and Channel B is configured as an Serial bus. CPLD Programming commands are transmitted via the JTAG bus (channel A). Channel B has one dual port 4Kbyte FIFO for transmission from Host PC to the CPLD, it also has one dual port 4Kbyte FIFO for receiving data from the CPLD to the Host PC. The FT2232H chip provides its own 12 MHz clock and +3.3V and +1.8V power supplies. The +3.3V power supply output is used by the UnoMax for all of its +3.3V power budget.

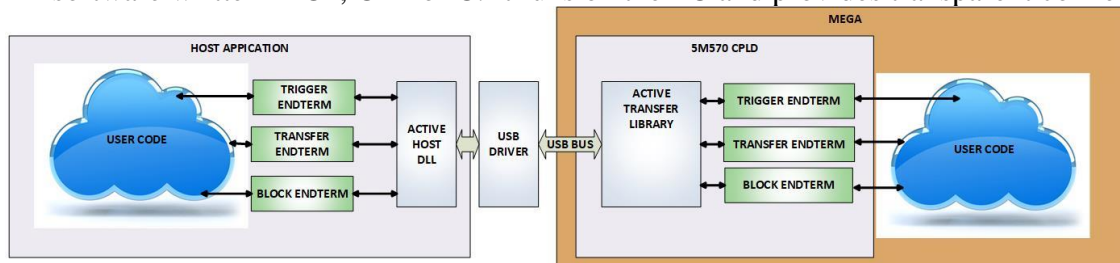
1.2.2 Serial USB Communications

The UnoMax CPLD Development system connects an FT2232H Dual High Speed USB (480 Mbits/sec) chip to the CPLD. The CPLD uses a dedicated channel on the FT2232H for serial transfers to the PC. Using the EPT Active Transfer Library, sustained speeds of 0.1 Mbytes/sec can be achieved. The transfers are bi-directional.



The FT2232H chip provides a means of data conversion from USB to serial data and serial to USB for data being sent from the CPLD to the PC. Channel A is configured as a JTAG bus and Channel B is configured as a single COM Port. Channel B has one dual port 4Kbyte FIFO for transmission from Host PC to the CPLD.

The Serial Communications with the UnoMax is handled easily using the Active Host API. The Active Host SDK is provided as a dll which easily interfaces to application software written in C#, C++ or C. It runs on the PC and provides transparent connection



from PC application code through the USB driver to the user CPLD code. The user code connects to “Endterms” in the Active Host dll. These Host “Endterms” have complementary HDL “Endterms” in the Active Transfer Library. Users have seamless bi-directional communications at their disposal in the form of:

- Trigger Endterm
- Transfer Endterm

UnoMax CPLD Development System User Manual

- Block Endterm

User code writes to the Endterms as function calls. Just include the address of the individual module (there are eight individually addressable modules of each Endterm). Immediately after writing to the selected Endterm, the value is received at the HDL Endterm in the CPLD.

1.2.3 Host PC Connection

The UnoMax includes an LED that signifies the connection of the board with the Host PC. The connect LED has the word “CONNECT” in silkscreen next to the LED. This LED will only light up once the Host PC has correctly enumerated the USB device (FT2232HQ chip). When this LED is lit up it can tell the user three things:

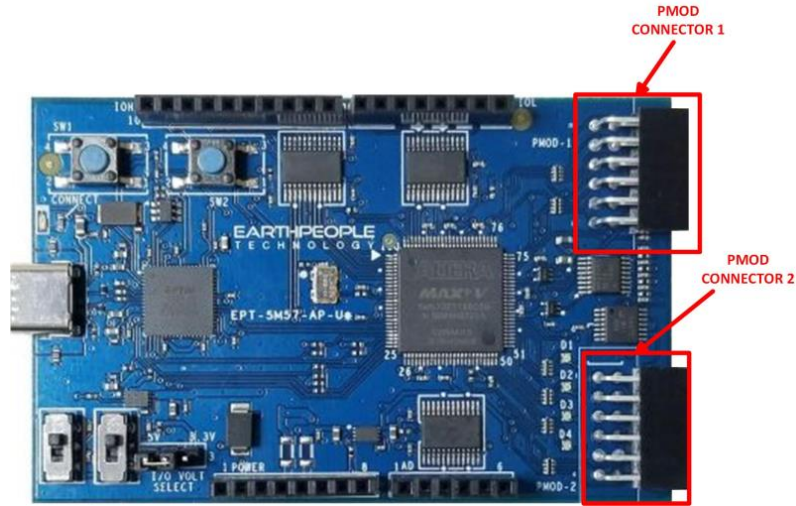
- Power has been applied to the UnoMax via USB
- The FT2232HQ chip is working properly
- The Host PC has found the appropriate driver and will communicate with the UnoMax



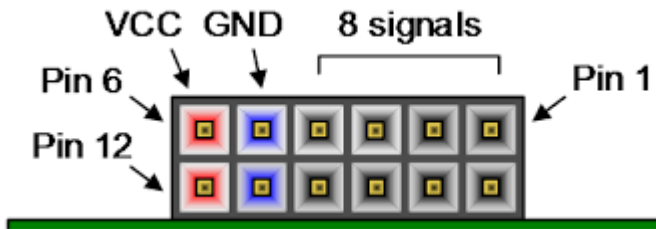
1.2.4 PMOD Connectors

The UnoMax includes two PMOD Connectors. These two connectors are located towards the rear of the board.

UnoMax CPLD Development System User Manual



The PMOD pinouts follows the standard pinout. Pin 1 is located in the upper right when facing the connector.



VCC is +3.3V and the Inputs and Outputs of the 8 signals are +3.3V only. The eight I/O's are connected directly to FPGA pins and can be designated as any communications standard.

PMOD

I2C	Type 1	Type 2	Type 3	Type 4
IO1	IO1	SS	CTS	CTS
IO2	IO5	INT	INT	INT
IO3	IO2	MOSI	RTS	TX
IO4	IO6	RST	RST	RST
SCL	IO3	MISO	IO7	RX
SDA	IO7	IO7	RX	RX
GND	IO4	SCK	IO8	TX
3.3V	IO8	IO8	IO8	IO8
	GND	GND	GND	GND
	3.3V	3.3V	3.3V	3.3V
	3.3V	3.3V	3.3V	3.3V

UnoMax CPLD Development System User Manual

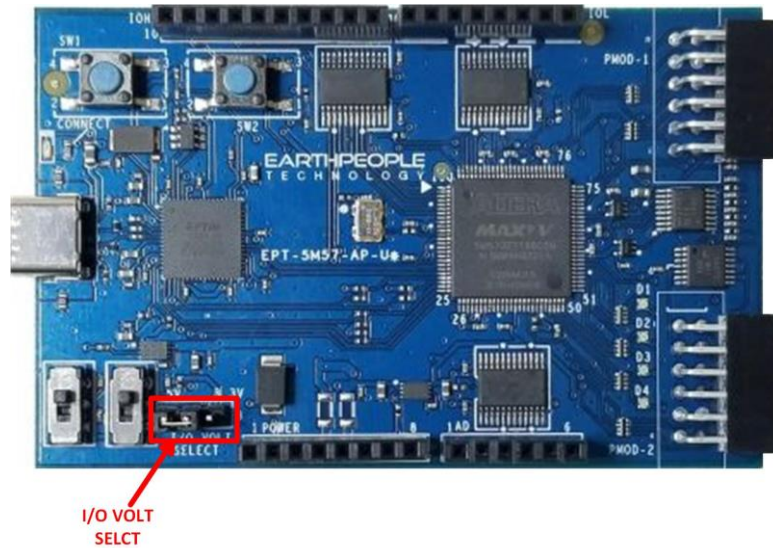
The PMOD have the following connections to the MAX V chip:

PMOD Pin Number	Signal Name	MAX V Pin Number
1-1	PMOD_1_1	83
1-2	PMOD_1_2	81
1-3	PMOD_1_3	77
1-4	PMOD_1_4	58
1-7	PMOD_1_7	76
1-8	PMOD_1_8	78
1-9	PMOD_1_9	82
1-10	PMOD_1_10	84
2-1	ADC_AIN0	NA
2-2	ADC_AIN1	NA
2-3	ADC_AIN2	NA
2-4	ADC_AIN3	NA
2-7	PMOD_2_7	76
2-8	PMOD_2_8	78
2-9	PMOD_2_9	82
2-10	PMOD_2_10	84

1.2.5 Inputs and Outputs

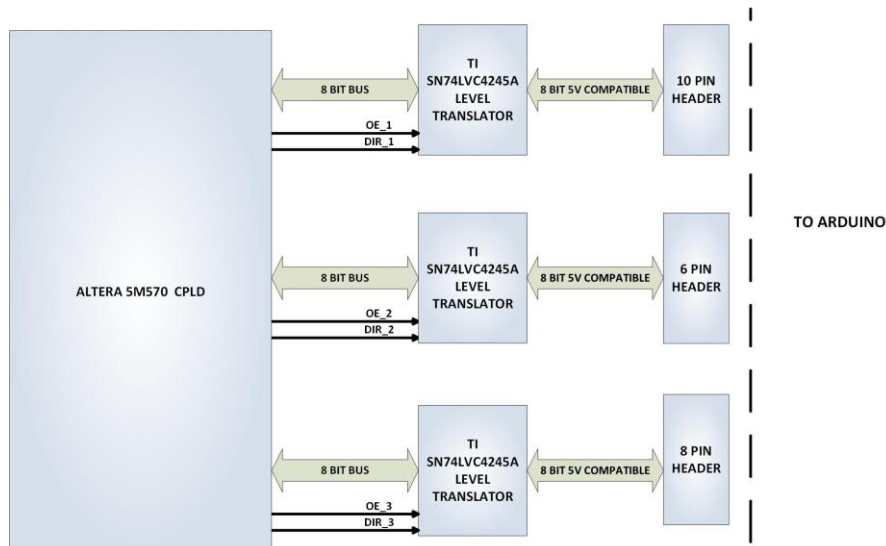
There are 24 Inputs/Outputs which are selectable between +3.3V and +5 Volt. JMP1 is used to select which voltage the 24 Inputs/Outputs are set to.

UnoMax CPLD Development System User Manual



The I/O's are organized as three 8 bit directional ports. Each port must be defined as input or output. This means that all 8 bits of a port will point in the same direction, depending on the direction bit of the transceiver. The direction bit can be changed at any time, so that a port can change from input to output in minimum setup time of 6 nanoseconds. Each port also has an enable pin. This enable pin will enable or disable the bits of the port. If the port is disabled, the bits will “float”.

TRANSCIVERS PROVIDE PROTECTION FROM OVERVOLTAGE/
OVERCURRENT BETWEEN USER HEADERS AND CPLD I/O's



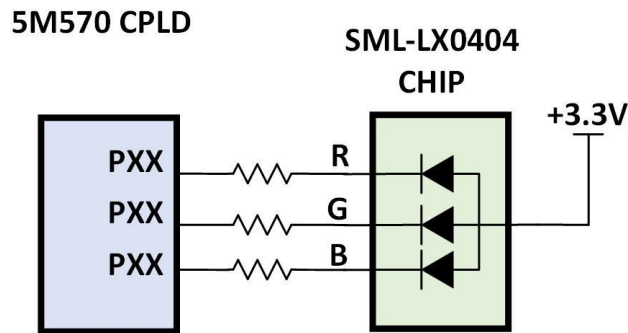
UnoMax CPLD Development System User Manual

1.2.6 User LEDs

The User LEDs are four RGB LEDs. These LEDs are SML-LX0404 chips and are for use only with +3.3V.



The SML-LX0404 chip is a current sink and are connected to pins on the MAX V CPLD. The anode is connected to +3.3V. The series resistors are calculated for current limiting based on +3.3V.



Each series resistor uses a 220 Ohm in a resistor array. In order to light up the each LED, the user code must assert a zero on the associated signal for the LED. To turn off the LED, assert High Z on the signal.

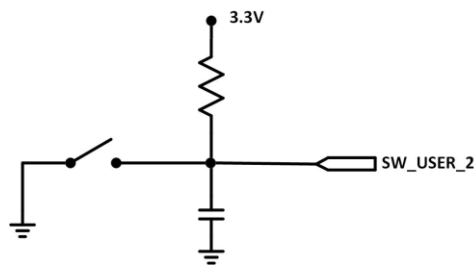
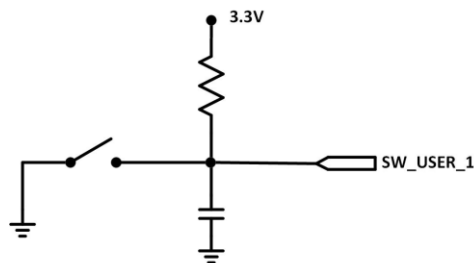
UnoMax CPLD Development System User Manual

The LED RGB signals are organized on the following pins from the MAX V chip:

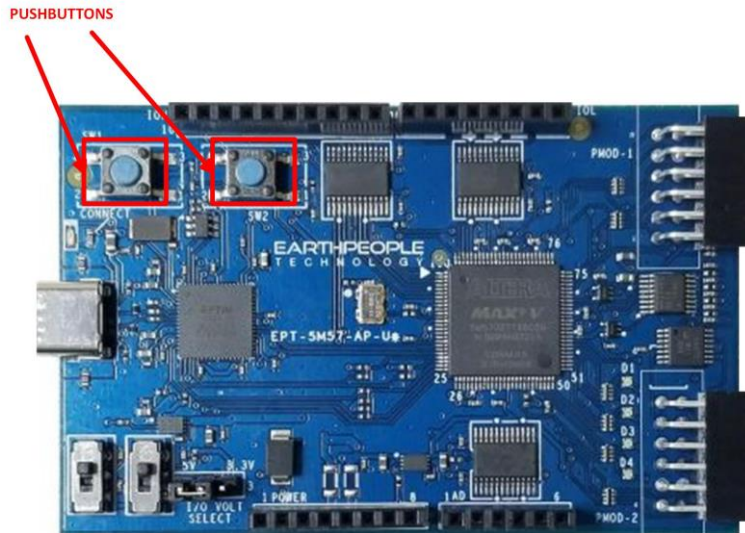
LED Number	Signal Name	MAX V Pin Number
D1	LED_GREEN_1_N	57
D1	LED_BLUE_1_N	56
D1	LED_RED_1_N	55
D2	LED_GREEN_2_N	54
D2	LED_BLUE_2_N	53
D2	LED_RED_2_N	52
D3	LED_GREEN_3_N	51
D3	LED_BLUE_3_N	50
D3	LED_RED_3_N	49
D4	LED_GREEN_4_N	48
D4	LED_BLUE_4_N	47
D4	LED_RED_4_N	43

1.2.7 User Push Buttons

The UnoMax includes two push button switches. Both are momentary contact switches. They include a 1uF cap to ground to debounce both switches.



UnoMax CPLD Development System User Manual

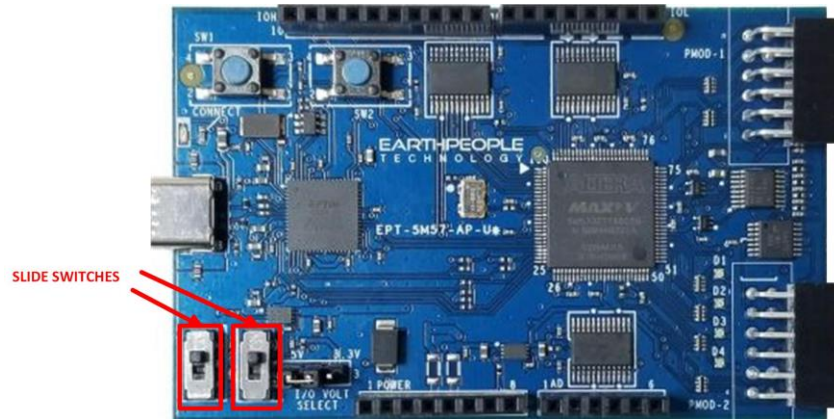
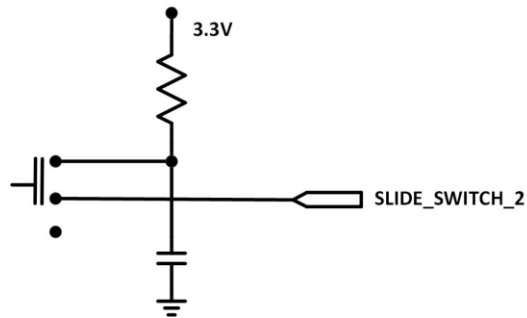
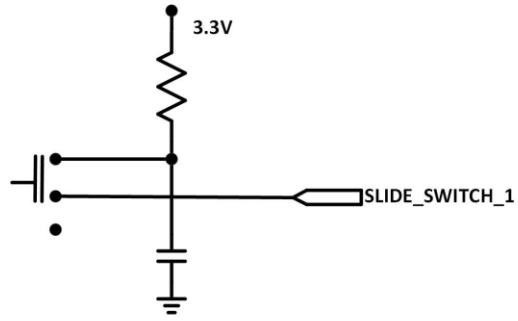


Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
SW1	SW_USER_1	20	SW_USER_1
SW2	SW_USER_2	21	SW_USER_2

1.2.8 Slide Switches

The UnoMax includes two slide switches. Both are full contact switches. They include a 1uF cap to ground to debounce both switches.

UnoMax CPLD Development System User Manual

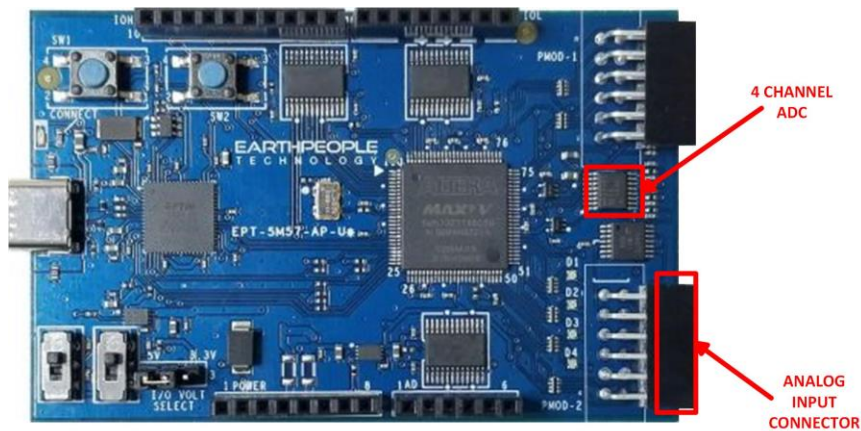


Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
SW3	SLIDE_SWITCH_1	16	SLIDE_SWITCH_1
SW4	SLIDE_SWITCH_2	17	SLIDE_SWITCH_2

UnoMax CPLD Development System User Manual

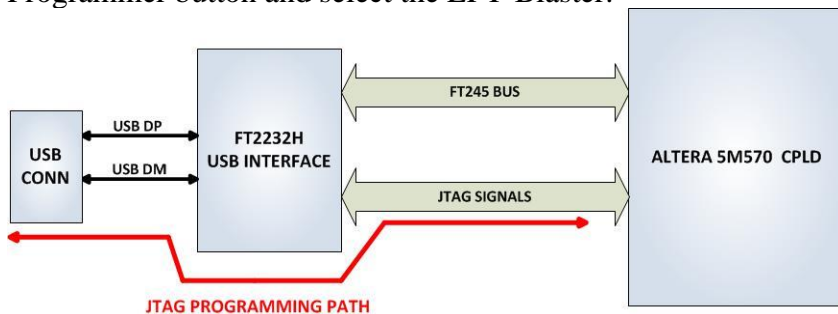
1.2.9 Analog Inputs

The UnoMax includes a six pin analog input connector. This connector provides a path from the pins to the input of the four Op-Amp buffers. Each Op-Amp includes a 1MHz low pass filter. Each Op-Amp provides a buffer for the analog signals to the ADC inputs.



1.2.10 JTAG

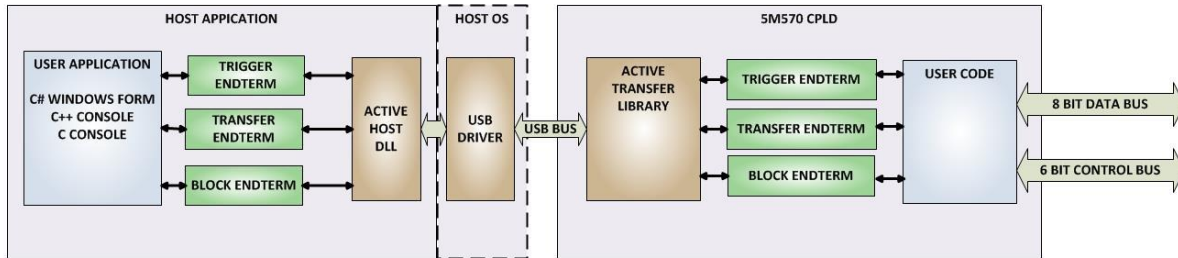
The UnoMax uses the second channel of the FT2232H chip as a dedicated CPLD programming port. The CPLD must be programmed via JTAG signals and the FT2232H has built in JTAG signals. The CPLD can be programmed directly from Quartus Prime Lite by using the “jtag_hw_mbfidi_blaster.dll”. Just click on the Programmer button and select the EPT-Blaster.



1.3 Active Host EndTerms

The Active Host SDK is provided as a dll which easily interfaces to application software written in C#, C++ or C. It runs on the PC and provides transparent connection

UnoMax CPLD Development System User Manual



from PC application code through the USB driver to the user CPLD code. The user code connects to “Endterms” in the Active Host dll. These Host “Endterms” have complementary HDL “Endterms” in the Active Transfer Library. Users have seamless bi-directional communications at their disposal in the form of:

- Trigger Endterm
- Transfer Endterm
- Block Endterm

User code writes to the Endterms as function calls. Just include the address of the individual module (there are eight individually addressable modules of each Endterm). Immediately after writing to the selected Endterm, the value is received at the HDL Endterm in the CPLD.

Receiving data from the CPLD is made simple by Active Host. Active Host transfers data from the CPLD as soon as it is available. It stores the transferred data into circular buffer. When the transfer is complete, Active Host invokes a callback function which is registered in the users application. This callback function provides a mechanism to transparently receive data from the CPLD. The user application does not need to schedule a read from the USB or call any blocking threads.

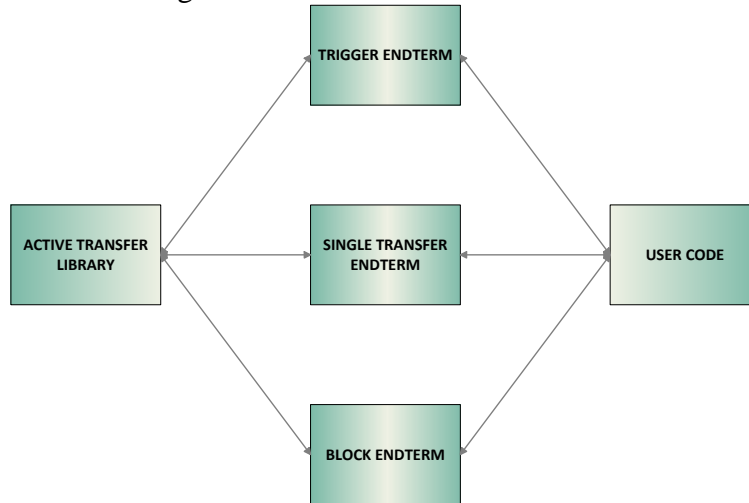
1.4 Active Transfer EndTerms

The Active Transfer Library is a portfolio of HDL modules that provides an easy to use yet powerful USB transfer mechanism. The user HDL code communicates with EndTerms in the form of modules. These EndTerm modules are commensurate with the Active Host EndTerms. There are three types of EndTerms in the Active Transfer Library:

- Trigger Endterm
- Transfer Endterm
- Block Endterm

UnoMax CPLD Development System User Manual

They each have a simple interface that the user HDL code can use to send or receive data across the USB. Writing to an EndTerm will cause the data to immediately arrive



at the commensurate EndTerm in the Active Host/user application. The transfer through the USB is transparent. User HDL code doesn't need to set up Endpoints or respond to Host initiated data requests. The whole process is easy yet powerful.

2 EPT Drivers

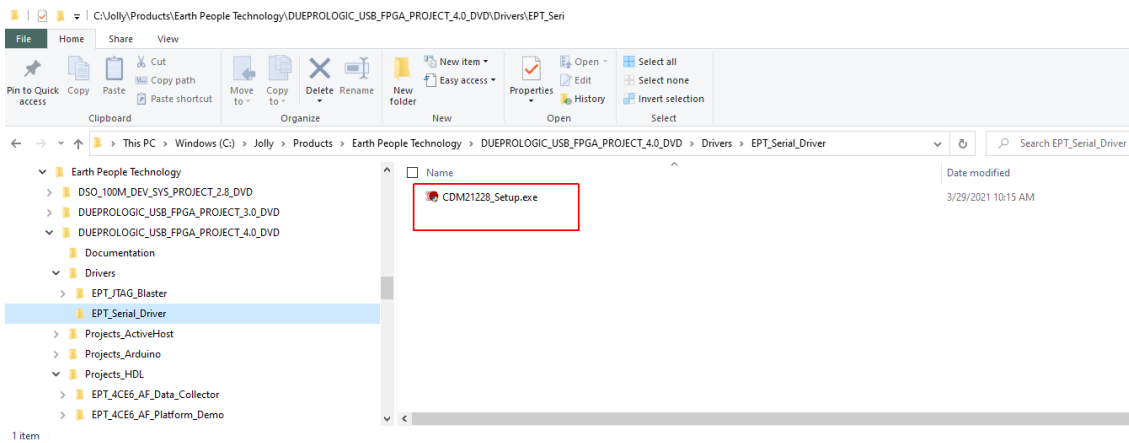
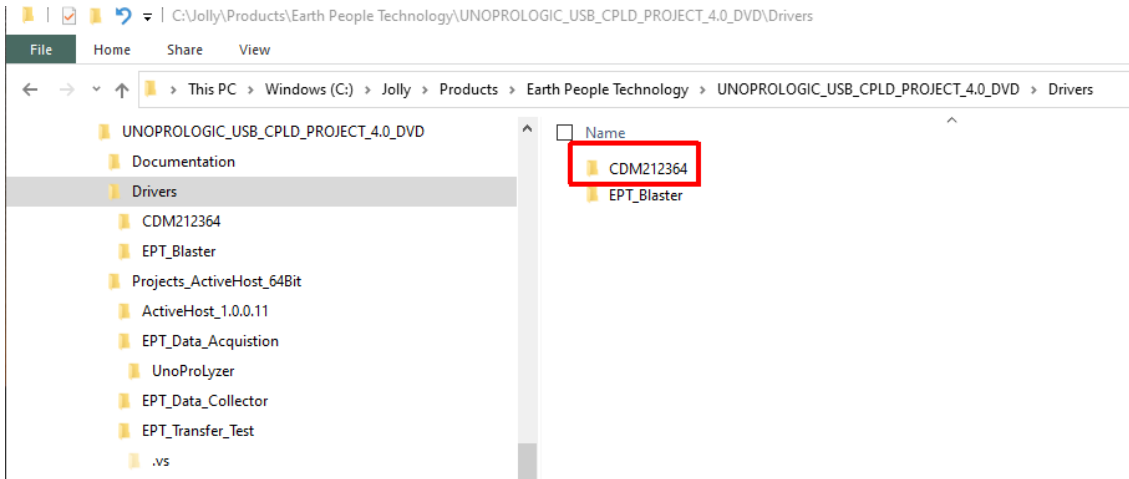
The UnoMax Development system requires drivers for any interaction between PC and the board. The communication between the two consists of programming the CPLD and data transfer. In both cases, the USB Driver is required. This will allow Windows to recognize the USB Chip and setup a pathway for Windows to communicate with the USB hardware.

2.1 USB Driver

The UnoMax uses an FTDI FT2232H USB to Serial chip. This chip provides the USB interface to the PC and the serial/FIFO interface to the CPLD. The FT2232H requires the use of the EPT USB driver. To install the driver onto your PC, use the CDM212xxx Folder. The installation of the FTDI 2.12.28 driver is easily accomplished by double clicking the CDM21228_Setup.exe.

Locate the CDM212xxx folder in the Drivers folder of the UnoMax Development System CD using Windows Explorer.

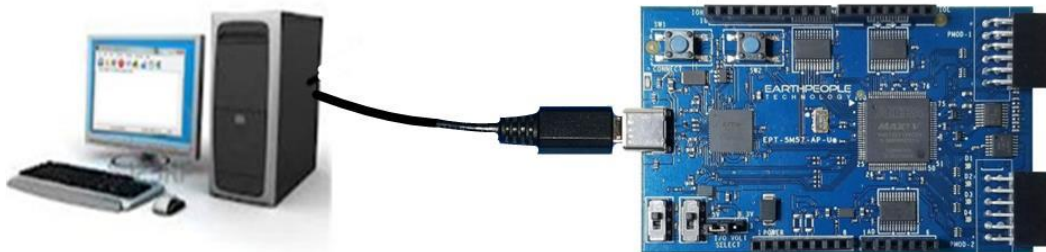
UnoMax CPLD Development System User Manual



Double click on the *.exe file and select the default settings when the software tool queries the user.

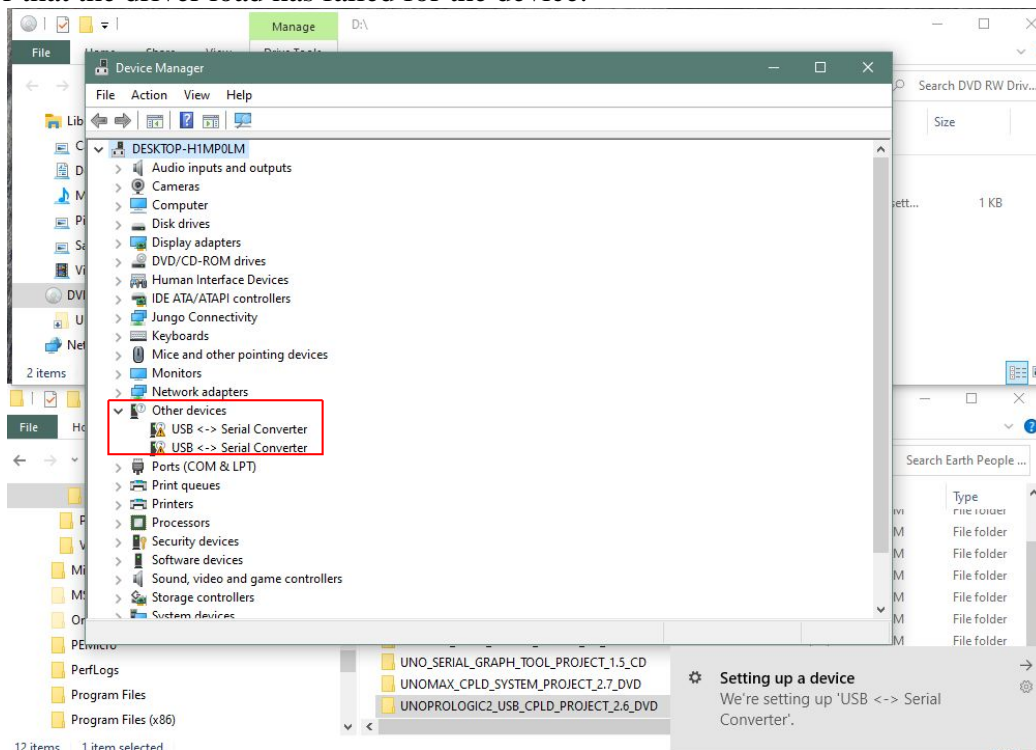
Plug in the UnoMax device into an available USB port.

UnoMax CPLD Development System User Manual



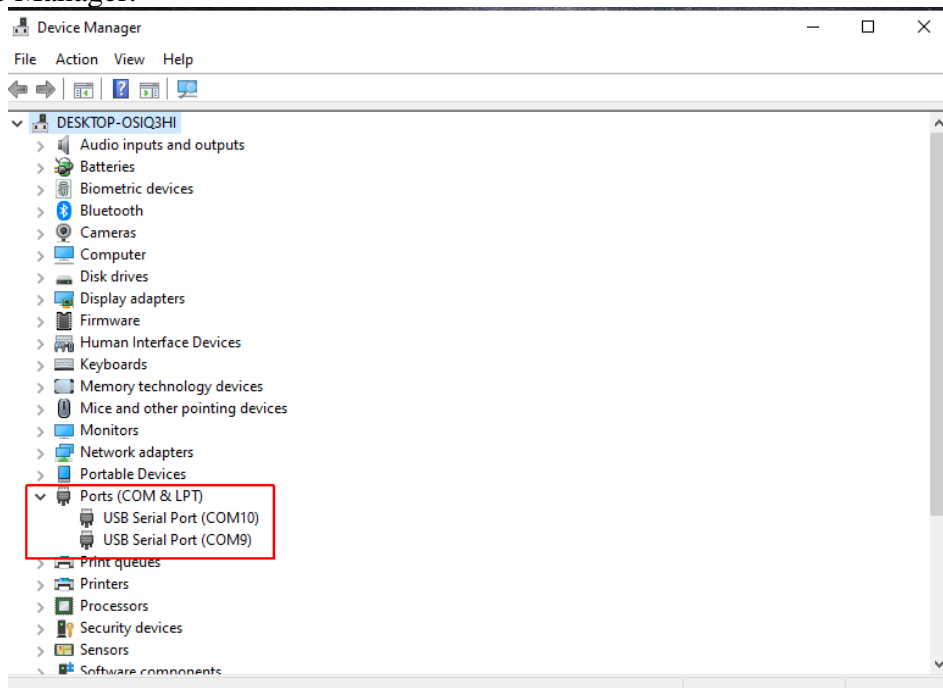
Windows will attempt to locate a driver for the USB device. When it does not find one, it will report an error, “Device driver software was not successfully installed”. Ignore this error.

If Windows cannot load a driver for the UnoMax, a notification window will inform the user that the driver load has failed for the device.



UnoMax CPLD Development System User Manual

If the driver is successfully installed, Windows will inform the user. The user can check Device Manager to ensure the correct driver was installed for the UnoMax. The UnoMax will show up as two COM Ports under the “Ports (COM & LPT)” under the Device Manager.



When this is complete, the drivers are installed and the UnoMax can be used for programming and USB data transfers.

2.2 *JTAG DLL Insert to Quartus Prime Lite*

The JTAG DLL Insert to Quartus Prime Lite allows the Programmer Tool under Quartus to recognize the UnoMax. The UnoMax can then be selected and perform programming of the CPLD. The file, `jtag_hw_mbftdi_blaster.dll` must be placed into the folder that hosts the `jtag_server` for Quartus.

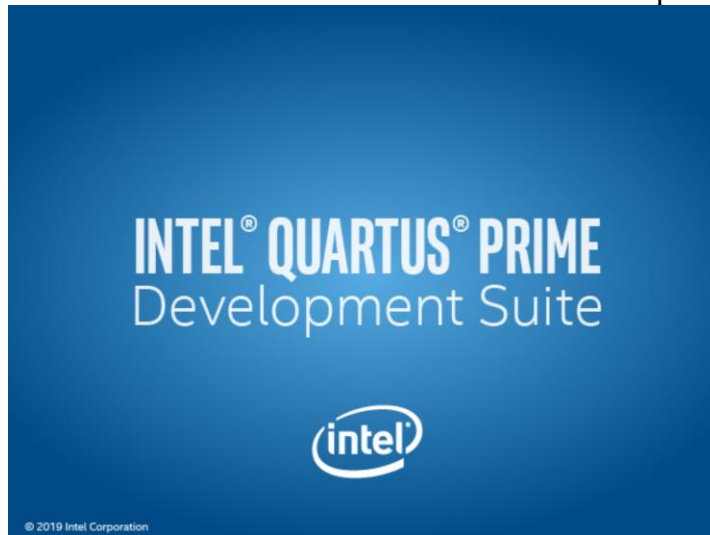
2.2.1 **Installing Quartus**

You can download the Quartus Prime Lite by following the directions in the Section Downloading Quartus.



UnoMax CPLD Development System User Manual

If you don't need to download Quartus, double click on the QuartusLiteSetup-xxx.xxx.xxx-windows .exe (the xxx is the build number of the file, it is subject to change). The Quartus Prime Web Edition will start the installation process.



When the install shield window pops up click "Yes" or if needed, enter the administrator password for the users PC. Click "Ok"

Next, skip the "Download Quartus" section. Go down to the "Quartus Installer" section to complete the Quartus installation.



UnoMax CPLD Development System User Manual

2.2.2 Downloading Quartus

The first thing to do in order build a project in Quartus is to download and install the application. You can find the latest version of Quartus at:

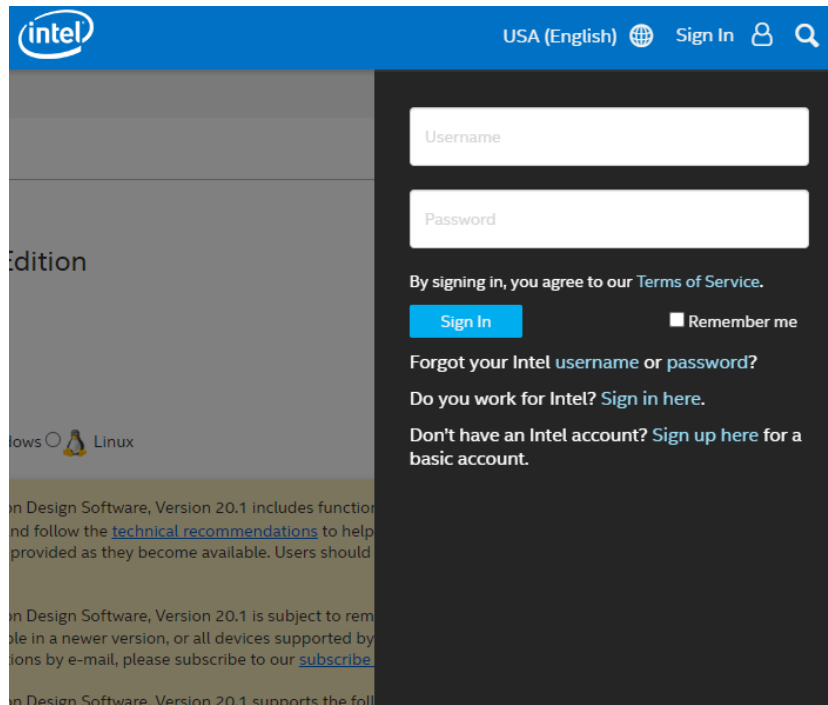
[Intel FPGA Quartus Prime Lite](#)

You will first need to apply for an account with Intel. Then use your login and password to access the download site. Click on the Download Windows Version.

The screenshot shows the Intel website's download center for FPGAs. At the top, there is a blue navigation bar with "Products", "Solutions", and "Support" on the left, the Intel logo in the center, and "USA (English)", a globe icon, a "Sign In" button with a user icon, and a search icon on the right. Below the navigation bar, the page title is "Download Center for FPGAs". On the left side, there is a vertical menu with blue buttons for "Design Software", "Embedded Software", "Archives", "Licensing", "Programming Software", "Drivers", "Board System Design", "Board Layout and Test", and "Legacy Software". The main content area is titled "Quartus Prime Lite Edition" and includes the following information: "Release date: June, 2020", "Latest Release: v20.1", "Select edition: Lite" (with a dropdown arrow), and "Select release: 20.1" (with a dropdown arrow). Below this, it says "Operating System" with icons for Windows and Linux. On the right side, there is a logo for "Intel' Quartus' Prime Design Software". At the bottom of the main content area, there is a yellow box containing two green checkmarks and text: "The Quartus Prime Lite Edition Design Software, Version 20.1 includes functional and security updates. Users should keep their software up-to-date and follow the technical recommendations to help improve security. Additional security updates are planned and will be provided as they become available. Users should promptly install the latest version upon release." and "The Quartus Prime Lite Edition Design Software, Version 20.1 is subject to removal from the web when support for all devices in this release are available in a newer version, or all devices supported by this version are obsolete. If you would like to receive customer notifications by e-mail, please subscribe to our subscribe to our customer notification mailing list."

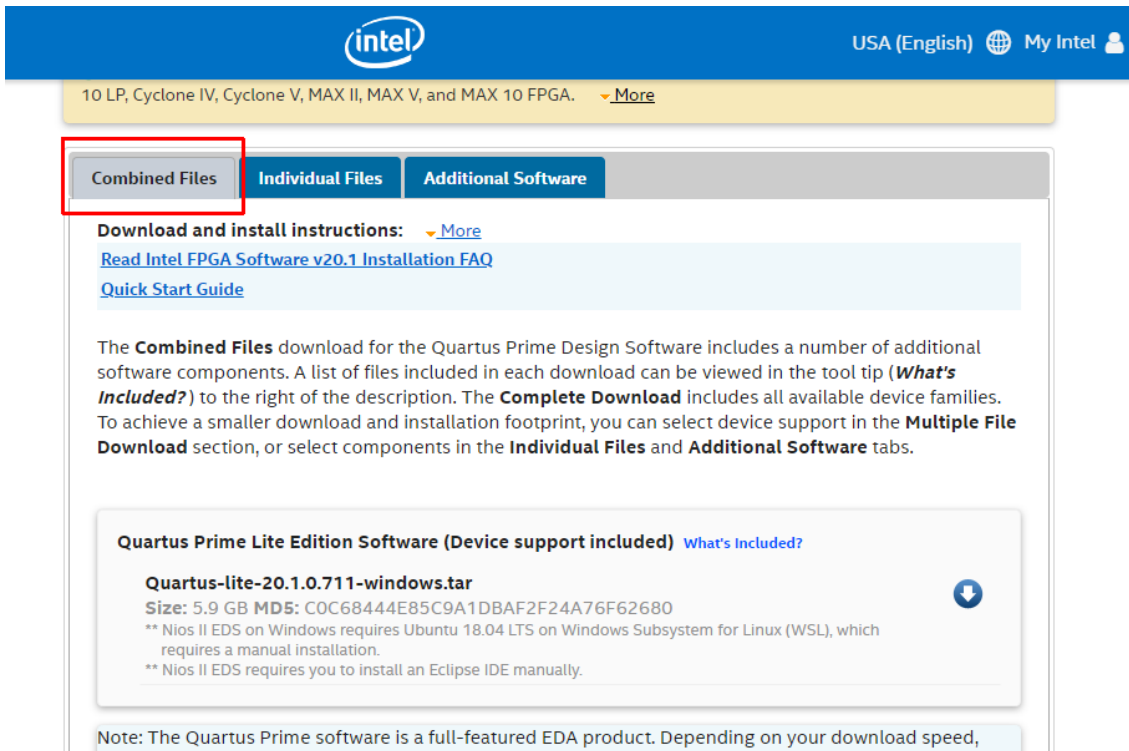
The next page will require you to sign into your “myAltera” account. If you do not have one, follow the directions under the box, “Don’t have an account?”

UnoMax CPLD Development System User Manual



Once you have created your myAltera account, enter the User Name and Password. The next window will ask you to allow pop ups so that the file download can proceed.

UnoMax CPLD Development System User Manual




10 LP, Cyclone IV, Cyclone V, MAX II, MAX V, and MAX 10 FPGA. [More](#)

Combined Files Individual Files Additional Software

Download and install instructions: [More](#)
[Read Intel FPGA Software v20.1 Installation FAQ](#)
[Quick Start Guide](#)

The **Combined Files** download for the Quartus Prime Design Software includes a number of additional software components. A list of files included in each download can be viewed in the tool tip (**What's Included?**) to the right of the description. The **Complete Download** includes all available device families. To achieve a smaller download and installation footprint, you can select device support in the **Multiple File Download** section, or select components in the **Individual Files** and **Additional Software** tabs.

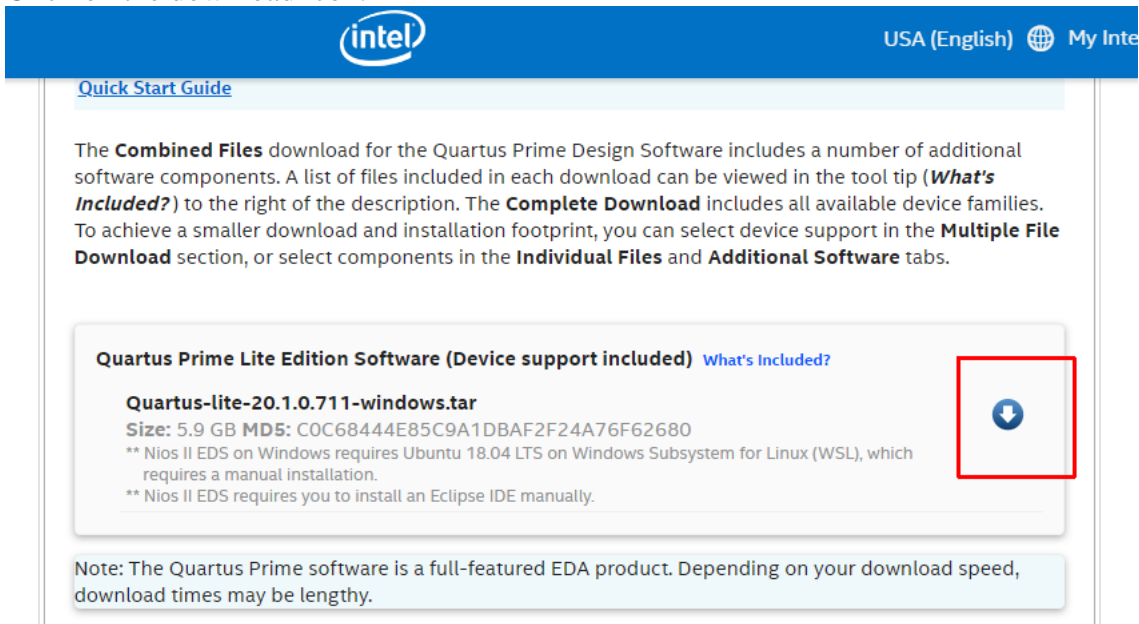
Quartus Prime Lite Edition Software (Device support included) [What's Included?](#)

Quartus-lite-20.1.0.711-windows.tar 

Size: 5.9 GB MD5: COC68444E85C9A1DBAF2F24A76F62680
 ** Nios II EDS on Windows requires Ubuntu 18.04 LTS on Windows Subsystem for Linux (WSL), which requires a manual installation.
 ** Nios II EDS requires you to install an Eclipse IDE manually.

Note: The Quartus Prime software is a full-featured EDA product. Depending on your download speed,


Click on the download icon.



[Quick Start Guide](#)

The **Combined Files** download for the Quartus Prime Design Software includes a number of additional software components. A list of files included in each download can be viewed in the tool tip (**What's Included?**) to the right of the description. The **Complete Download** includes all available device families. To achieve a smaller download and installation footprint, you can select device support in the **Multiple File Download** section, or select components in the **Individual Files** and **Additional Software** tabs.

Quartus Prime Lite Edition Software (Device support included) [What's Included?](#)

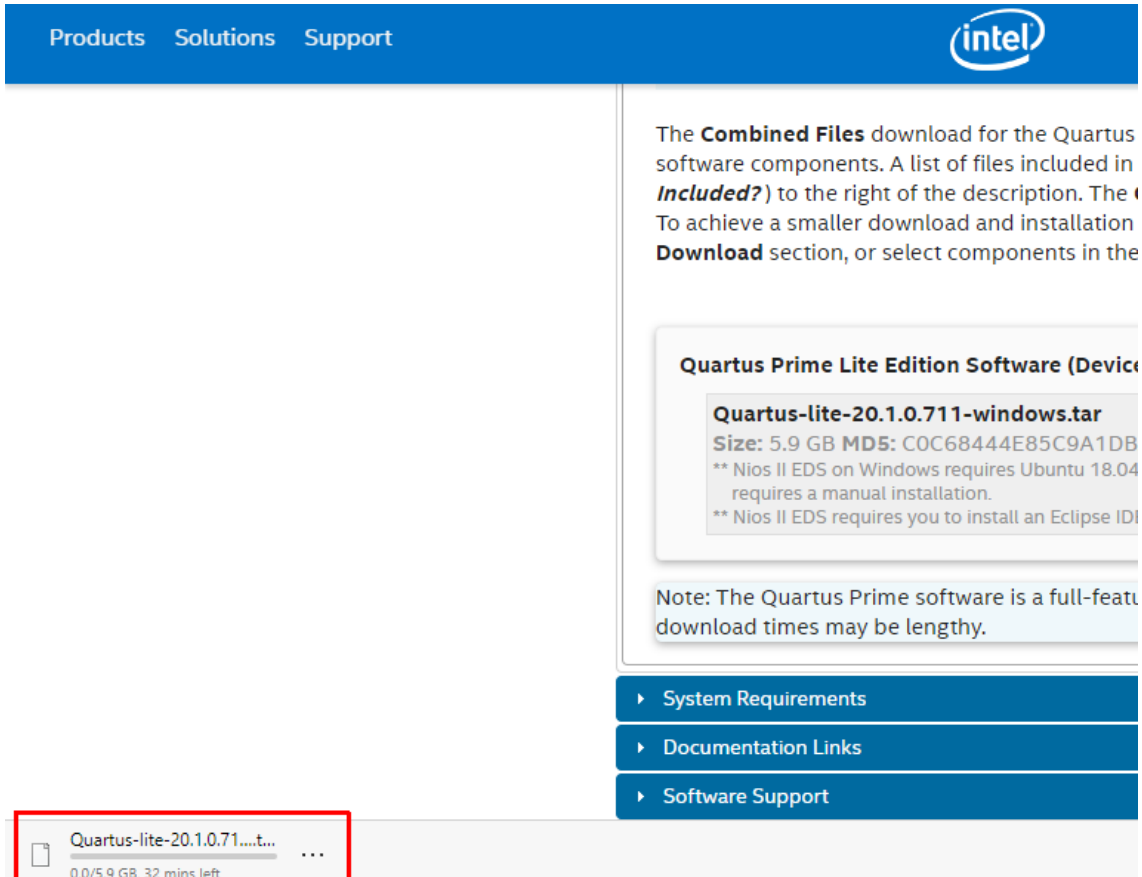
Quartus-lite-20.1.0.711-windows.tar 

Size: 5.9 GB MD5: COC68444E85C9A1DBAF2F24A76F62680
 ** Nios II EDS on Windows requires Ubuntu 18.04 LTS on Windows Subsystem for Linux (WSL), which requires a manual installation.
 ** Nios II EDS requires you to install an Eclipse IDE manually.

Note: The Quartus Prime software is a full-featured EDA product. Depending on your download speed, download times may be lengthy.

UnoMax CPLD Development System User Manual

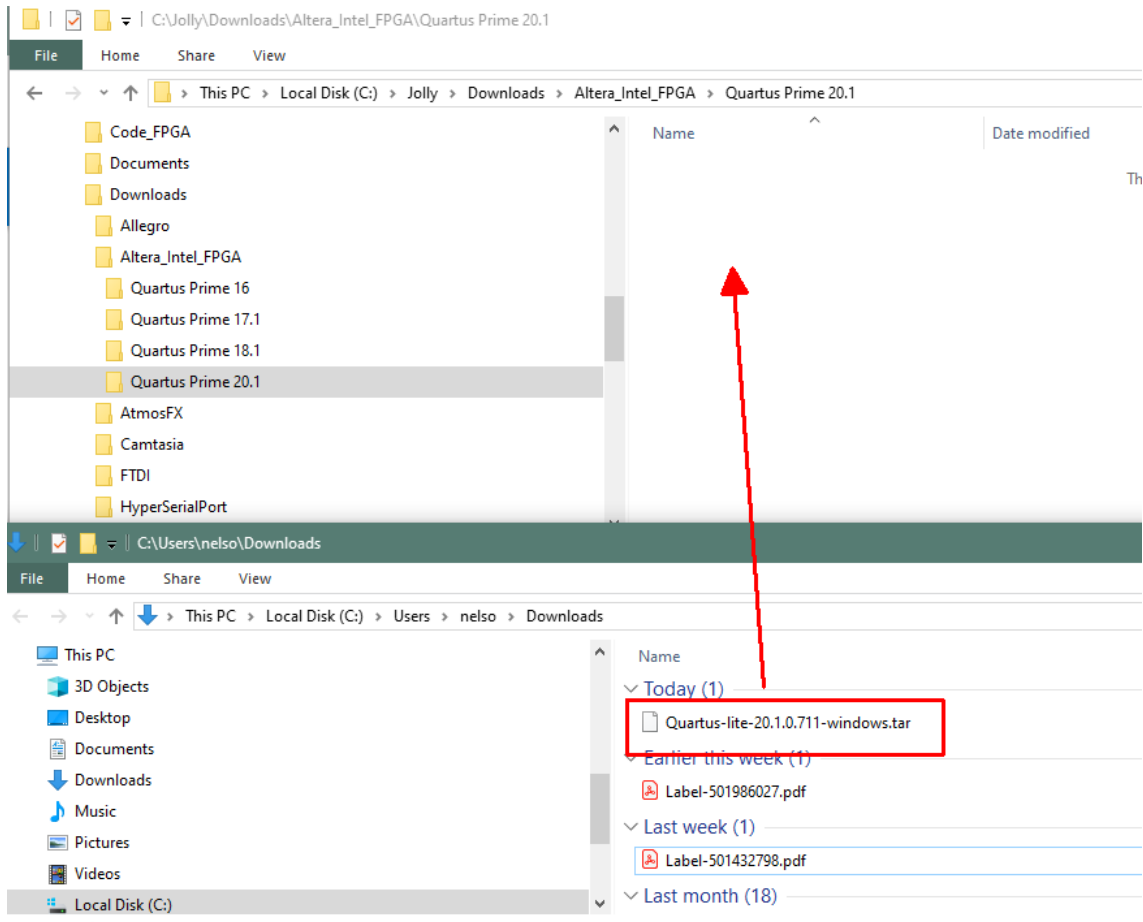
This will start the download.



The screenshot shows the Intel website interface for downloading Quartus Prime Lite Edition Software. The top navigation bar includes "Products", "Solutions", "Support", and the Intel logo. The main content area contains a description of the "Combined Files" download, a list of included files, and a note about the software being a full-featured download. Below the description are three blue buttons: "System Requirements", "Documentation Links", and "Software Support". At the bottom, a download progress bar is visible, showing the file name "Quartus-lite-20.1.0.711...t..." and the progress "0.0/5.9 GB, 32 mins left". The progress bar is highlighted with a red box.

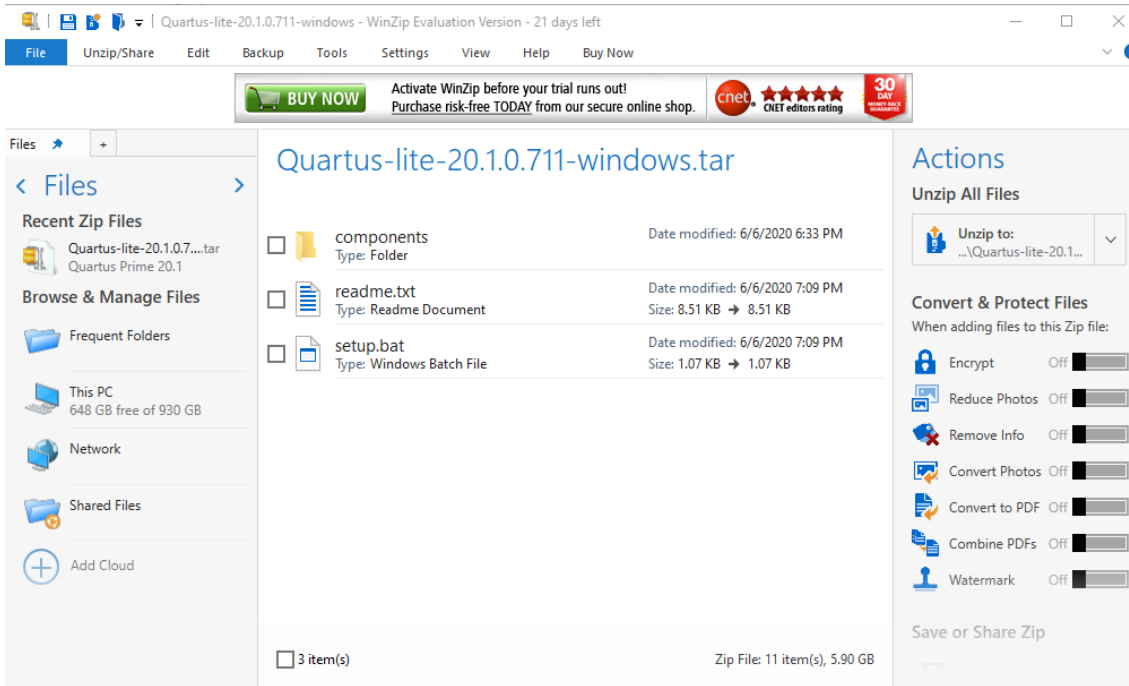
The file is 5.9 GB, so this could take a couple of hours depending on your internet connection. When download is complete, store the *.tar file in a directory on your PC.

UnoMax CPLD Development System User Manual

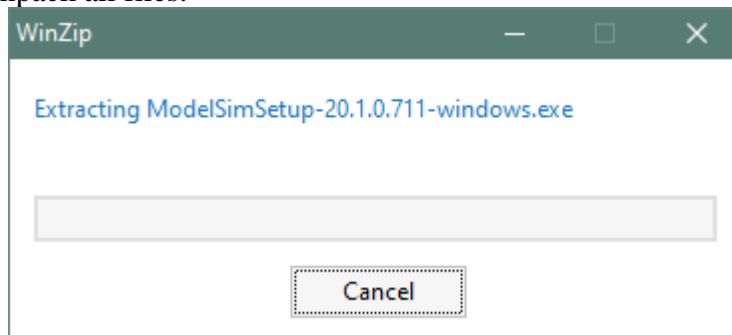


Use a tool such as WinZip to Extract the *.tar file.

UnoMax CPLD Development System User Manual



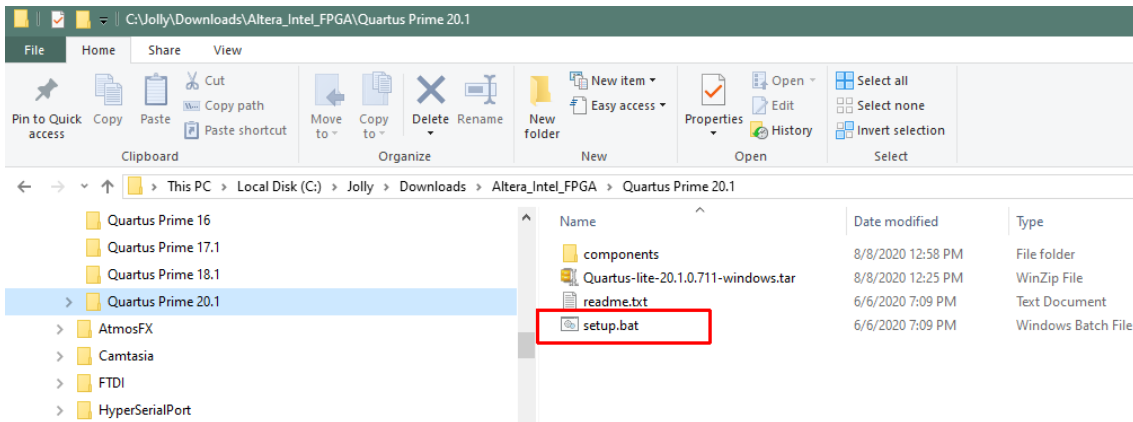
The tool will unpack all files.



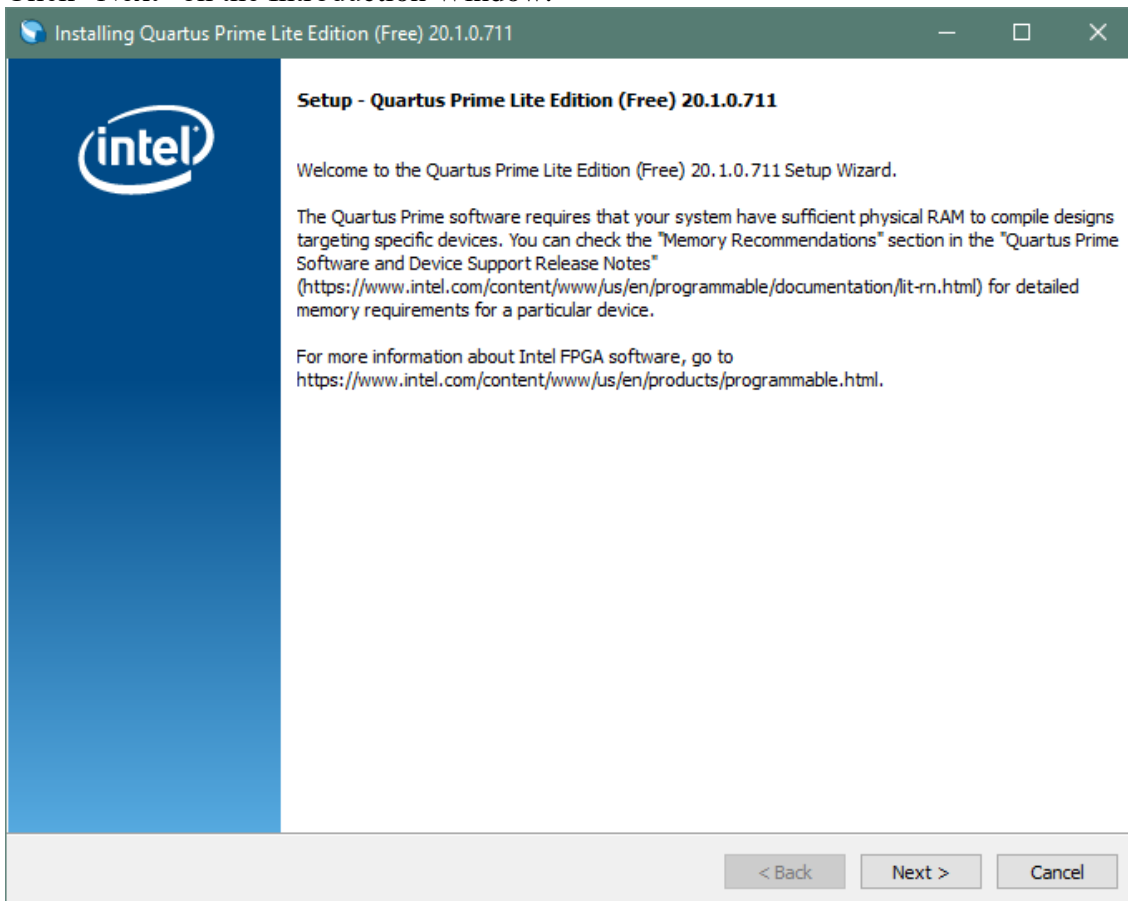
2.2.3 Quartus Installer

When the unpacking finishes from the previous section, double click the setup.bat file in the download folder.

UnoMax CPLD Development System User Manual

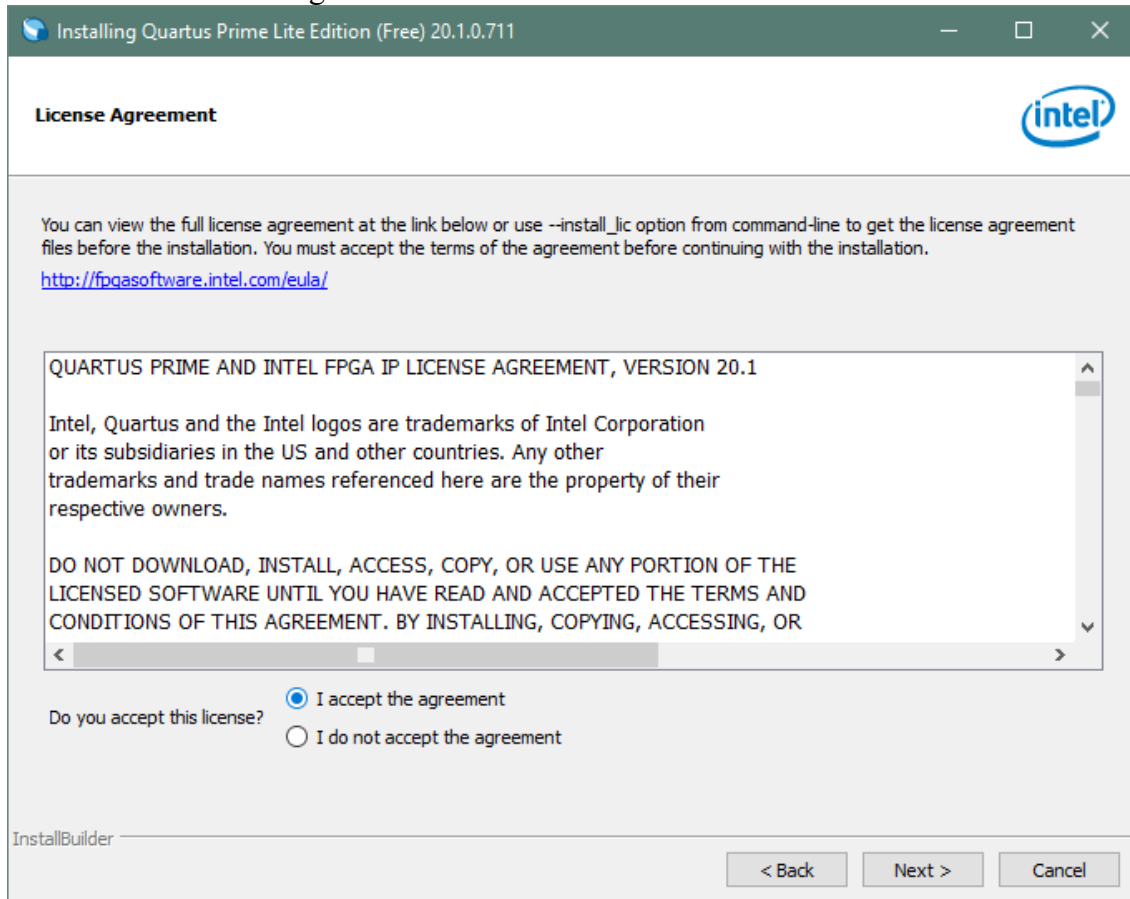


Click “Next” on the Introduction Window.



UnoMax CPLD Development System User Manual

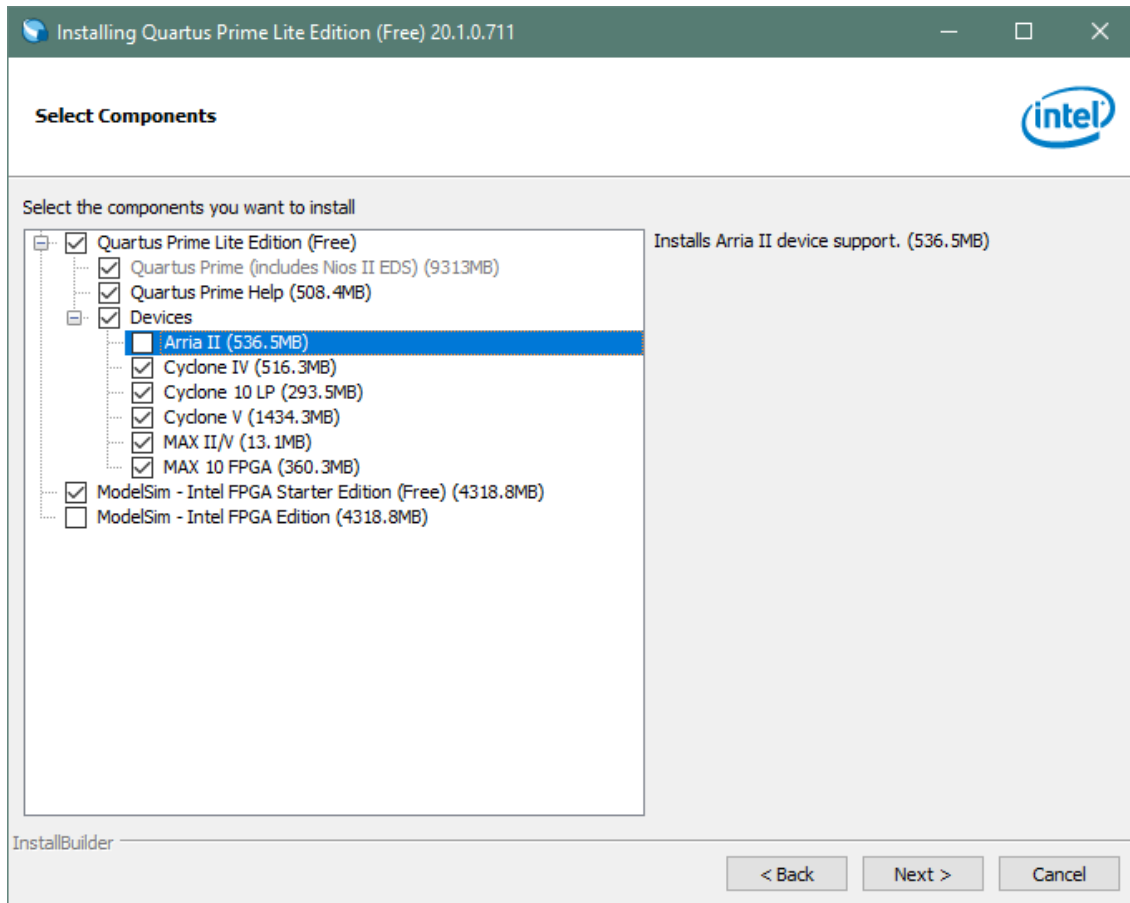
Click the checkbox to agree to the license terms. Then click “Next”.



Click “Next” and accept the defaults.

At the Select Products Window, de-select the Quartus Prime Subscription Edition by clicking on its check box so that the box is not checked. Then click on the check box by the Quartus Prime Web Edition (Free).

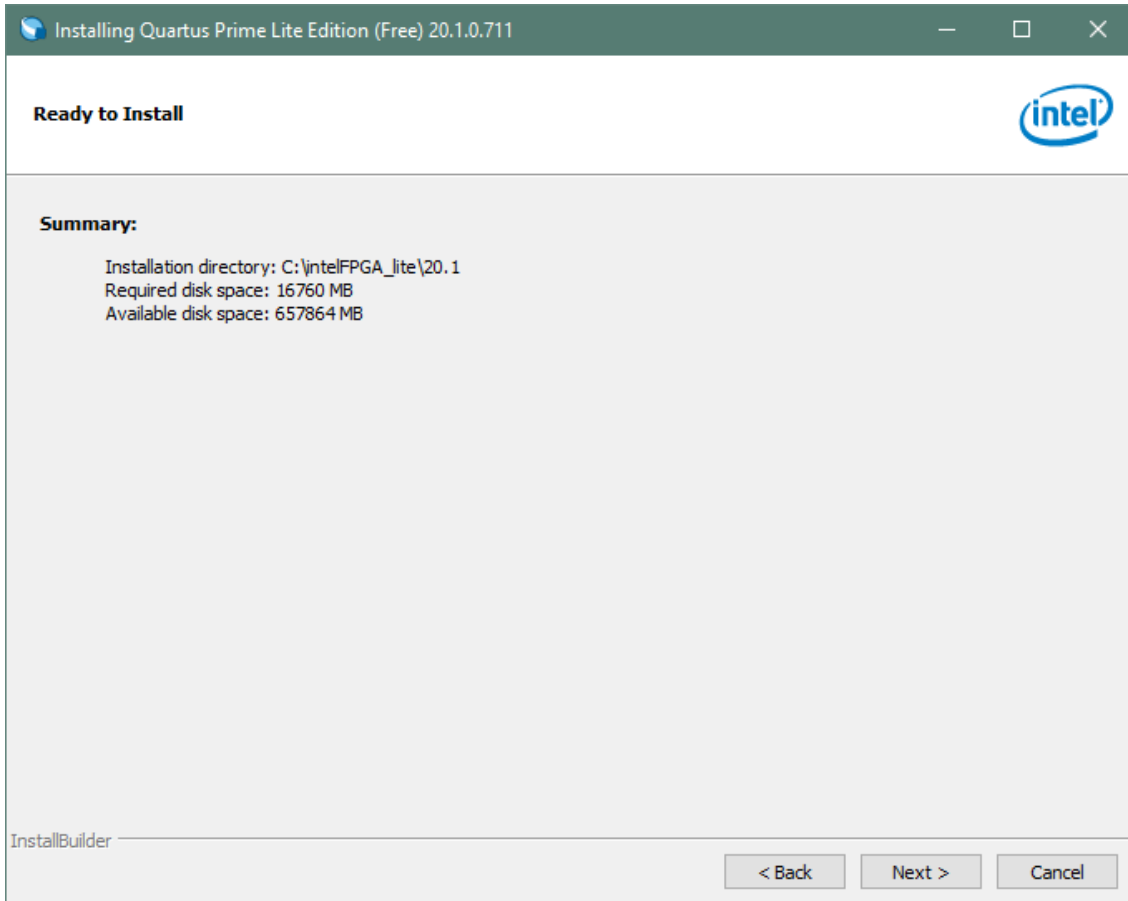
UnoMax CPLD Development System User Manual



Click “Next” to accept the defaults



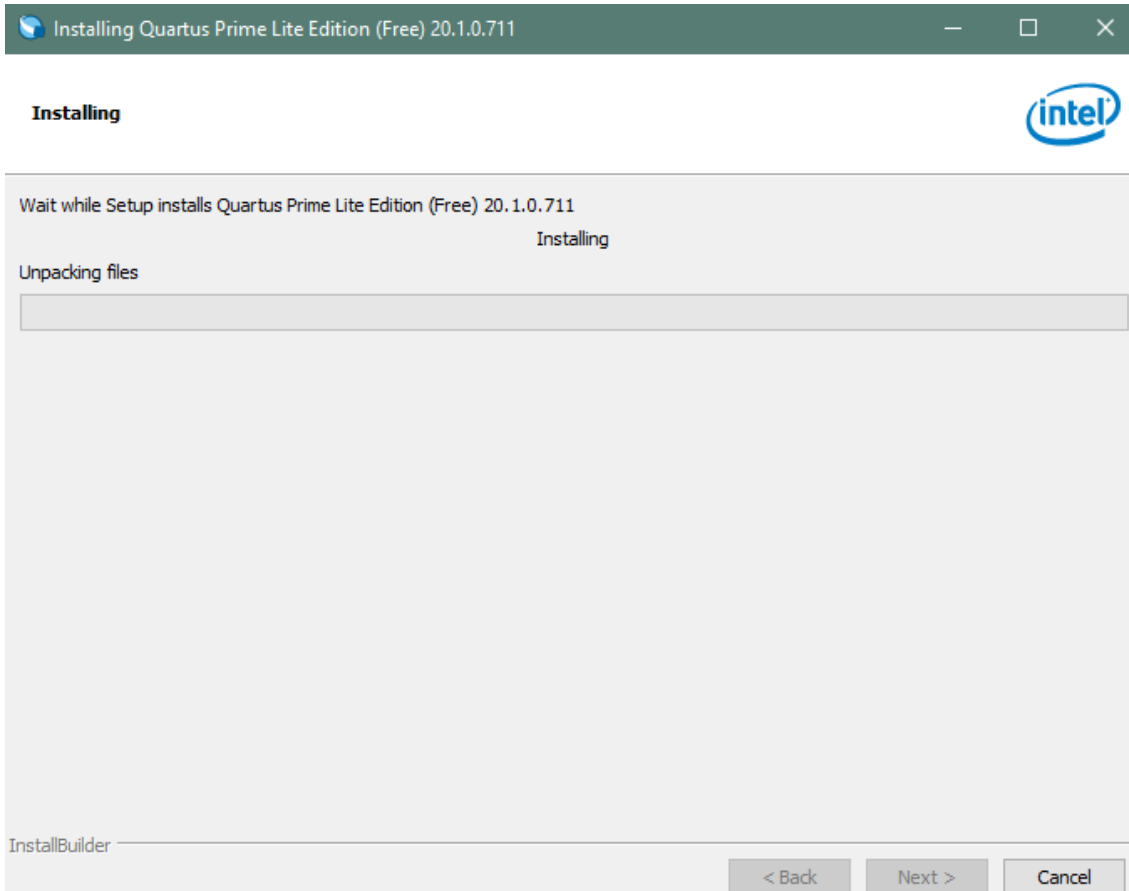
UnoMax CPLD Development System User Manual



Click "Next" to accept the defaults

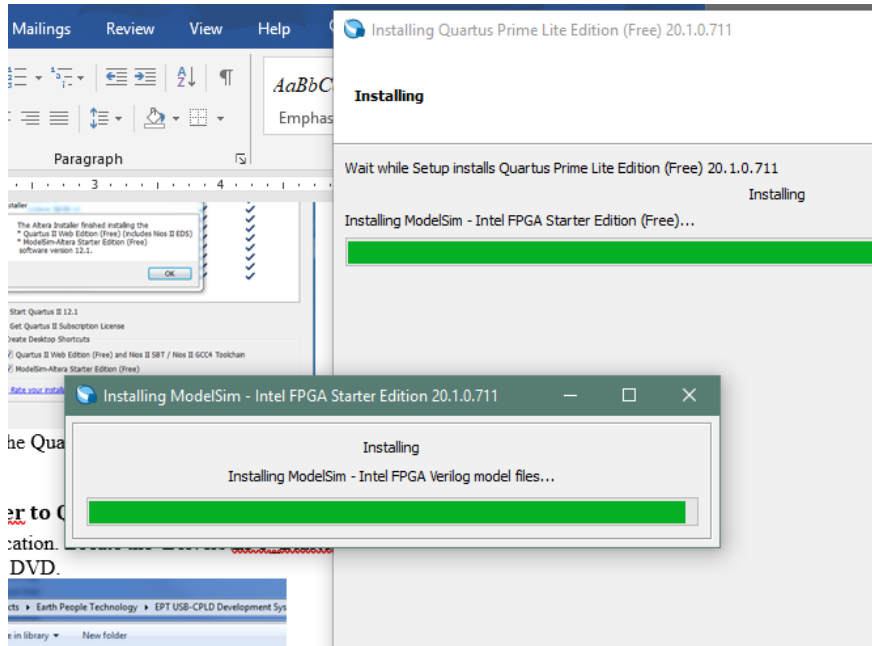


UnoMax CPLD Development System User Manual

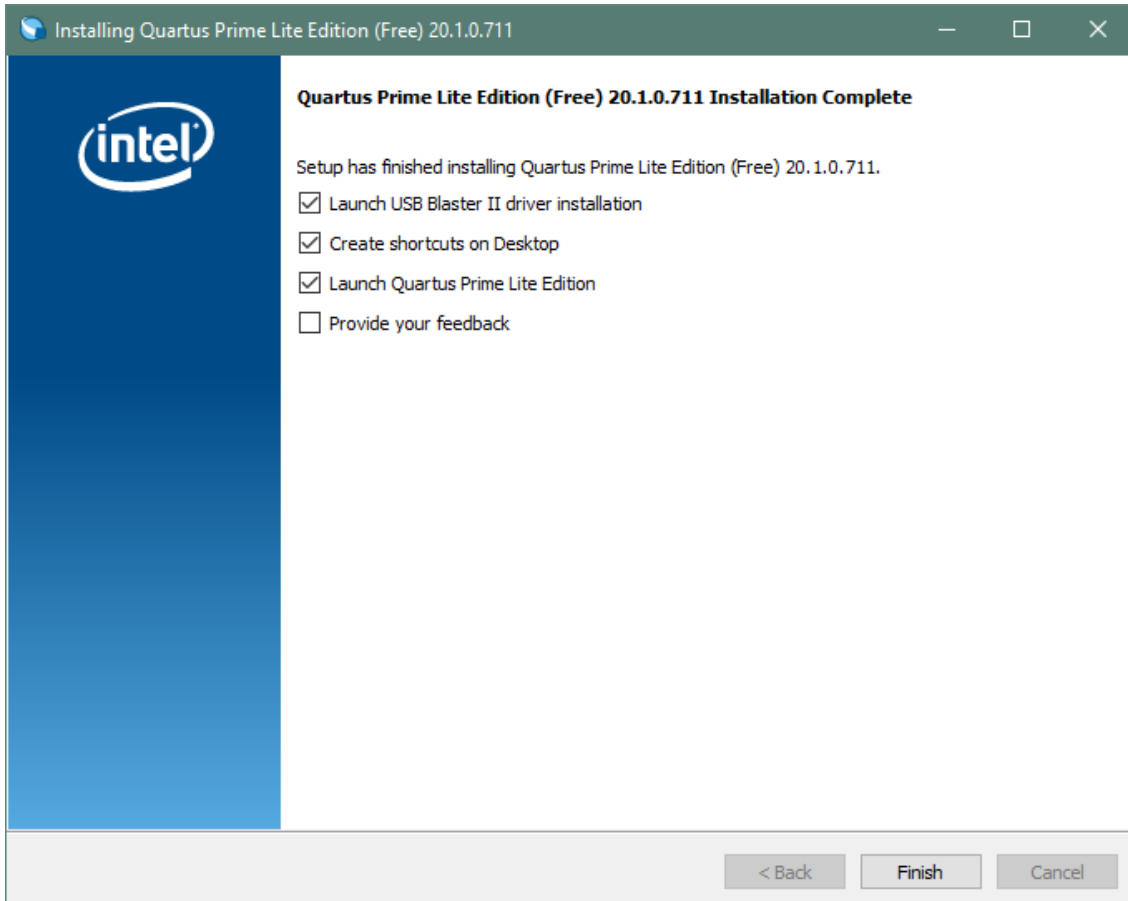


Wait for the installation to complete.

UnoMax CPLD Development System User Manual

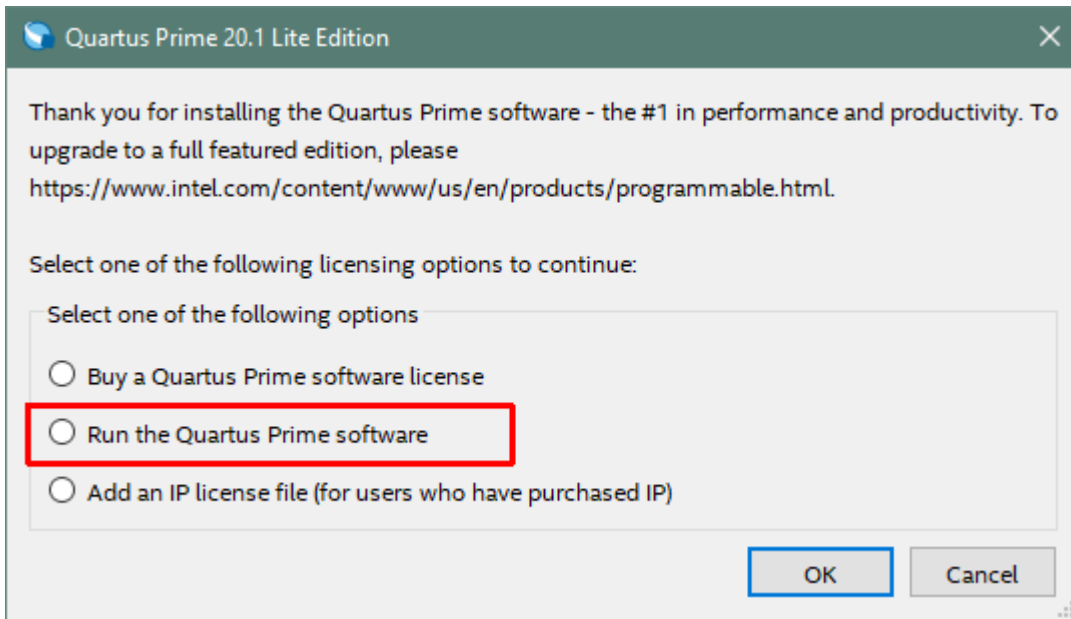


UnoMax CPLD Development System User Manual



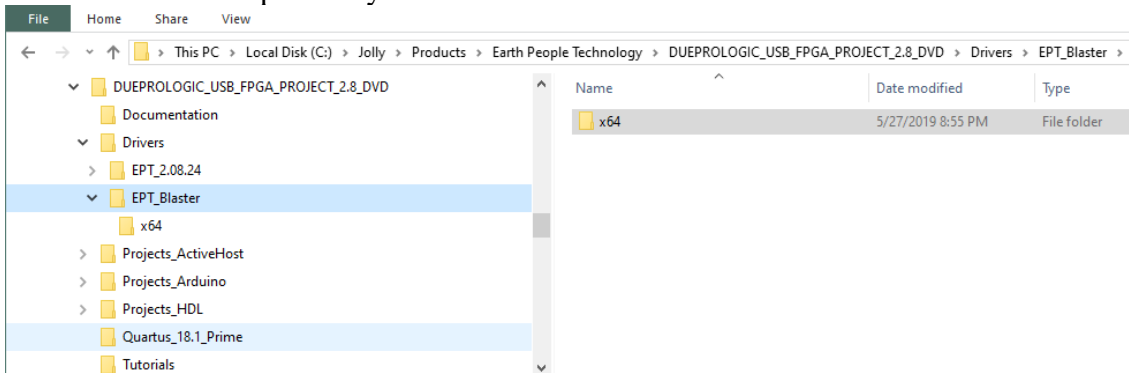
Click "Ok", then click "Finish". The Quartus Prime is now installed and ready to be used.

UnoMax CPLD Development System User Manual



2.2.4 Adding the EPT_Blaster to Quartus Prime

Close out the Quartus Prime application. Locate the \Drivers\EPT_Blaster folder on the EPT FPGA Development System DVD.

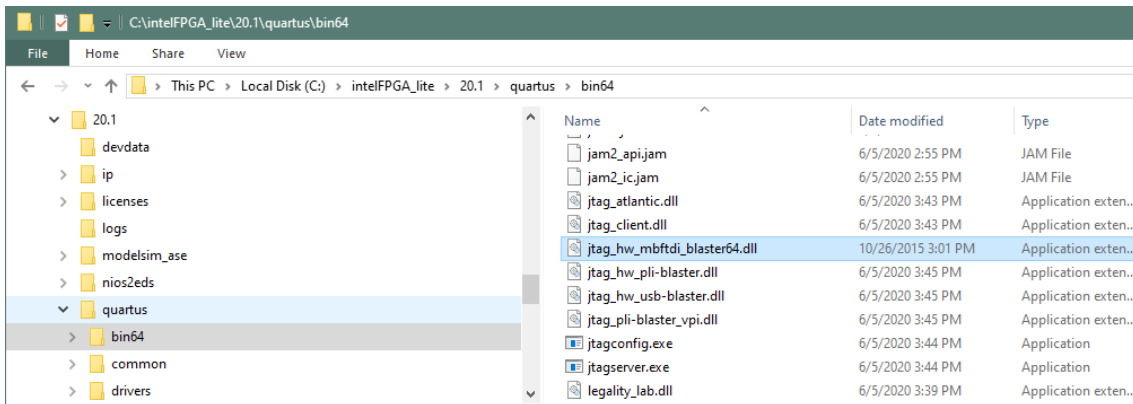


Follow these directions:

1. Open the C:\UNOMAX Development System DVD\Drivers\EPT_Blaster\x64 folder.
2. Select the file “jtag_hw_mbfddi_blaster.dll” and copy it.
3. Browse over to C:\intel\FPGA_lite\xx.x\quartus\bin64.
4. Right click in the folder and select Paste

UnoMax CPLD Development System User Manual

5. Click Ok.
6. Open the Quartus Prime application.



The DLL is installed and the JTAG server should recognize it. Go to the section “Programming the FPGA” of this manual for testing of the programming. If the driver is not found in the Programmer Tool->Hardware Setup box, see the JTAG DLL Insert to Quartus Prime Troubleshooting Guide.

2.3 Active Host Application DLL

Download the latest version of Microsoft Visual C# Express environment from Microsoft. It’s a free download.

<https://visualstudio.microsoft.com/vs/express/>

Go to the website and click on the “+” icon next to the Visual C# Express.



UnoMax CPLD Development System User Manual

A screenshot of a web browser showing the Microsoft Visual Studio Express download page. The browser's address bar displays "https://visualstudio.microsoft.com/vs/express/". The page features a navigation menu with "Visual Studio" selected, and a "Free Visual Studio" button. The main content area has the heading "Visual Studio Express" and a sub-heading "Download Visual Studio Community for a fully-featured and extensible IDE; An updated alternative to Visual Studio Express." Below this is a purple button labeled "Download Community 2019" and a link "Learn more about Visual Studio Community".

Click on the “Express 20xx for Windows Desktop” hypertext.

A screenshot of the Visual Studio selection page. The browser address bar shows "https://visualstudio.microsoft.com/vs/express/". The page has the heading "Help me choose. I am a..." and a sub-heading "Choose from the options below to see what version of Visual Studio is right for you". Below this is a dropdown menu with the text "I am a..." and a downward arrow.

Still want Visual Studio Express?

[Express 2017 for Windows Desktop](#)

Supports building managed and native desktop applications.*

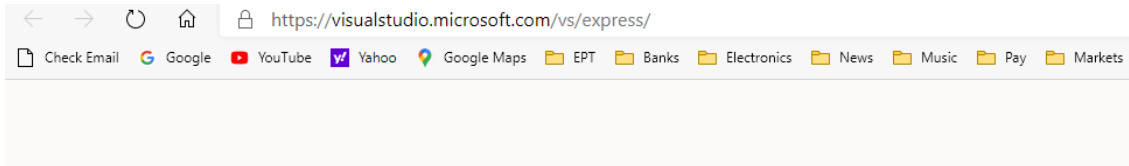
[Express 2015 for Windows Desktop](#)

Supports the creation of desktop applications for Windows.

The download manager file will download the “WDEpress.exe” file.



UnoMax CPLD Development System User Manual



Still want Visual Studio Express?

[Express 2017 for Windows Desktop](#)

Supports building managed and native desktop applications.*

[Express 2015 for Windows Desktop](#)

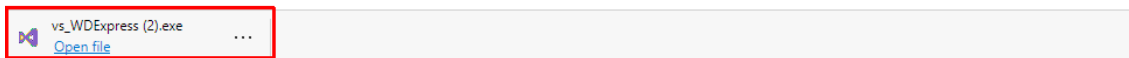
Supports the creation of desktop applications for Windows.

[Express 2015 for Web](#)

Create standards-based, responsive websites, web APIs, or real-time online experiences using ASP.NET.

[Express 2015 for Windows 10](#)

Provides the core tools for building compelling, innovative apps for Universal Windows Platform. Windows is required.



Right click on the WDEpress.exe.

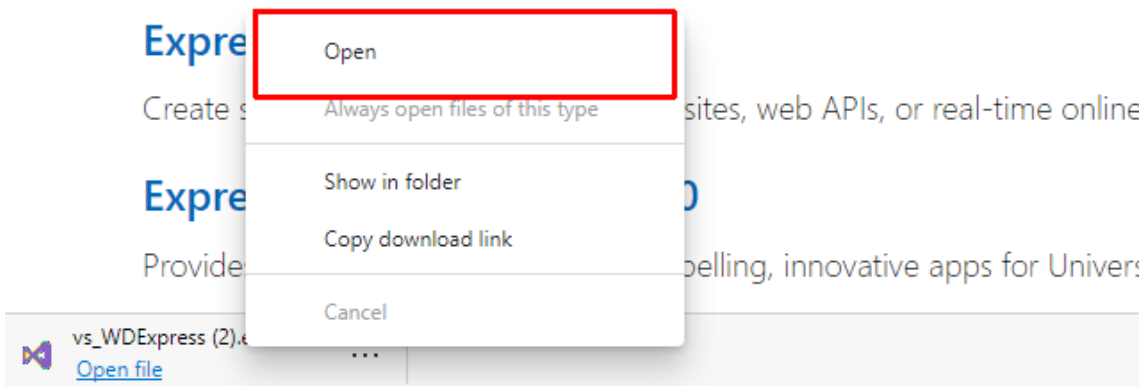
Still want Visual Studio Express?

[Express 2017 for Windows Desktop](#)

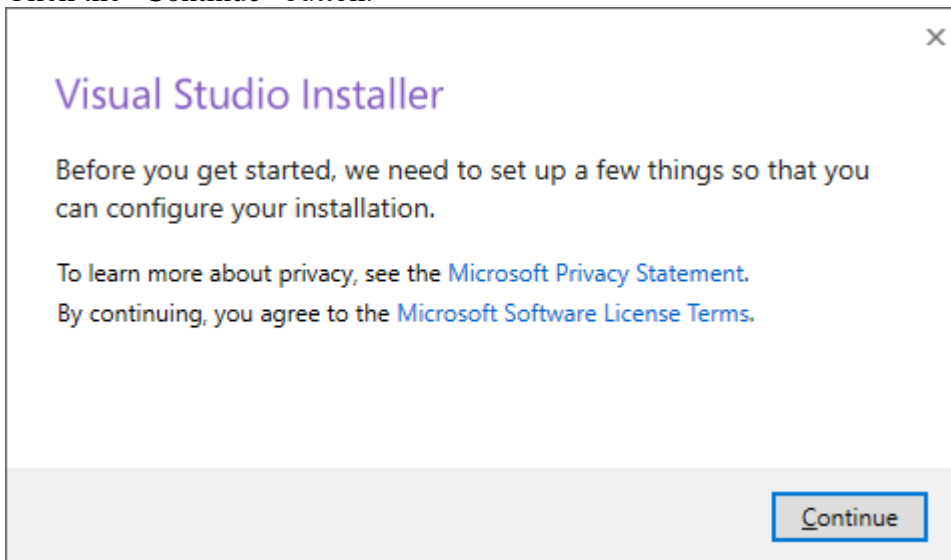
Supports building managed and native desktop applications.*

[Express 2015 for Windows Desktop](#)

Supports the creation of desktop applications for Windows.



Click the “Continue” button.

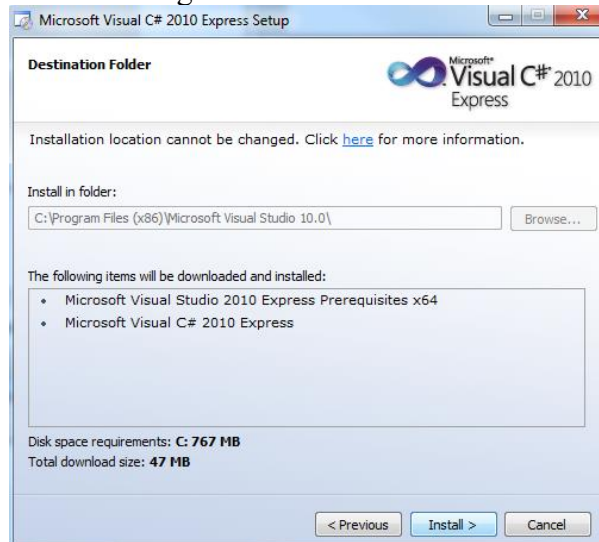


Next, follow the on screen windows and accept the default answers.

UnoMax CPLD Development System User Manual



Click “Next”, accept the license agreement. Click “Next”.



Visual C# 2010 Express will install. This may take up to twenty minutes depending on your internet connection.

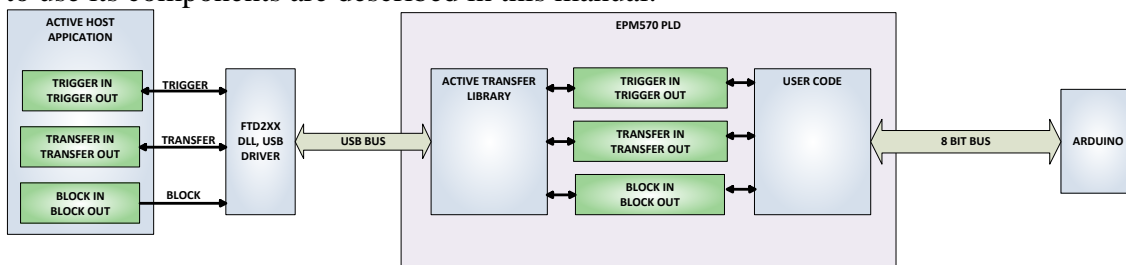
UnoMax CPLD Development System User Manual



The installed successfully window will be displayed when Visual C# Express is ready to use.

3 Active Transfer Library

The Active Transfer Library is an HDL library designed to transfer data to and from the UnoMax via High Speed USB. It is a set of pre-compiled HDL files that the user will add to their project before building it. The description of what the library does and how to use its components are described in this manual.



3.1 EPT Active Transfer System Overview

The Active Transfer System components consist of the following:

- active_serial_library.v
- ft_245_state_machine.v
- endpoint_registers.vqm
- active_trigger.v
- active_transfer.v
- active_block.v

The Active_Serial_Library provides the communication to the USB hardware. While separate Input and Output buses provide bi-directional communications with the plug in modules. See Figure 6 for an overview of the EPT Active_Transfer system.

Figure 6 EPT Active Transfer Library Overview

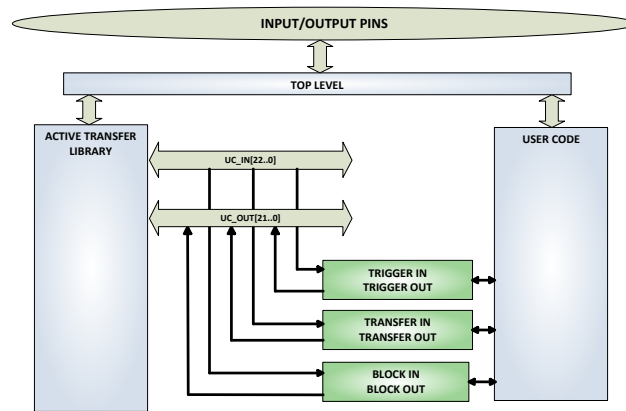


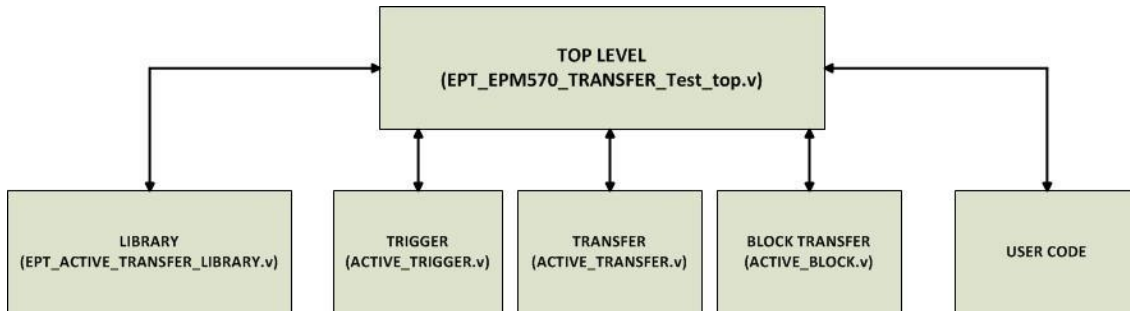
Figure 6 shows how the modules of the EPT Active Transfer Library attach to the overall user project. The EPT Active_Transfer_Library.vqm, Active_Trigger.v, Active_Transfer.v and Active_Block.v modules are instantiated in the top level of the user project. The User_Code.v module is also instantiated in the top level. The Active_Transfer modules communicate with the User_Code through module parameters. Each module is a bi-directional component that facilitates data transfer from PC to CPLD. The user code can send a transfer to the Host, and the Host can send a transfer to the user code. This provides significant control for both data transfers and signaling from the user code to PC. The Triggers are used to send momentary signals that can turn on (or off) functions in user code or PC. The Active Transfer is used to send a single byte. And the Active Block is used to send a block of data. The Active_Transfer and Active_Block modules have addressing built into them. This means the user can declare up to 8 individual instantiations of Active_Transfer or Active_Block, and send/receive data to each module separately.

3.2 Active Transfer Library

The Active Transfer Library contains the command, control, and data transfer mechanism that allows users to quickly build powerful communication schemes in the CPLD. Coupled with the Active Host application on the PC, this tools allows users to focus on creating programmable logic applications and not have to become distracted

UnoMax CPLD Development System User Manual

by USB Host drivers and timing issues. The Active Transfer Library is pre-compiled file that the user will include in the project files.





UnoMax CPLD Development System User Manual

```
1  //#####
2  //#
3  //# Copyright   Earth People Technology Inc. 2023
4  //#
5  //#
6  //# File Name:   EPT_5M57_AP_U2_Top.v
7  //# Author:     R. Jolly
8  //# Date:       January 26, 2023
9  //# Revision:   A
10 //#
11 //# Development: USB Test Tool Interface board
12 //# Application: Altera MAX V CPLD
13 //# Description: This file contains verilog code which will allow access
14 //#               to Active Transfer Library. The MAX V receives its commands via
15 //#               USB.
16 //#
17 //#
18 //#
19 //#####
20 //#
21 //# Revision History:
22 //#   DATE       VERSION   DETAILS
23 //#   1/26/23    A         Created      RJJ
24 //#
25 //#
26 //#
27 //#
28 //#####
29
30 `timescale 1ns/1ps
31
32
33
34 //#####
35 //* Module Declaration
36 //#####
37
38 module EPT_5M57_AP_U2_Top (
39
40
41     input wire          CLK_66MHZ,
42     input wire          RST,
43
44     //UART COMMAND BUS
45     output wire         UART_OUT, //To external
46     input wire          UART_IN, //From external
47
```




UnoMax CPLD Development System User Manual

```
47
48     input wire    [1:0]    LB_SER,        //XIOH -- J10
49     input wire    [5:0]    LB_AD,         //AD  -- J9
50     input wire    [7:0]    LB_IOH,       //XIOH -- J10
51     input wire    [7:0]    LB_IOL,       //XIOL -- J8
52
53
54     //Transceiver Control Signals
55     output wire    TR_DIR_1,
56     output wire    TR_OE_1,
57
58     output wire    TR_DIR_2,
59     output wire    TR_OE_2,
60
61     output wire    TR_DIR_3,
62     output wire    TR_OE_3,
63
64     input wire    SW_USER_1,
65     input wire    SW_USER_2,
66
67     input wire    SLIDE_SWITCH_1,
68     input wire    SLIDE_SWITCH_2,
69
70     output reg    LED_1_GREEN,
71     output reg    LED_1_RED,
72     output reg    LED_1_BLUE,
73
74     output reg    LED_2_GREEN,
75     output reg    LED_2_RED,
76     output reg    LED_2_BLUE,
```

The interface from the library to the user code is two uni directional buses, UC_IN[22:0] and UC_OUT[20:0]. The UC_IN[22:0] bus is an output bus (from the library, input bus to the Active Modules) that is used channel data, address, length and control information to the Active Modules. The UC_OUT[21:0] bus is an input bus (to the library, output bus from the Active Modules) that is used to communicate data, address, length, and control information to the Active Modules.

UnoMax CPLD Development System User Manual

```
556
557
558 //-----
559 // Instantiate the EPT Library
560 //-----
561
562 active_serial_library    ACTIVE_SERIAL_LIBRARY_INST
563 (
564     .aa                (aa) ,
565     .UART_OUT          (UART_OUT) ,
566     .UART_IN           (uart_txd) ,
567
568     .UC_IN             (UC_IN) ,
569     .UC_OUT            (UC_OUT) //,
570
571     // .STATE_OUT      (active_serial_state_out),
572     // .TEST_OUT       (active_transfer_test_out)
573
574 );
575
576 //-----
577 // Instantiate the EPT Library
578 //
579 //
```

The control bus UART_IN and UART_OUT are used to channel data, and control signals to the USB interface chip. These signals are connected directly to input and output pins of the CPLD.

3.2.1 Active Trigger EndTerm

The Active Trigger has eight individual self resetting, active high, signals. These signals are used to send a momentary turn on/off command to Host/User code. The Active Trigger is not addressable so the module will be instantiated only once in the top level.



UnoMax CPLD Development System User Manual

```
743 wire [22*3-1:0] uc_out_m;  
744 eptWireOR # (.N(3)) wireOR (UC_OUT, uc_out_m);  
745 active_trigger ACTIVE_TRIGGER_INST  
746 (   
747     .uc_clk (CLK_66),  
748     .uc_reset (RST),  
749     .uc_in (UC_IN),  
750     .uc_out (uc_out_m[ 0*22 +: 22 ]),  
751  
752     .trigger_to_host (trigger_to_host),  
753     .trigger_to_device (trigger_in_byte)  
754 );  
755  
756
```

To send a trigger, decide which bit (or multiple bits) of the eight bits you want to send the trigger on. Then, set that bit (or bits) high. The Active Transfer Library will send a high on that trigger bit for one clock cycle (66 MHz), then reset itself to zero. The bit can stay high on the user code and does not need to be reset to zero. However, if the user sends another trigger using the trigger byte, then any bit that is set high will cause a trigger to occur on the Host side.

UnoMax CPLD Development System User Manual

```

277 //-----
278 // Detect Trigger Out to Host
279 //-----
280 always @(TRIGGER_OUT or trigger_in_reset or reset)
281 begin
282     if(!reset)
283         trigger_to_host = 8'h0;
284     else if (trigger_in_reset)
285         trigger_to_host = 8'h0;
286     else if (TRIGGER_OUT > 8'h0)
287         trigger_to_host = TRIGGER_OUT;
288 end
289
290 //-----
291 // Reset Trigger Out to Host
292 //-----
293 always @(posedge CLK_66 or negedge reset)
294 begin
295     if(!reset)
296     begin
297         trigger_in_reset <= 0;
298     end
299     else
300     begin
301         if (trigger_to_host > 0)
302             trigger_in_reset <= 1'b1;
303         else
304             trigger_in_reset <= 0;
305     end
306 end

```

So, care should be used if the user code uses byte masks to send triggers. It is best to set only the trigger bits needed for a given time when sending triggers.

The user code must be setup to receive triggers from the Host. This can be done by using an asynchronous always block. Whenever a change occurs on a particular trigger bit (or bits), a conditional branch can detect if the trigger bit is for that block of code. Then, execute some code based on that trigger.



UnoMax CPLD Development System User Manual

```
308 //-----  
309 // Detect Trigger In  
310 //-----  
311 always @(trigger_in_byte or trigger_in_reset or reset)  
312 begin  
313     if(!reset)  
314     begin  
315         trigger_in_detect = 1'b0;  
316     end  
317     else if (trigger_in_reset)  
318     begin  
319         trigger_in_detect = 1'b0;  
320     end  
321     else if (trigger_in_byte > 8'h0)  
322     begin  
323         trigger_in_detect = 1'b1;  
324     end  
325 end  
326  
327 //-----  
328 // Store the value of Trigger In  
329 //-----  
330 always @(posedge CLK_66 or negedge reset)  
331 begin  
332     if(!reset)  
333     begin  
334         trigger_in_store <= 8'h0f;  
335         trigger_in_reg <= 1'b0;  
336         trigger_in_reset <= 1'b0;  
337     end  
338     else if (trigger_in_detect & !trigger_in_reg)  
339     begin  
340         if(trigger_in_byte != 0)  
341             trigger_in_store[7:0] <= trigger_in_byte[7:0];  
342             trigger_in_reg <= 1'b1;  
343     end  
344     else if (trigger_in_reg)  
345     begin  
346         trigger_in_reg <= 1'b0;  
347         trigger_in_reset <= 1'b1;  
348     end  
349     else if (!trigger_in_detect)  
350     begin  
351         trigger_in_reg <= 1'b0;  
352         trigger_in_reset <= 1'b0;  
353     end
```

3.2.2 Active Transfer EndTerm

The Active Transfer module is used to send or receive a byte to/from the Host. This is useful when the user's microcontroller needs to send a byte from a measurement to the Host for display or processing. The Active Transfer module is addressable, so up to eight individual modules can be instantiated and separately addressed.

```

757 active_transfer          ACTIVE_TRANSFER_INST
758 (
759     .uc_clk                (CLK_66),
760     .uc_reset              (reset),
761     .uc_in                 (UC_IN),
762     .uc_out                (uc_out_m[ 1*22 +: 22 ]),
763
764     .start_transfer        (transfer_out_reg),
765     .transfer_received     (transfer_in_received),
766
767     .uc_addr               (3'h2),
768
769     .transfer_to_host      (transfer_out_byte),
770     .transfer_to_device    (transfer_in_byte)
771 );
772
  
```

To send a byte to the Host, select the appropriate address that corresponds to an address on Host side. Place the byte in the “transfer_to_host” parameter, then strobe the “start_transfer” bit. Setting the “start_transfer” bit to high will send one byte from the “transfer_to_host” byte to the Host on the next clock high signal (66 MHz). The “start_transfer” bit can stay high for the duration of the operation of the device, the Active Transfer module will not send another byte. In order to send another byte, the user must cycle the “start_transfer” bit to low for a minimum of one clock cycle (66 MHz). After the “start_transfer” bit has been cycled low, the rising edge of the bit will cause the byte on the “transfer_to_host” parameter to transfer to the host.

UnoMax CPLD Development System User Manual

```

181 //-----
182 // Transfer byte to Device
183 //-----
184 always @(TRANSFER_OUT_EN or reset)
185 begin
186     if(!reset)
187     begin
188         transfer_out_detect = 1'b0;
189     end
190     else
191     begin
192         if(transfer_to_device_reset)
193             transfer_out_detect = 1'b0;
194         else if(TRANSFER_OUT_EN)
195         begin
196             transfer_out_byte = TRANSFER_OUT_BYTE;
197             transfer_out_detect = 1'b1;
198         end
199     end
200 end
201
202 //-----
203 // Reset transfer_to_device_reset
204 //-----
205 always @(posedge CLK_66 or negedge reset)
206 begin
207     if (!reset)
208     begin
209         transfer_to_device_reset <= 1'b0;
210     end
211     else
212     begin
213         if(transfer_out_detect)
214             transfer_to_device_reset <= 1'b1;
215         else
216             transfer_to_device_reset <= 1'b0;
217     end
218 end

```

UnoMax CPLD Development System User Manual

To receive a byte, the Active Host will send a byte using it's dll. The user code must monitor the transfer_received port. The transfer_received port will assert high for one clock cycle (66 MHz) when a byte is ready for reading on the transfer_to_device port. User code should use an asynchronous always block to detect when the transfer_received port is asserted. Upon assertion, the user code should read the byte from the transfer_to_device port into a local register.

```

220 //-----
221 // Transfer to Host
222 //-----
223 always @(posedge CLK_66 or negedge reset)
224 begin
225     if (!reset)
226     begin
227         transfer_out <= 1'b0;
228         transfer_out_reg <= 1'b0;
229         transfer_out_byte <= 8'h0;
230     end
231     else
232     begin
233         if(start_transfer_byte & !transfer_out)
234         begin
235             transfer_out_byte <= TRANSFER_HOST_BYTE;
236             transfer_out_reg <= 1'b1;
237             transfer_out <= 1'b1;
238         end
239         else if(start_transfer_byte & transfer_out)
240         begin
241             transfer_out_reg <= 1'b0;
242             transfer_out <= 1'b1;
243         end
244         else if(!start_transfer_byte & transfer_out)
245         begin
246             transfer_out_reg <= 1'b0;
247             transfer_out <= 1'b0;
248         end
249     end
250 end

```

3.2.3 Active Block EndTerm

The Active Block module is designed to transfer blocks of data between Host and User Code and vice versa. This allows buffers of data to be transferred with a minimal amount of code. The Active Block module is addressable, so up to eight individual



UnoMax CPLD Development System User Manual

modules can be instantiated and separately addressed. The length of the block to be transferred must also be specified in the `uc_length` port.

```
811 active_block          BLOCK_TRANSFER_INST
812 (
813     .uc_clk            (CLK_66),
814     .uc_reset         (RST),
815     .uc_in            (UC_IN),
816     .uc_out          (uc_out_m[ 2*22 +: 22 ]),
817
818     .start_transfer   (block_out_reg),
819     .transfer_received (block_in_rcv),
820
821     .transfer_ready   (block_byte_ready),
822
823     .uc_addr          (3'h4),
824     .uc_length        (BLOCK_COUNT_8),
825
826     .transfer_to_host (block_out_byte),
827     .transfer_to_device (block_in_data),
828
829     .STATE_OUT        (block_state_out),
830     .TEST_BUS         (block_out_test_bus)
831 );
832
833
```

To send a block, it's best to have buffer filled in a previous transaction, Then assert the `start_transfer` bit. This method is opposed to collecting and processing data bytes after the `start_transfer` bit has been asserted and data is being sent to the Host.

Once the buffer to send is filled with the requisite amount of data, the address and buffer length should be written to the `uc_addr` and `uc_length` ports. Set the `start_transfer` bit high, the user code should monitor the `transfer_ready` port. At the rising edge of the `transfer_ready` port, the byte at `transfer_to_host` port is transferred to the USB chip. Once this occurs, the user code should copy the next byte in the buffer to `transfer_to_host` port. On the next rising edge of `transfer-ready`, the byte at `transfer_to_host` will be transferred to the USB chip. This process continues until the number of bytes described by the `uc_length` have been transferred into the USB chip.

UnoMax CPLD Development System User Manual

```

542 //-----
543 // Registers to start Block Transfer Out
544 //-----
545 always @(posedge CLK_66 or negedge RST)
546 begin
547     if(!RST)
548     begin
549         block_out_reg <= 1'b0;
550         start_block_transfer_reg <= 1'b0;
551     end
552     else
553     begin
554         if(start_block_transfer & !start_block_transfer_reg)
555             start_block_transfer_reg <= 1'b1;
556         else if(start_block_transfer_reg & !block_out_reg)
557         begin
558             block_out_reg <= 1'b1;
559         end
560         else if(block_out_counter >= BLOCK_COUNT_8)
561         begin
562             block_out_reg <= 1'b0;
563             start_block_transfer_reg <= 1'b0;
564         end
565     end
566 end
567
568 //-----
569 // Data for Block Transfer Out
570 //-----
571 always @(posedge CLK_66 or negedge RST)
572 begin
573     if(!RST)
574     begin
575         block_out_counter <= 0;
576     end
577     else
578     begin
579         if(block_byte_ready)
580         begin
581             block_out_counter <= block_out_counter + 1'd1;
582         end
583         else if(block_out_counter >= BLOCK_COUNT_8 )
584         begin
585             block_out_counter <= 0;
586         end
587     end
588 end

```

To receive a buffer from the Host, the user code should monitor the `transfer_received` port for assertion. When the bit is asserted, the next rising edge of `transfer_ready` will indicate that the byte at `transfer_to_device` is ready for the user code to read.

[Add code snippet showing Active Block Module bytes received by the user code]

3.3 Timing Diagram for Active Transfer EndTerms

The Active Transfer Library uses the 66 MHz clock to organize the transfers to Host and transfer to Device. The timing of the transfers depends on this clock and the specifications of the USB chip. Users should use the timing diagrams to ensure proper operation of user code in data transfer.

3.3.1 Active Trigger EndTerm Timing

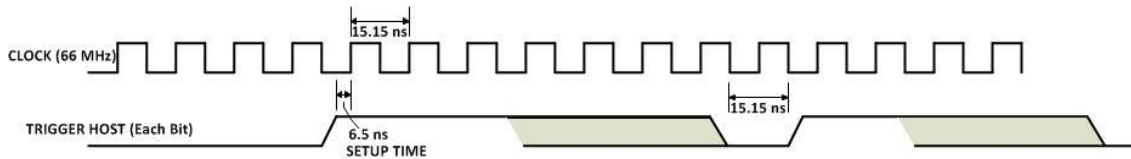


Figure xx Active Trigger to Host Timing

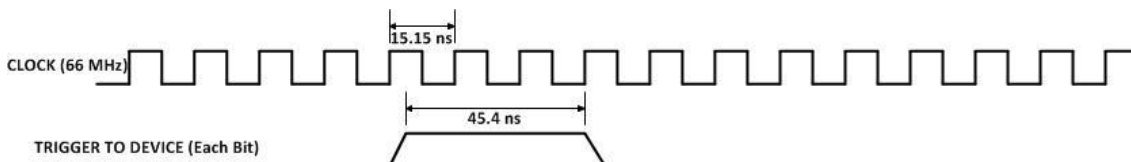


Figure xx Active Trigger to Device Timing

3.3.2 Active Transfer EndTerm Timing

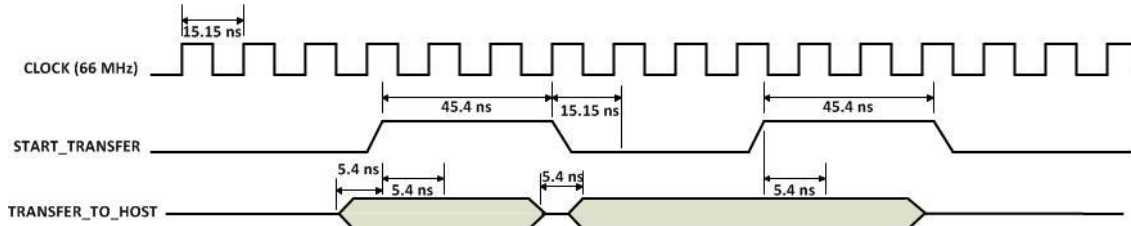


Figure xx Active Transfer To Host Timing

UnoMax CPLD Development System User Manual

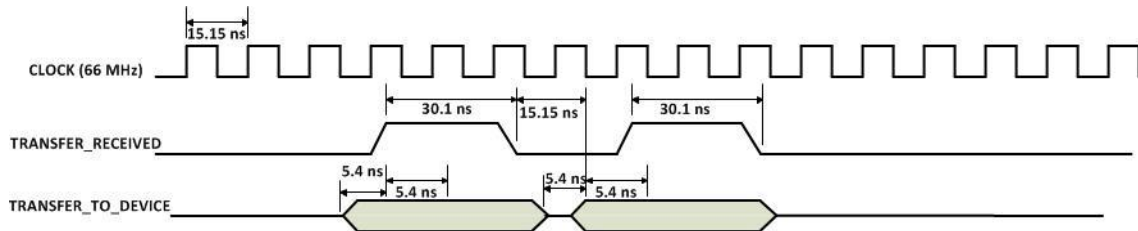


Figure xx Active Transfer To Device Timing

3.3.3 Active Block EndTerm Timing

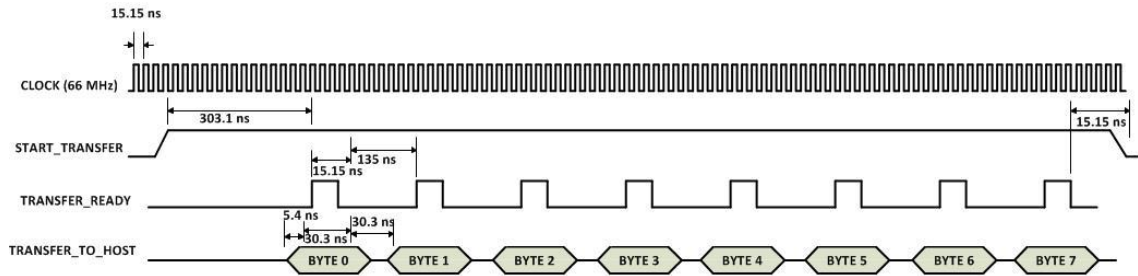


Figure xx Active Block To Host Timing

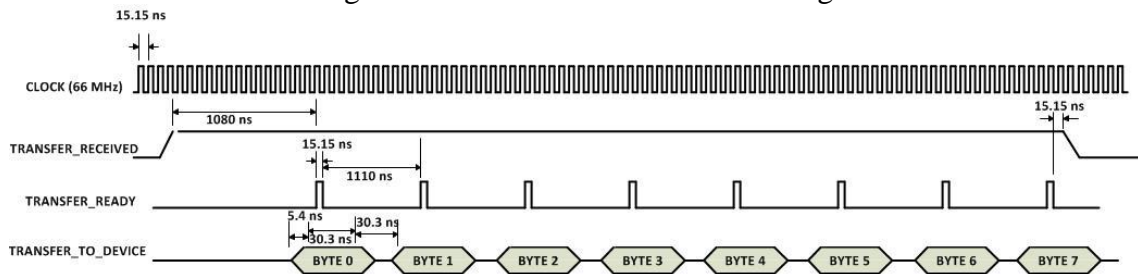
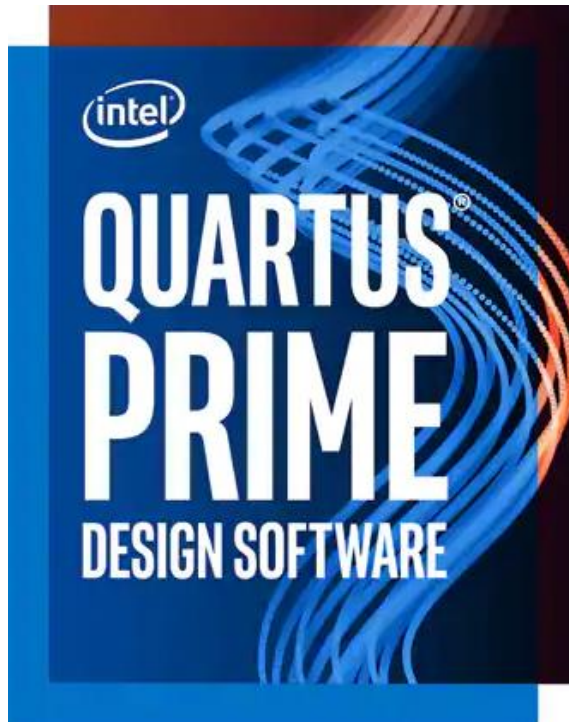


Figure xx Active Block To Device Timing

4 Compiling, Synthesizing, and Programming CPLD

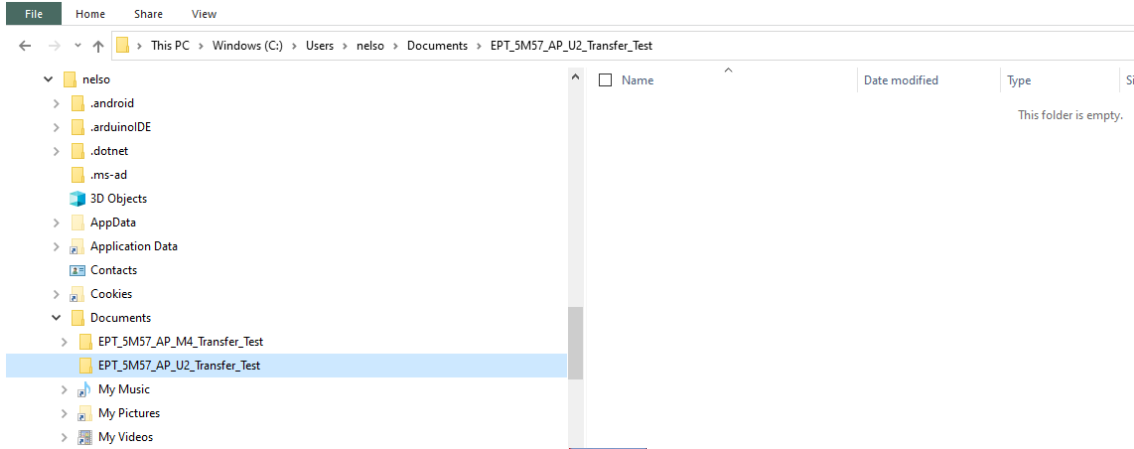


The CPLD on the UnoMax can be programmed with the Active Transfer Library and custom HDL code created by the user. Programming the CPLD requires the use of the Quartus Prime Lite software and a standard USB cable. There are no extra parts to buy, just plug in the USB cable. Once the user HDL code is written according to the syntax rules of the language (Verilog and VHDL) it can be compiled and synthesized using the Quartus Prime Lite software. This manual will not focus on HDL coding or proper coding techniques, instead it will use the example code to compile, synthesize and program the CPLD.

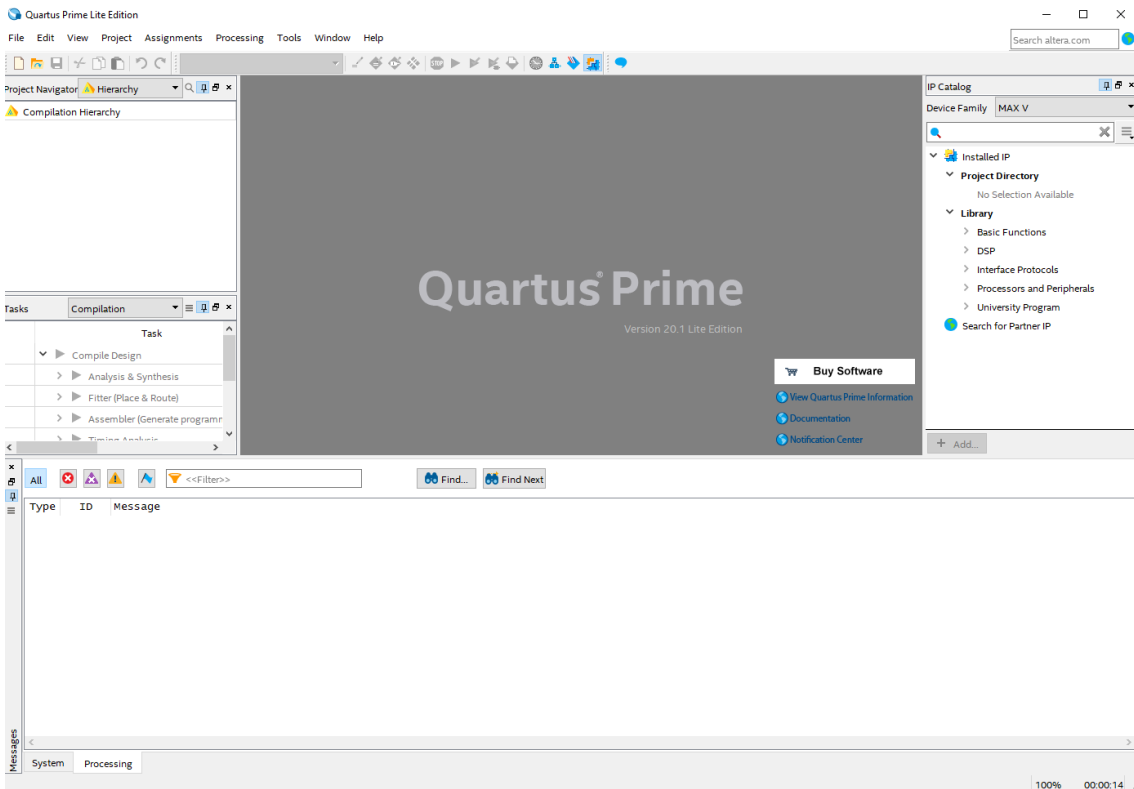
4.1 Setting up the Project and Compiling

Once the HDL code (Verilog or VHDL) is written and verified using a simulator, a project can be created using Quartus Prime Lite. Writing the HDL code and simulating it will be covered in later sections. Bring up Quartus Prime Lite, then use Windows Explorer to browse to C:\Users\nelso\Documents\EPT_5M57_AP_U2_Transfer_Test to create a new directory called: "EPT_Transfer_Test".

UnoMax CPLD Development System User Manual

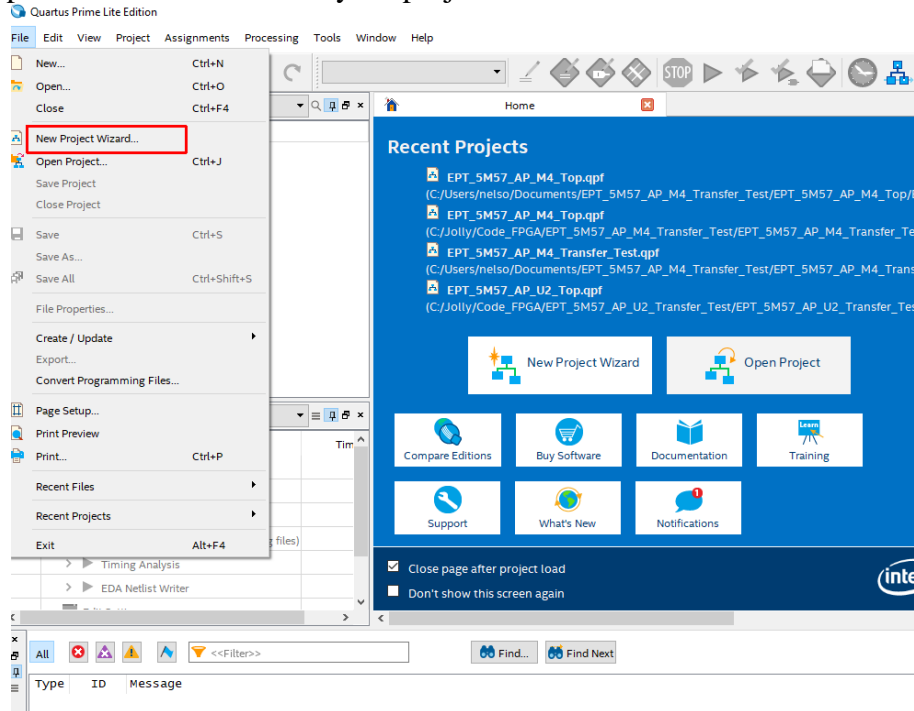


Open Quartus Prime by clicking on the icon .



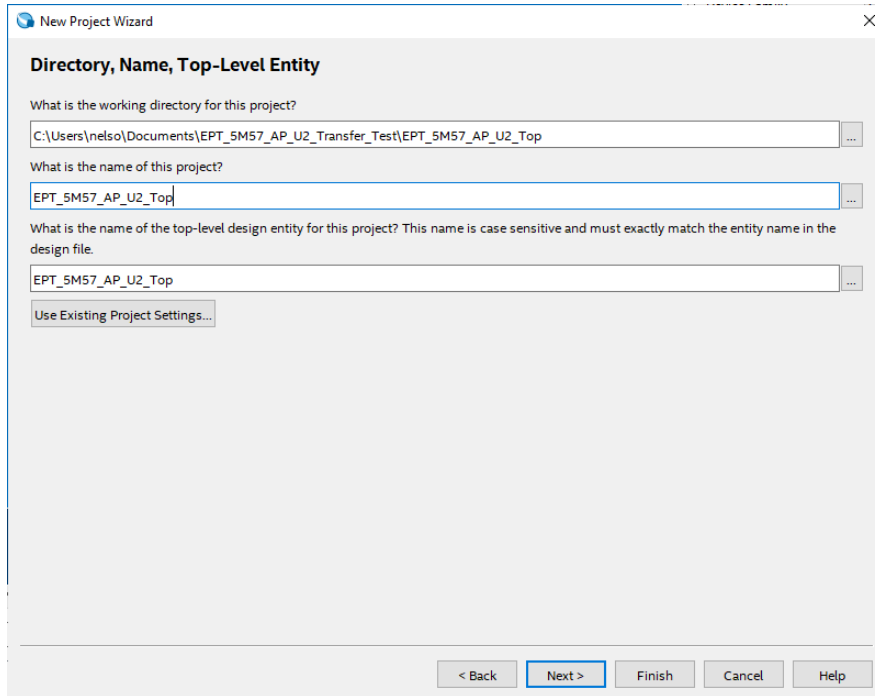
UnoMax CPLD Development System User Manual

Under Quartus, Select File->New Project Wizard. The Wizard will walk you through setting up files and directories for your project.



At the Top-Level Entity page, browse to the C:\Users\<<your name>\Documents\EPT_5M57_AP_U2_Transfer_Test\EPT_5M57_AP_U2_Top directory to store your project. Type in a name for your project “EPT_5M57_AP_U2_Top”.

UnoMax CPLD Development System User Manual



New Project Wizard

Directory, Name, Top-Level Entity

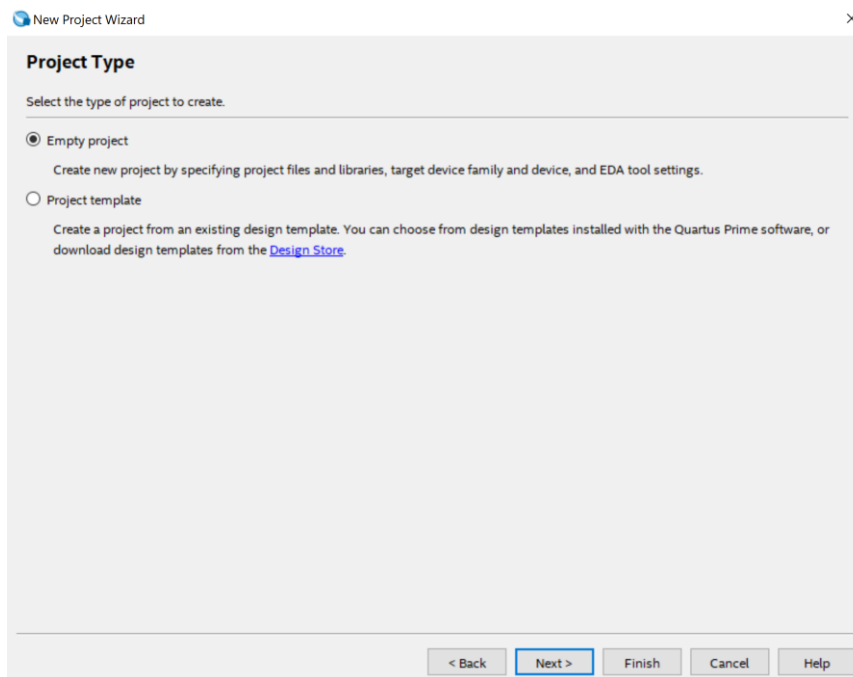
What is the working directory for this project?
C:\Users\nelso\Documents\EPT_5M57_AP_U2_Transfer_Test\EPT_5M57_AP_U2_Top

What is the name of this project?
EPT_5M57_AP_U2_Top

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.
EPT_5M57_AP_U2_Top

Use Existing Project Settings...

< Back Next > Finish Cancel Help



New Project Wizard

Project Type

Select the type of project to create.

Empty project
Create new project by specifying project files and libraries, target device family and device, and EDA tool settings.

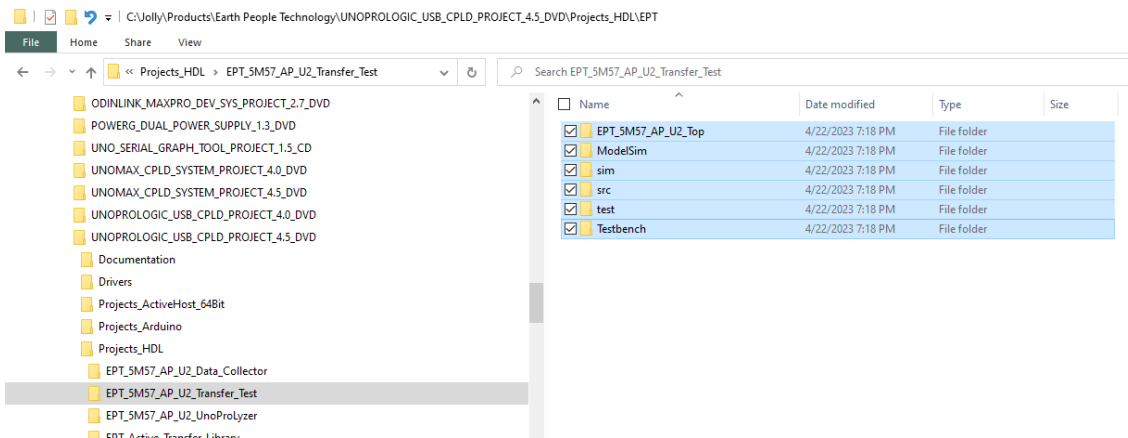
Project template
Create a project from an existing design template. You can choose from design templates installed with the Quartus Prime software, or download design templates from the [Design Store](#).

< Back Next > Finish Cancel Help

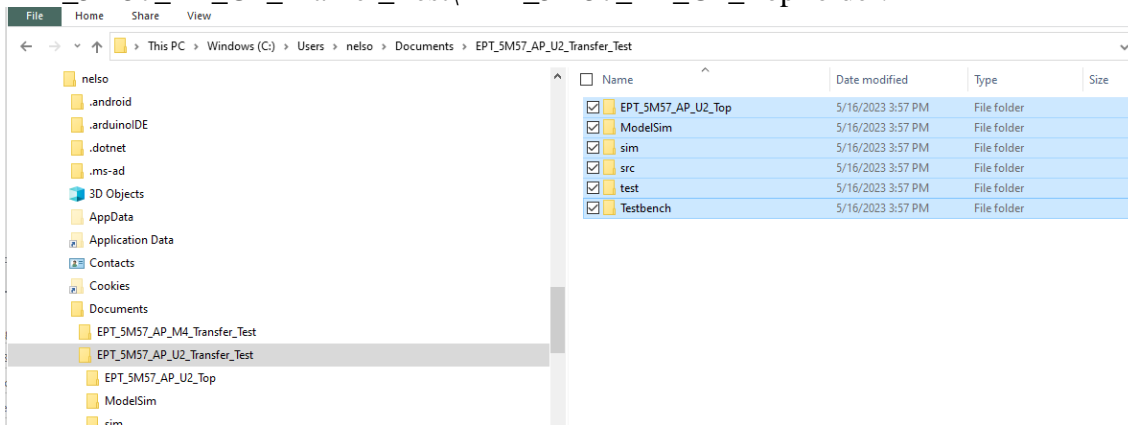


UnoMax CPLD Development System User Manual

Select Next. At the Add Files window: Browse to the \Projects_HDL\EPT_Transfer_Test\src folder of the UnoMax Development System DVD. Copy the files from the following directory.

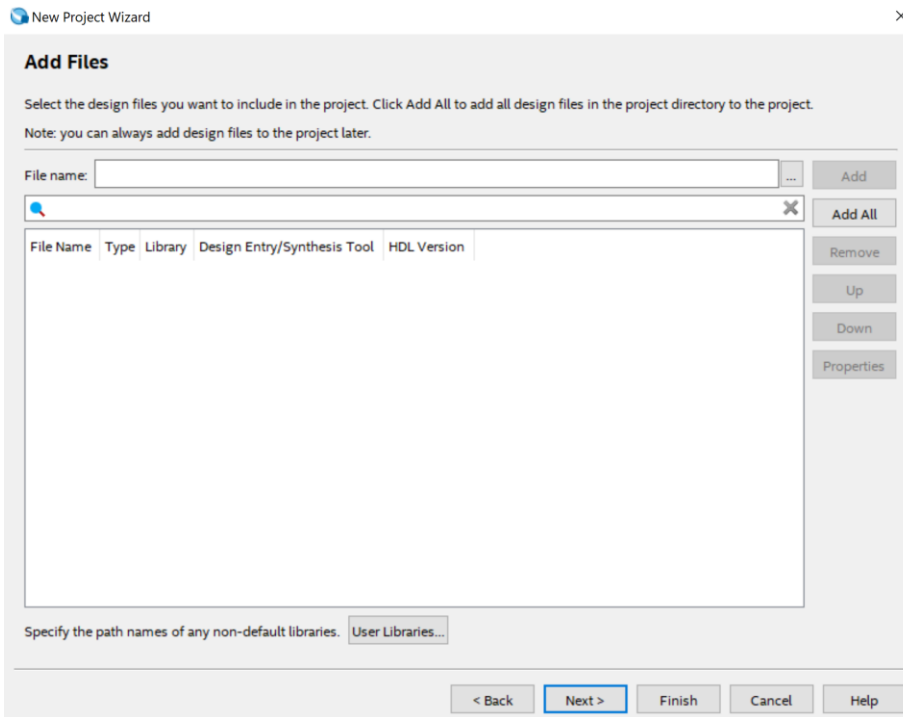


Then paste them in the C:/users/<your name>/Documents/EPT_5M57_AP_U2_Tranfer_Test\EPT_5M57_AP_U2_Top folder:

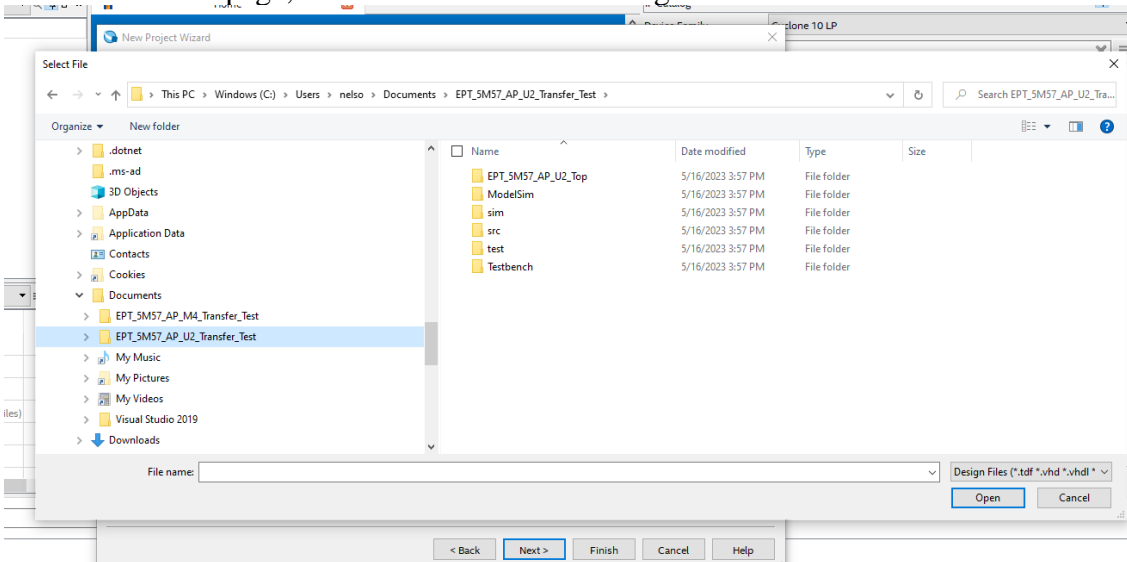


Then go to the “Add Files” page:

UnoMax CPLD Development System User Manual



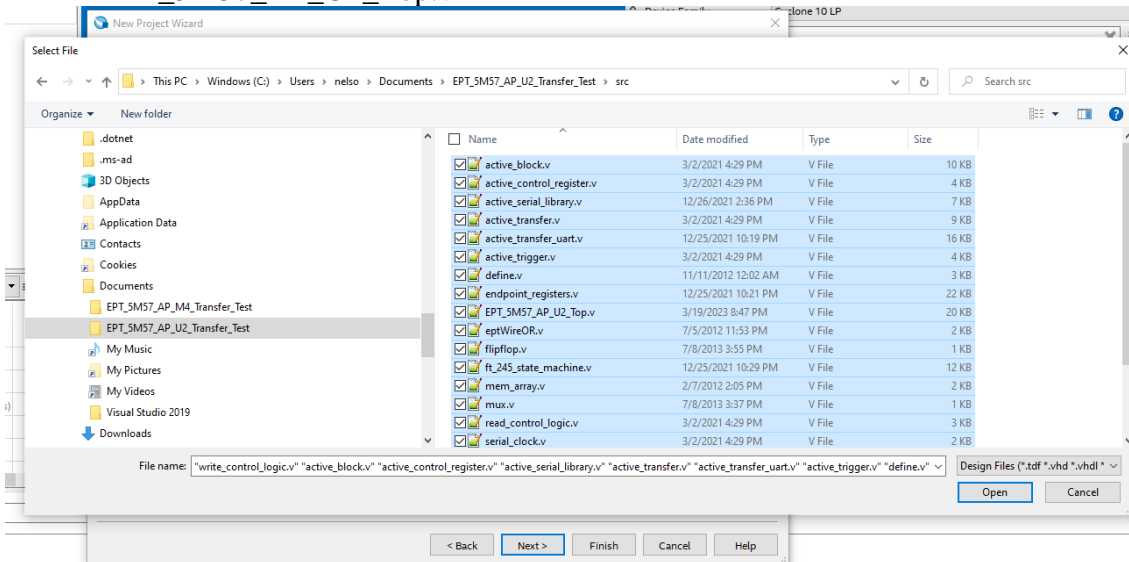
On the Resource page, click “Select File” and navigate to the folder:



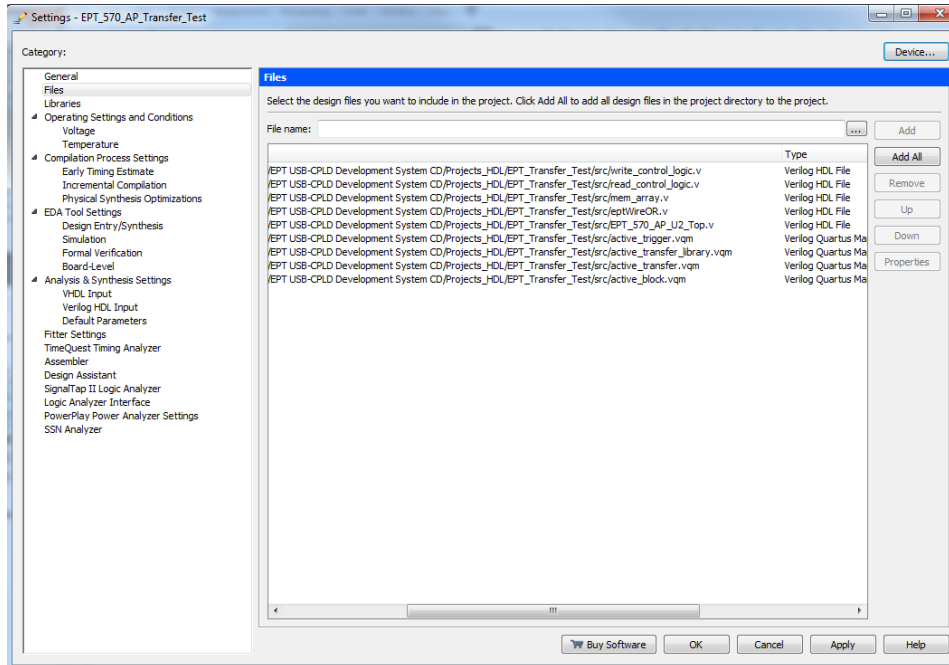
- Active_block.v
- Active_transfer.v
- Active_trigger.v

UnoMax CPLD Development System User Manual

- Active_Serial_library.v
- Endpoint_registers.v
- FT_245_state_machine.v
- eptWireOr.v
- mem_array.v
- read_control_logic.v
- Serial_clock.v
- Uart_receiver.v
- Uart_transmitter.v
- write_control_logic.v
- EPT_5M57_AP_U2_Top.v



UnoMax CPLD Development System User Manual



Select Next, at the Device Family group, select MAX V for Family. In the Available Devices group, browse down to 5M570ZT100C5 for Name.

UnoMax CPLD Development System User Manual

New Project Wizard

Family, Device & Board Settings

Device Board

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

To determine the version of the Quartus Prime software in which your target device is supported, refer to the [Device Support List](#) webpage.

Device family

Family: MAX V
Device: All

Target device

Auto device selected by the Fitter
 Specific device selected in 'Available devices' list
 Other: n/a

Show in 'Available devices' list

Package: Any
Pin count: Any
Core speed grade: Any
Name filter:
 Show advanced devices

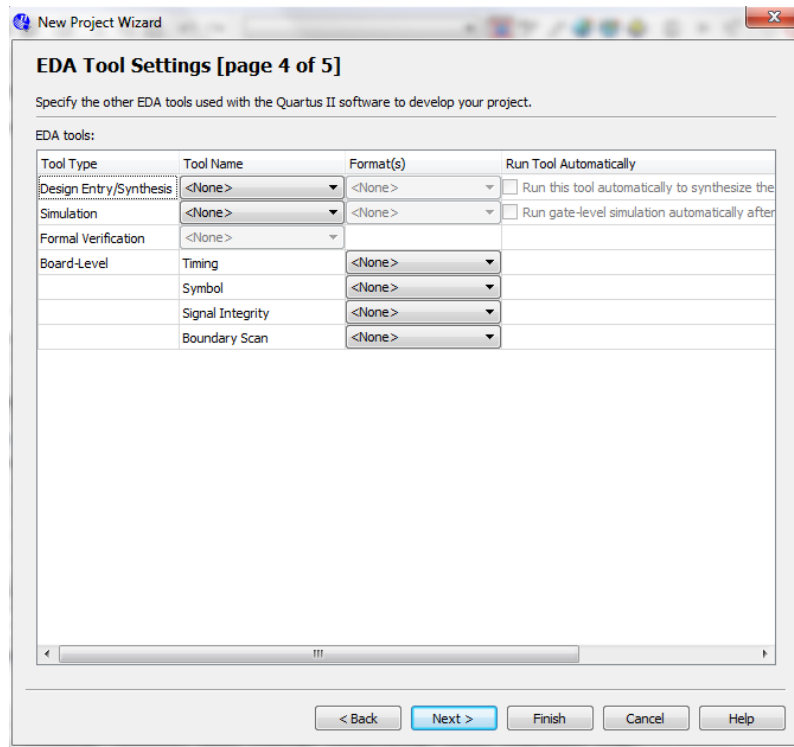
Available devices:

Name	Core Voltage	LEs	UFM blocks
5M570ZM100C4	1.8V	570	1
5M570ZM100C5	1.8V	570	1
5M570ZM100I5	1.8V	570	1
5M570ZT100A5	1.8V	570	1
5M570ZT100C4	1.8V	570	1
5M570ZT100C5	1.8V	570	1
5M570ZT100I5	1.8V	570	1
5M570ZT144C4	1.8V	570	1
5M570ZT144C5	1.8V	570	1
5M570ZT144I5	1.8V	570	1
5M1270ZF256A5	1.8V	1270	1
5M1270ZF256C4	1.8V	1270	1
5M1270ZF256C5	1.8V	1270	1

< Back Next > Finish Cancel Help

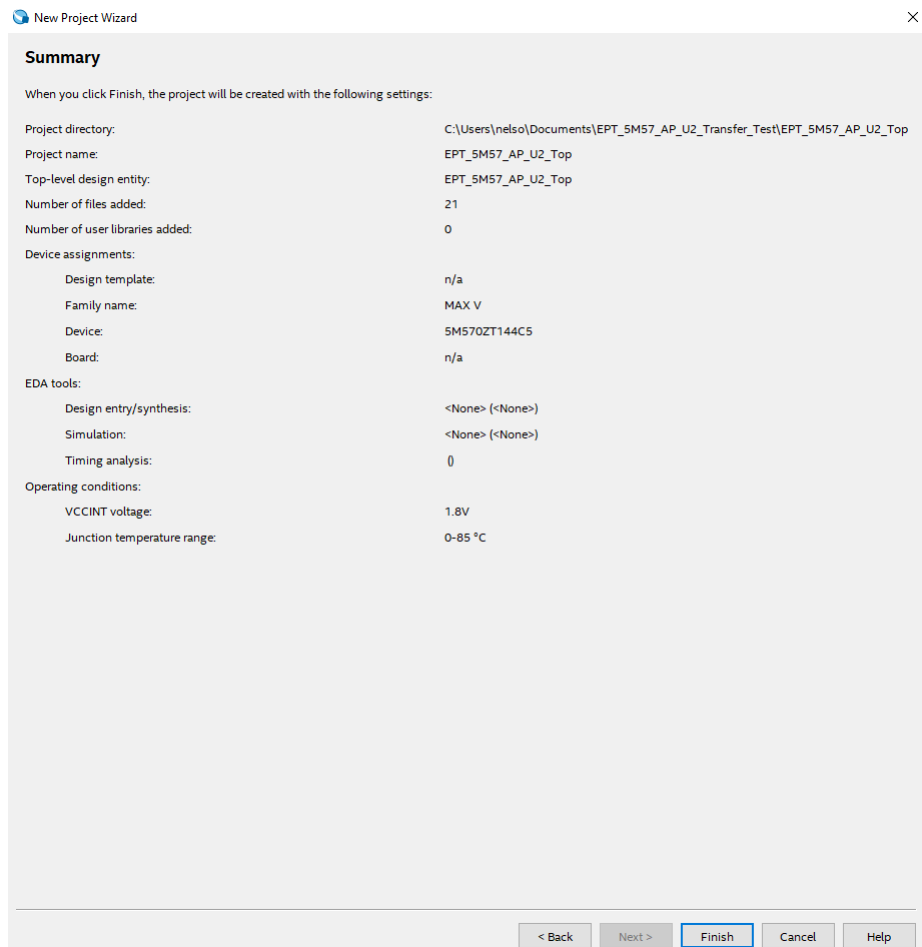
Select Next, leave defaults for the EDA Tool Settings.

UnoMax CPLD Development System User Manual



Select Next, then select Finish. You are done with the project level selections.

UnoMax CPLD Development System User Manual

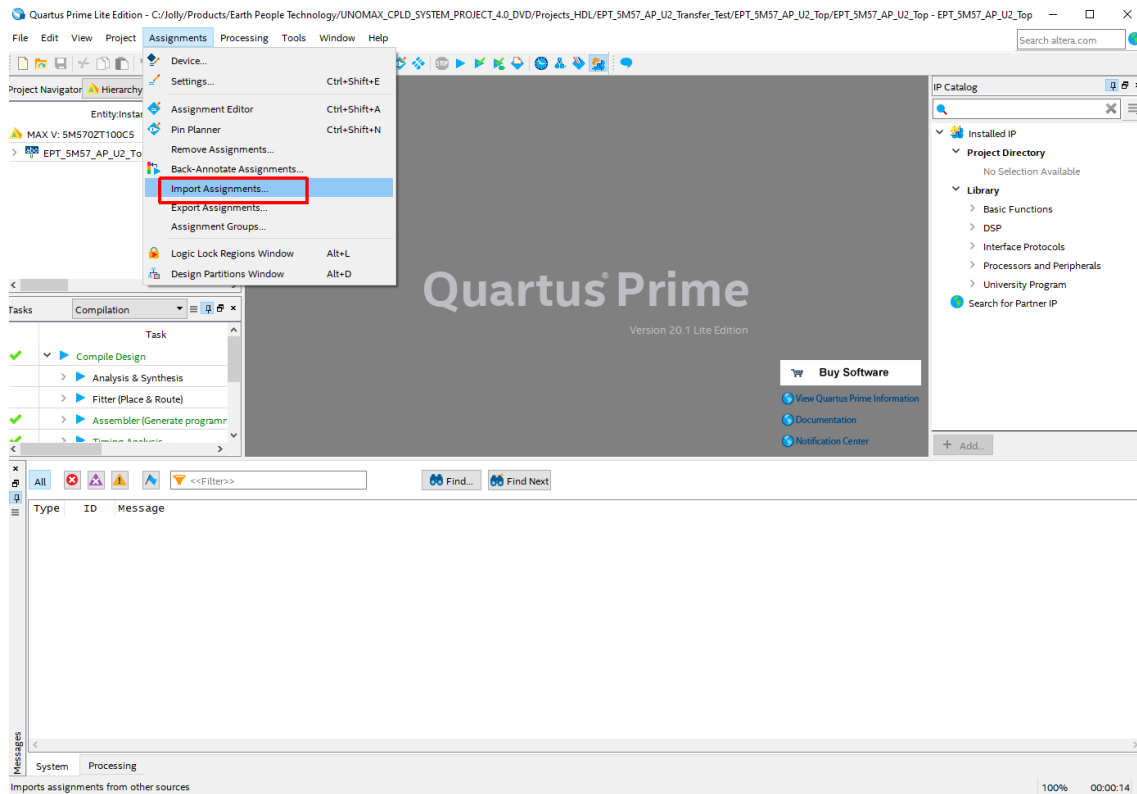


Next, we will select the pins and synthesize the project.

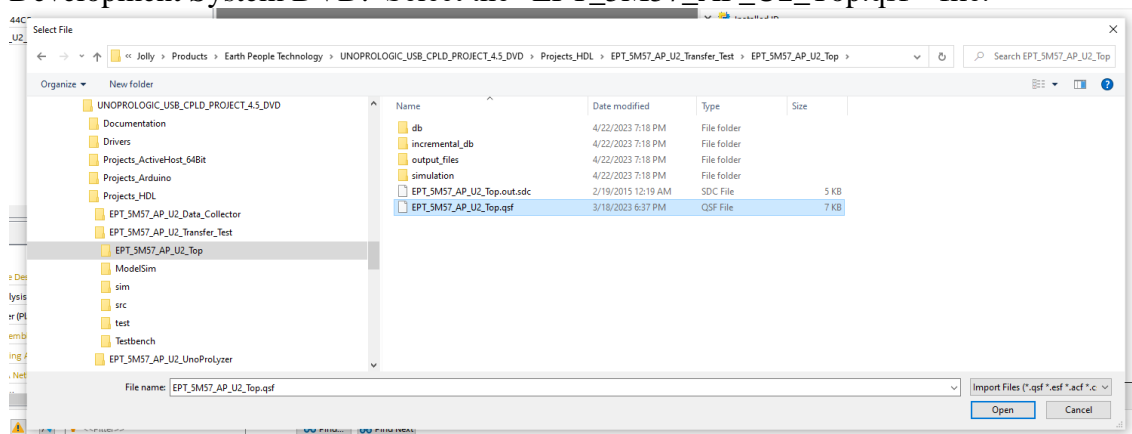
4.1.1 Selecting Pins and Synthesizing

With the project created, we need to assign pins to the project. The signals defined in the top level file (in this case: EPT_5M57_AP_U2_Top.v) will connect directly to pins on the CPLD. The Pin Planner Tool from Quartus Prime will add the pins and check to verify that our pin selections do not violate any restrictions of the device. In the case of this example we will import pin assignments that created at an earlier time. Under Assignments, Select Import Assignments.

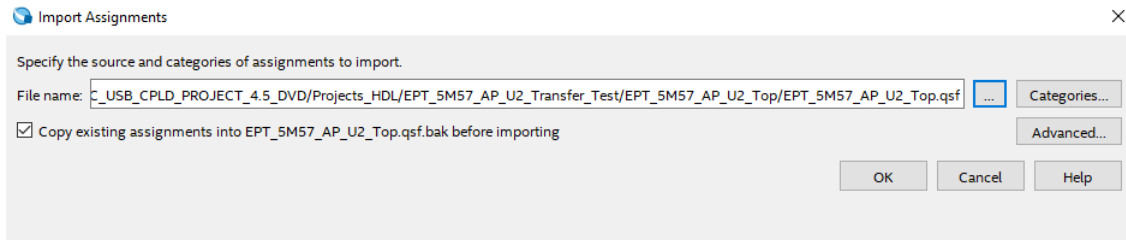
UnoMax CPLD Development System User Manual



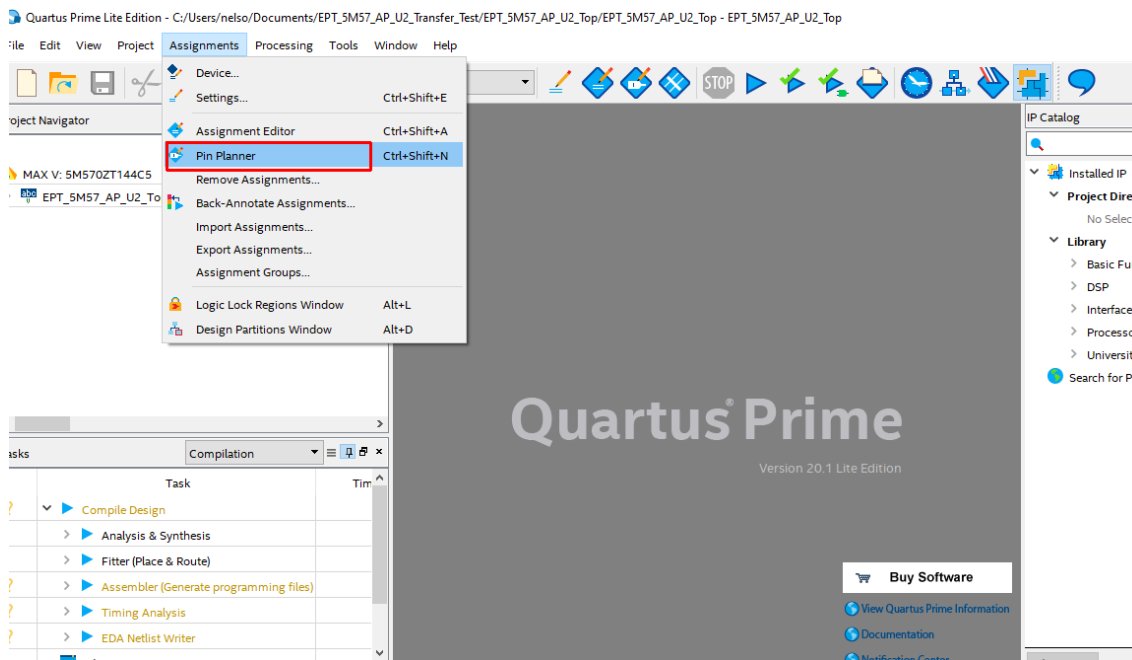
At the Import Assignment dialog box, Browse to the `\Projects_HDL\EPT_Transfer_Test\EPT_5M57_AP_U2_TOP` folder of the UnoMax Development System DVD. Select the “EPT_5M57_AP_U2_Top.qsf” file.



UnoMax CPLD Development System User Manual

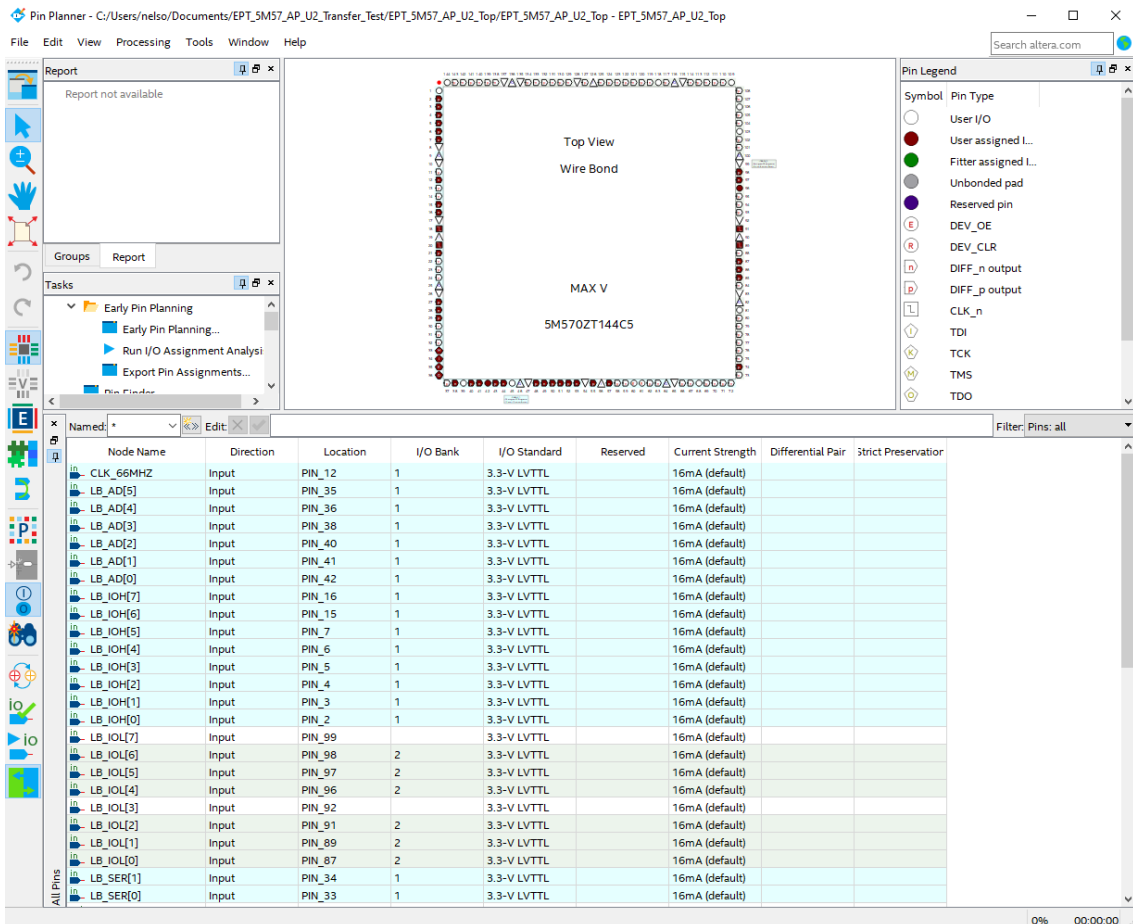


Click Ok. Under Assignments, Select Pin Planner. Verify the pins have been imported correctly.



The pin locations should not need to be changed for UnoMax Development System. However, if you need to change any pin location, just click on the “location” column for the particular node you wish to change. Then, select the new pin location from the drop down box.

UnoMax CPLD Development System User Manual

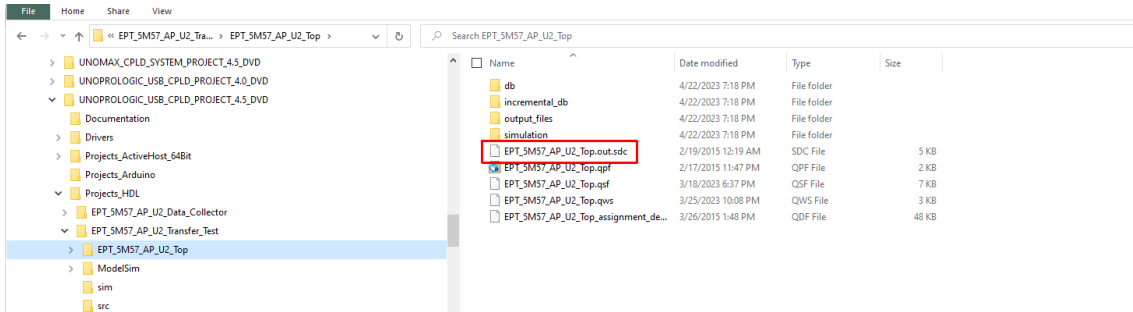


Exit the Pin Planner. Next, we need to add the Synopsys Design Constraint file. This file contains timing constraints which forces the built in tool called TimeQuest Timing Analyzer to analyze the path of the synthesized HDL code with setup and hold times of the internal registers. It takes note of any path that may be too long to appropriately meet the timing qualifications. For more information on TimeQuest Timing Analyzer, see

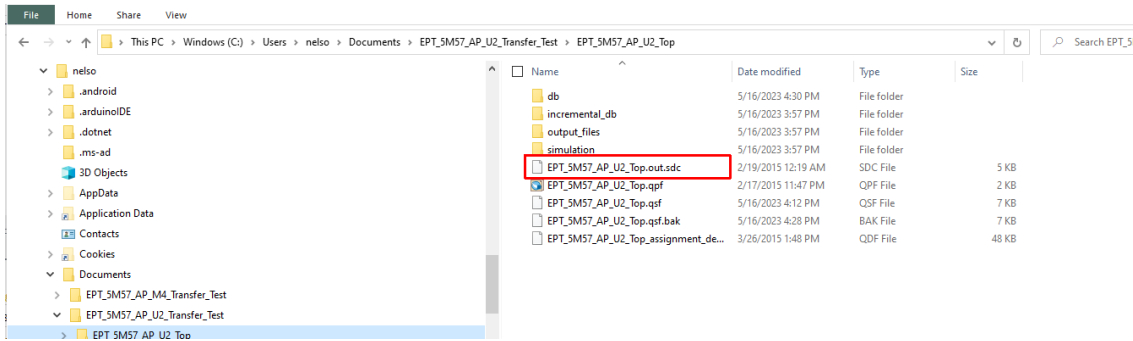
https://ftp.intel.com/Public/Pub/fpgaup/pub/Teaching_Materials/current/Tutorials/VHDL/Timequest.pdf

Browse to the \Projects_HDL\EPT_Transfer_Test \ EPT_5M57_AP_U2_Top folder of the MegaProLogic Development System DVD. Select the “EPT_5M57_AP_U2_Top.sdc” file.

UnoMax CPLD Development System User Manual

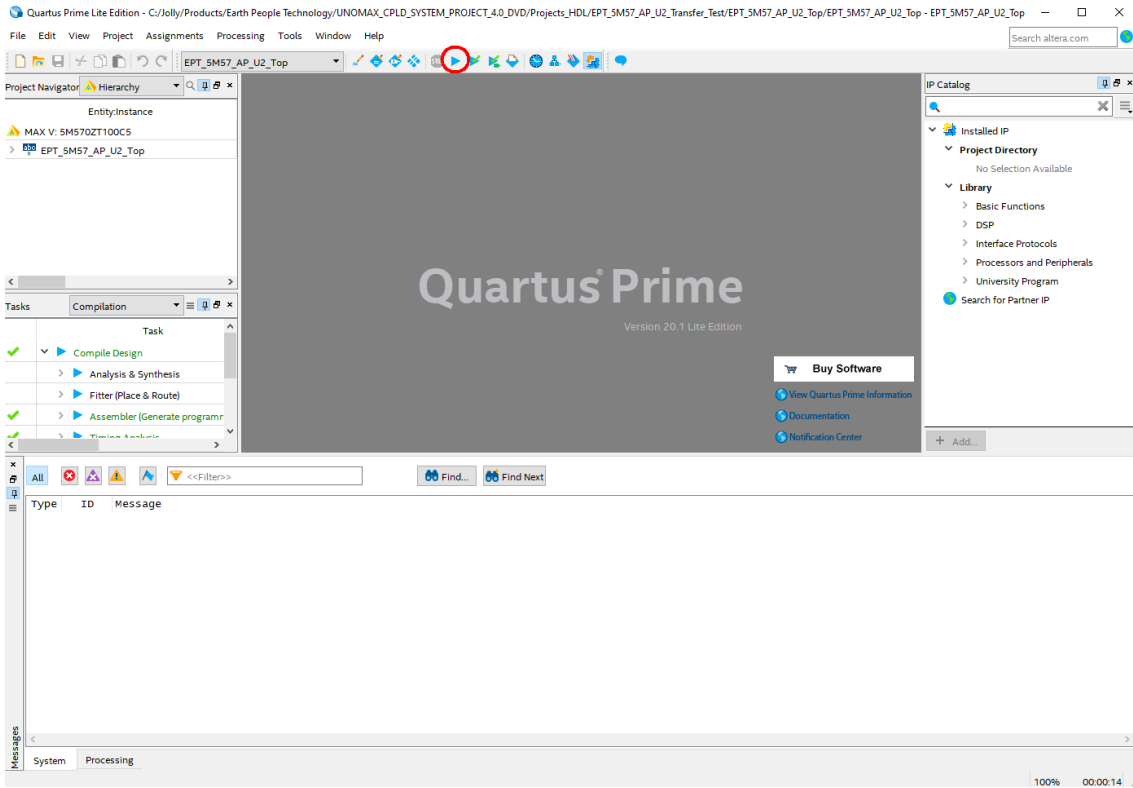


Copy the file and browse to C:\Users\nelso\Documents\EPT_5M57_AP_U2_Transfer_Test\EPT_5M57_AP_U2_Top directory. Paste the file.

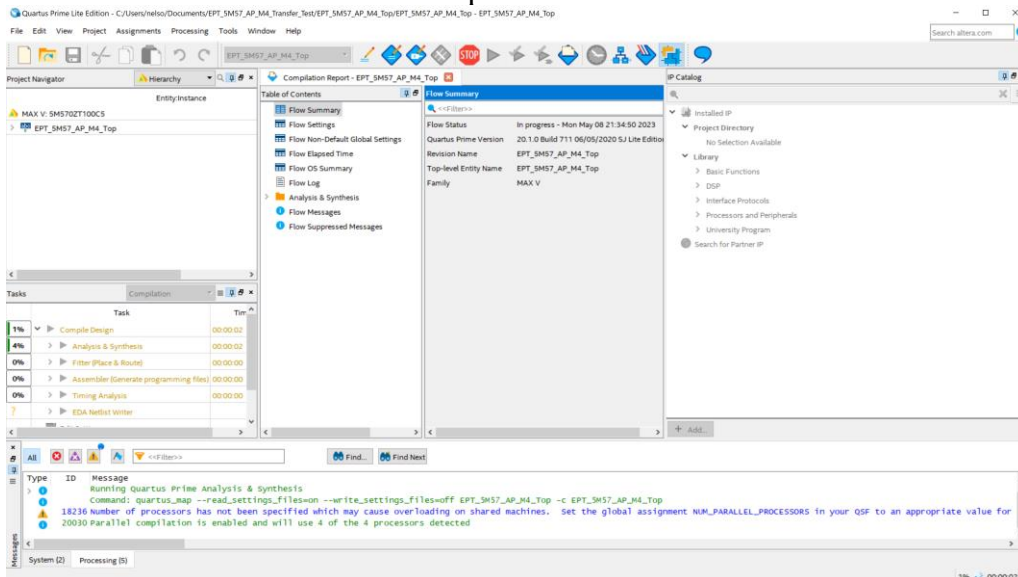


Select the Start Compilation button.

UnoMax CPLD Development System User Manual

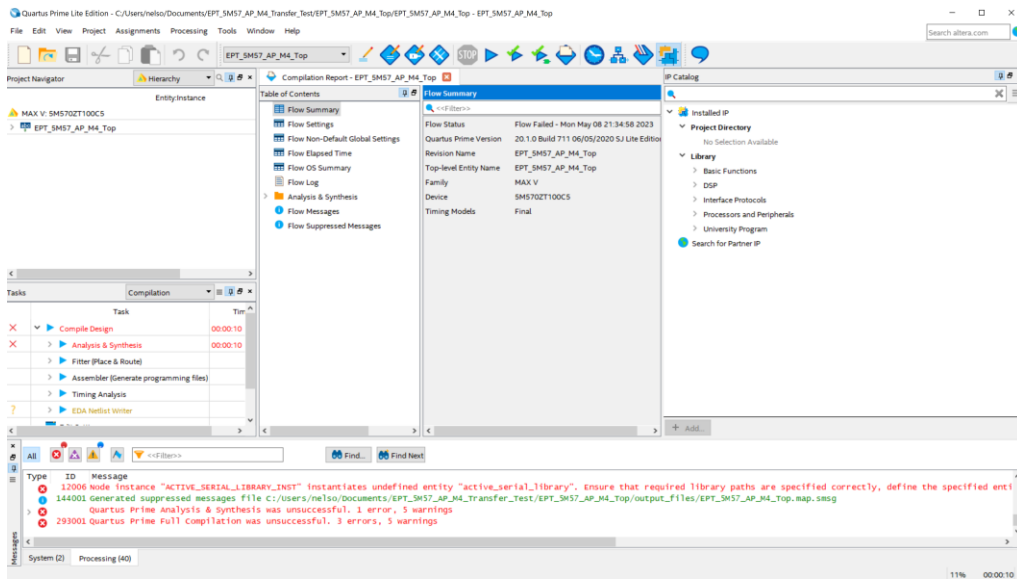


Then compilation will start.



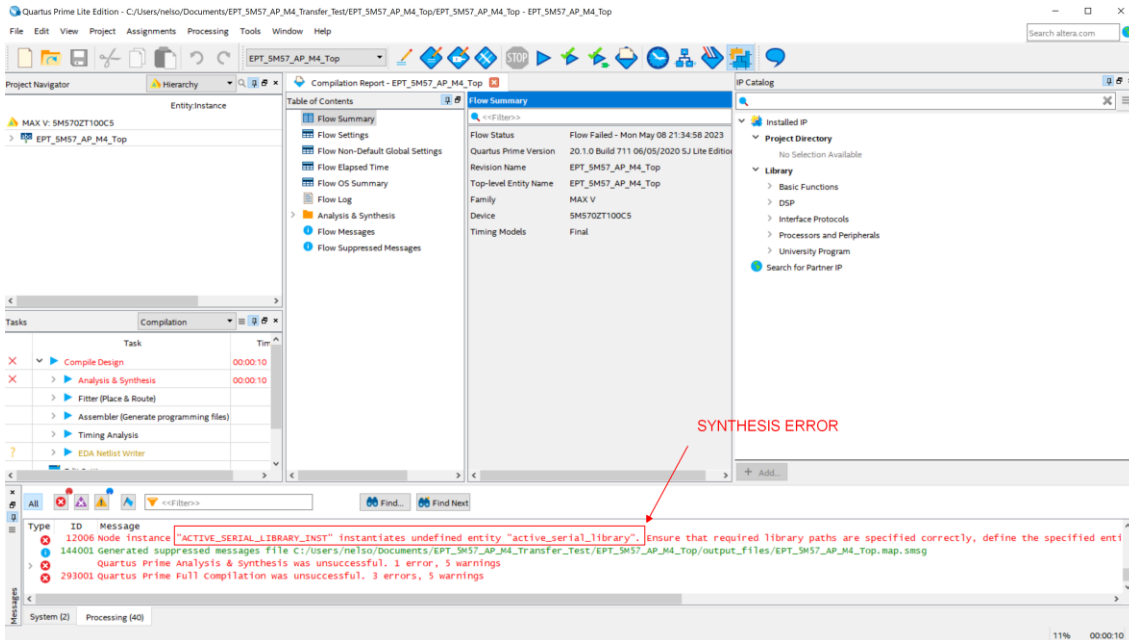
UnoMax CPLD Development System User Manual

If you forget to include a file or some other error you should expect to see a screen similar to this:

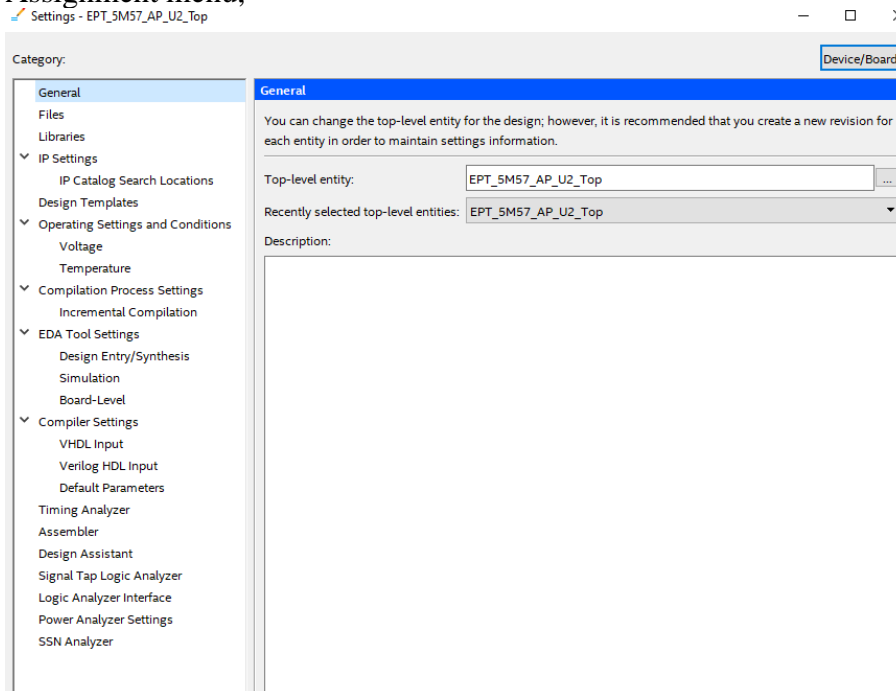


Click on the “Processing” tab at bottom to see the error.

UnoMax CPLD Development System User Manual

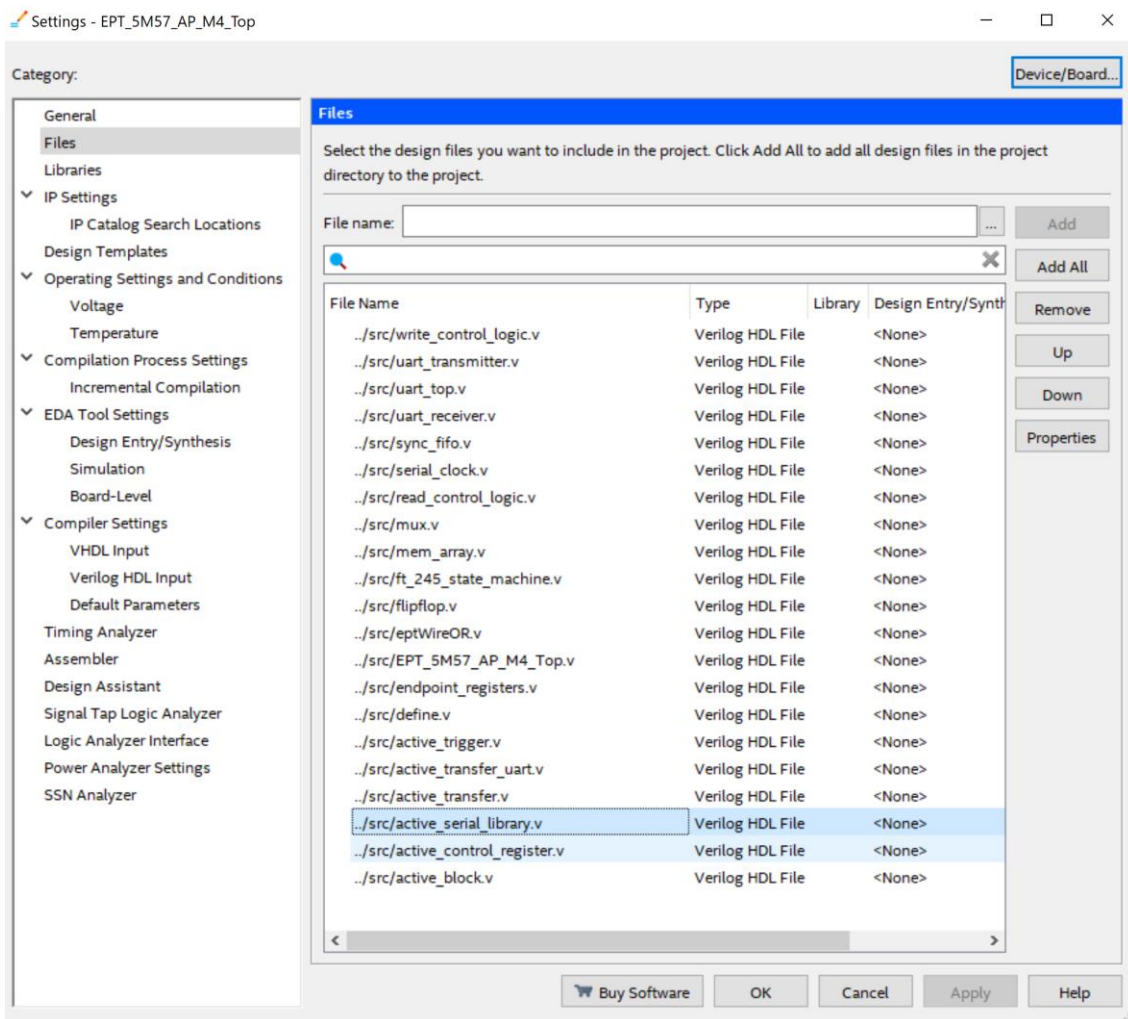


The error in this case is the missing file “active_serial_library.v”. Click on the Assignment menu,



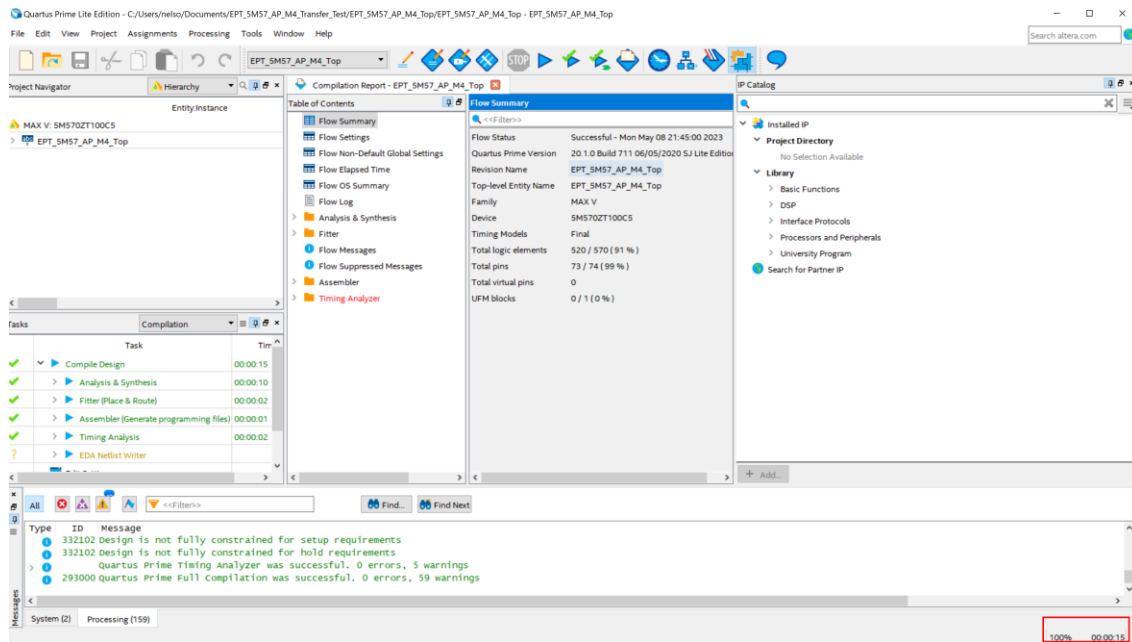
UnoMax CPLD Development System User Manual

then select Settings, then select Files. Add the “active_serial_library.v” file from the database.



Click Ok then re-run the Compile process. After successful completion, the screen should look like the following:

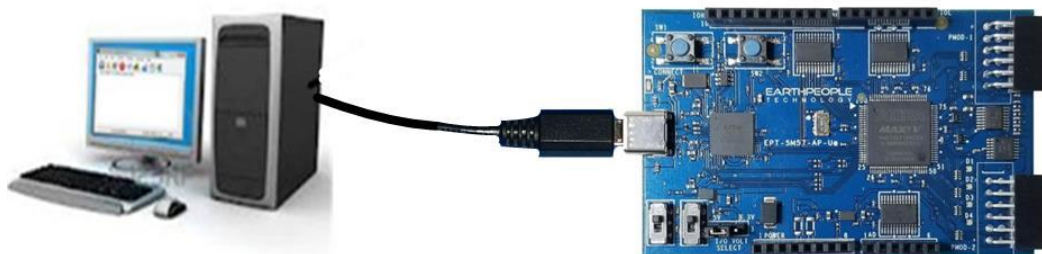
UnoMax CPLD Development System User Manual



At this point the project has been successfully compiled, synthesized and a programming file has been produced. See the next section on how to program the CPLD.

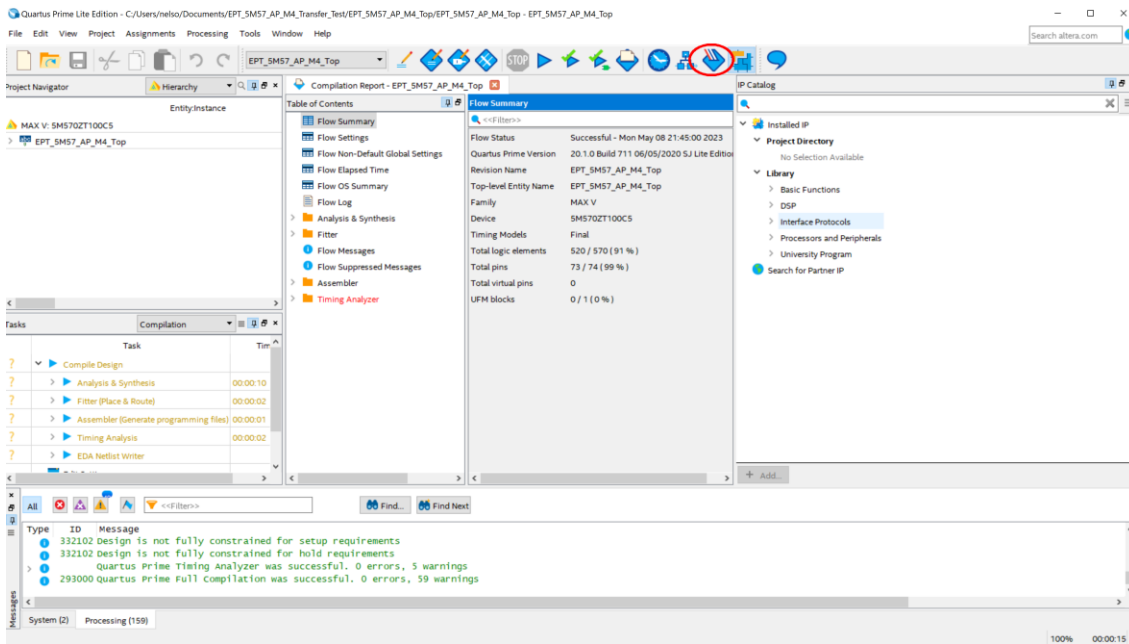
4.2 Programming the CPLD

Programming the CPLD is quick and easy. All that is required is a standard USB-C cable and the EPT_Blaster Driver DLL. Connect the UnoMax to the PC, open up Quartus Prime Lite, open the programmer tool, and click the Start button. To program the CPLD, follow the steps to install the USB Driver and the JTAG Driver Insert for Quartus Prime Lite.

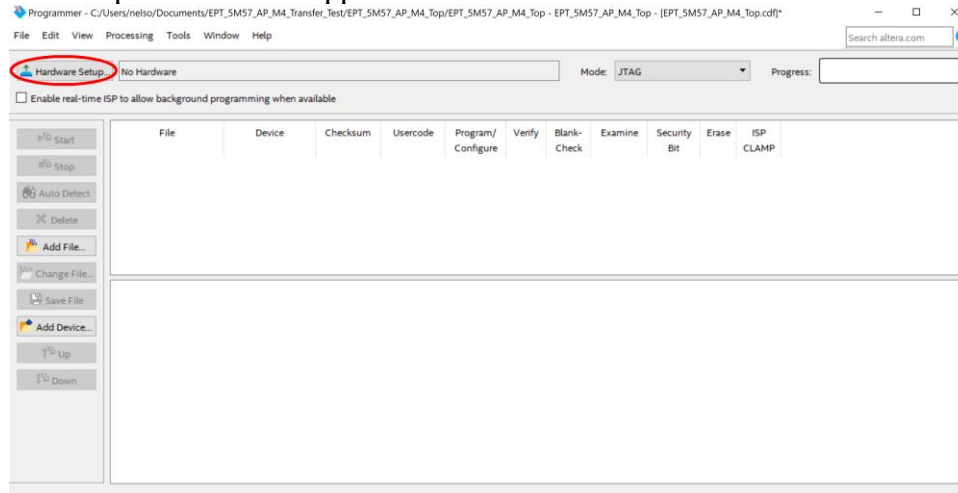


UnoMax CPLD Development System User Manual

If the project created in the previous sections is not open, open it. Click on the Programmer button.

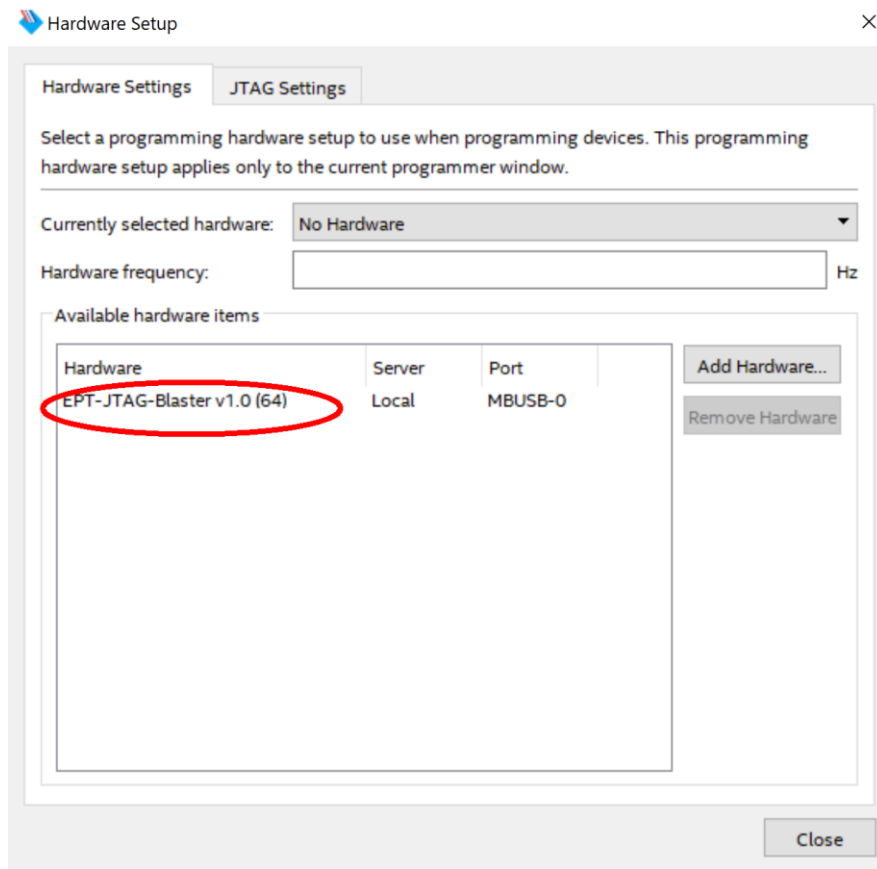


The Programmer Window will open up with the programming file selected. Click on the Hardware Setup button in the upper left corner.



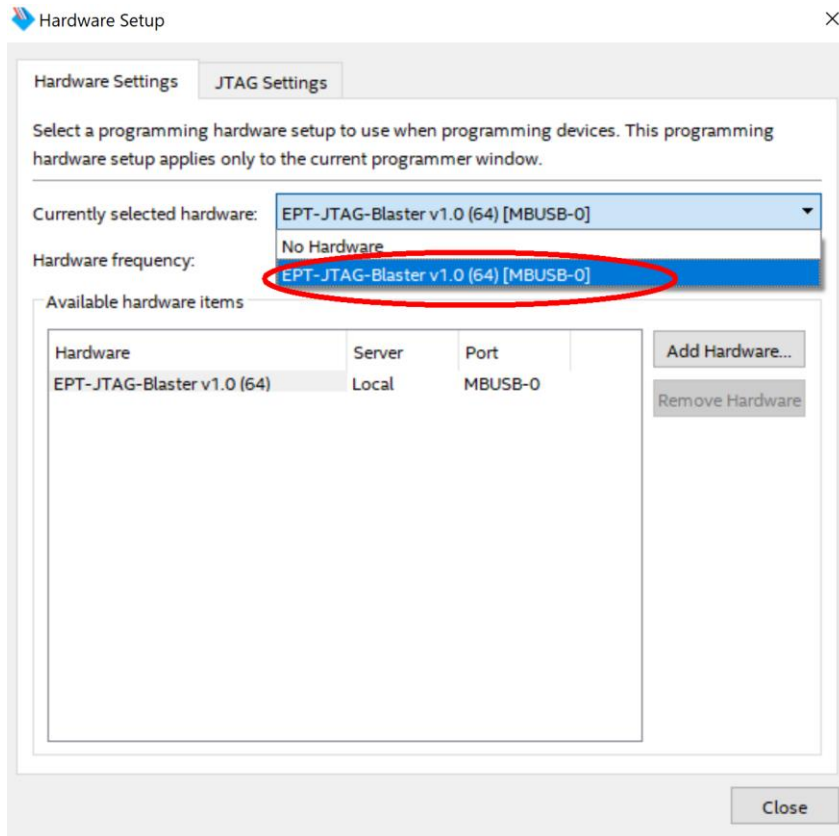
UnoMax CPLD Development System User Manual

The Hardware Setup Window will open. In the “Available hardware items”, double click on “EPT-Blaster v1.0”.

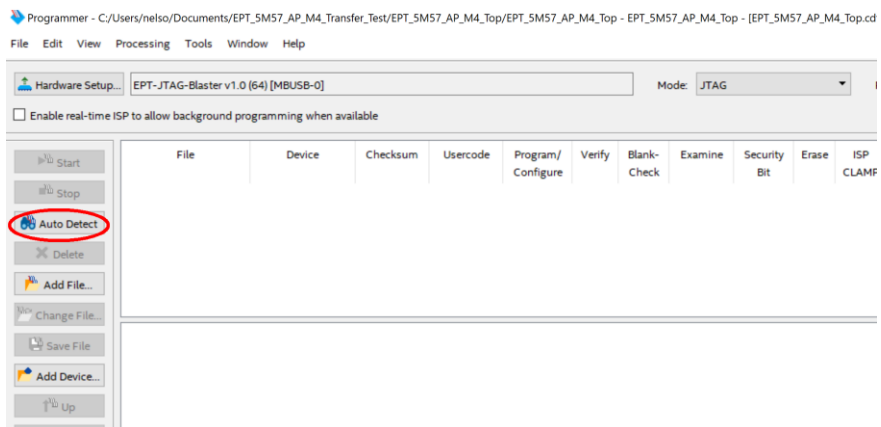


If you successfully double clicked, the “Currently selected hardware:” dropdown box will show the “EPT-Blaster v1.0 (64)”.

UnoMax CPLD Development System User Manual

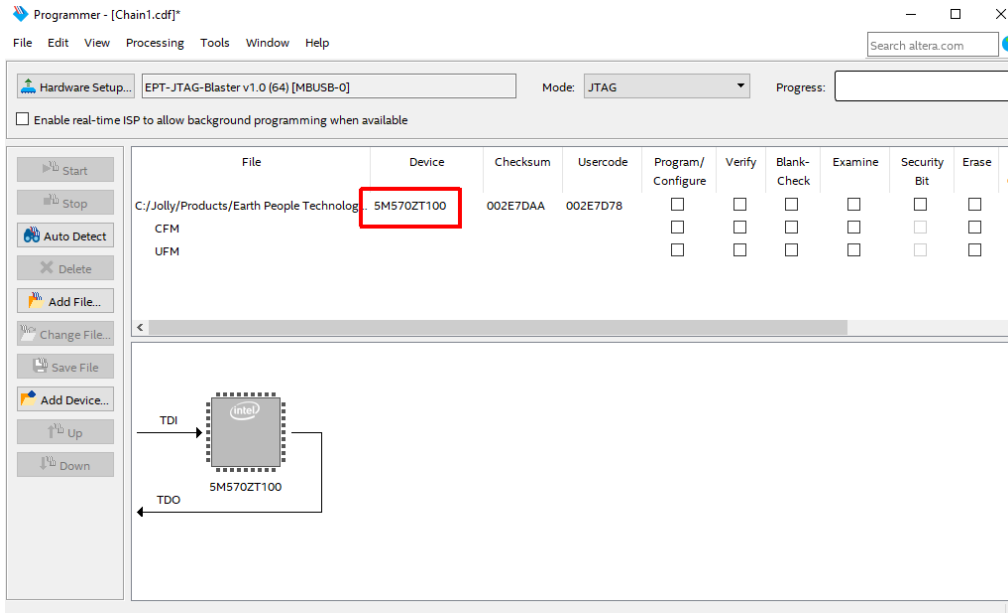


Click on the Auto-Detect button. This will verify that the EPT-Blaster driver can connect with the MegaProLogic device.

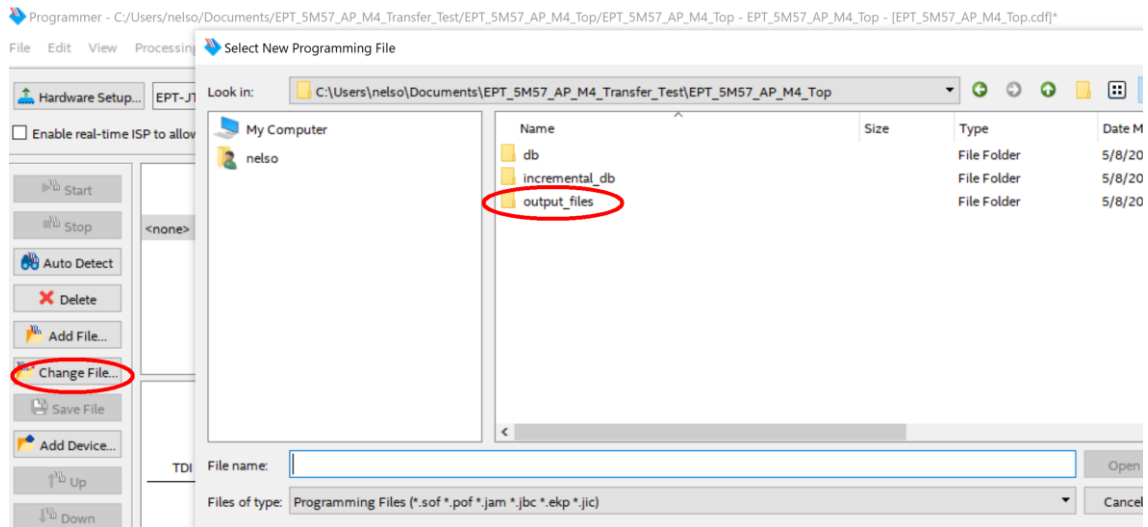


UnoMax CPLD Development System User Manual

Select the 5M570 under “Device”.

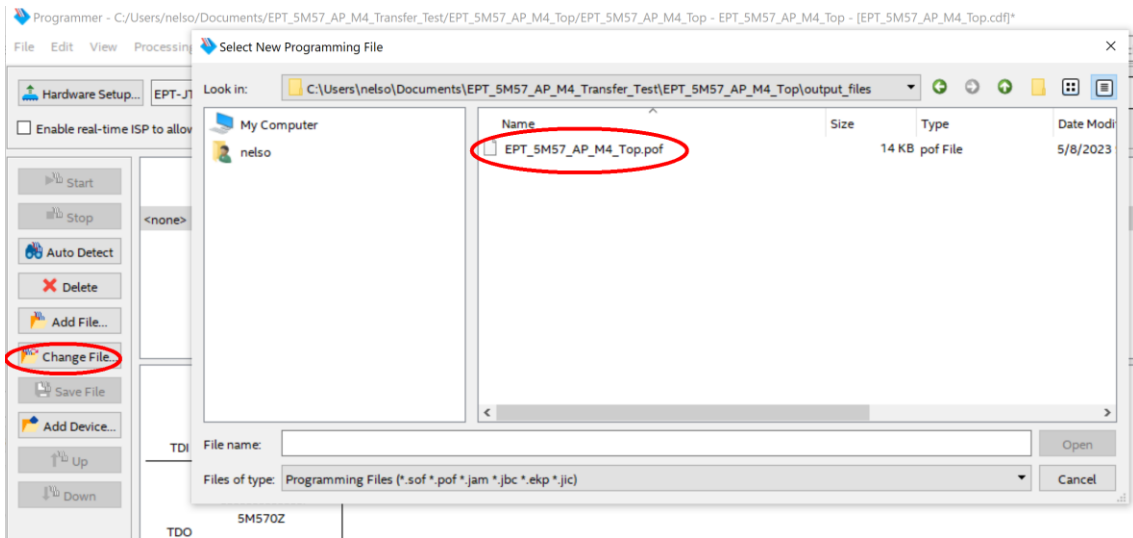


Click on the “Change File” button and browse to the output_files folder.



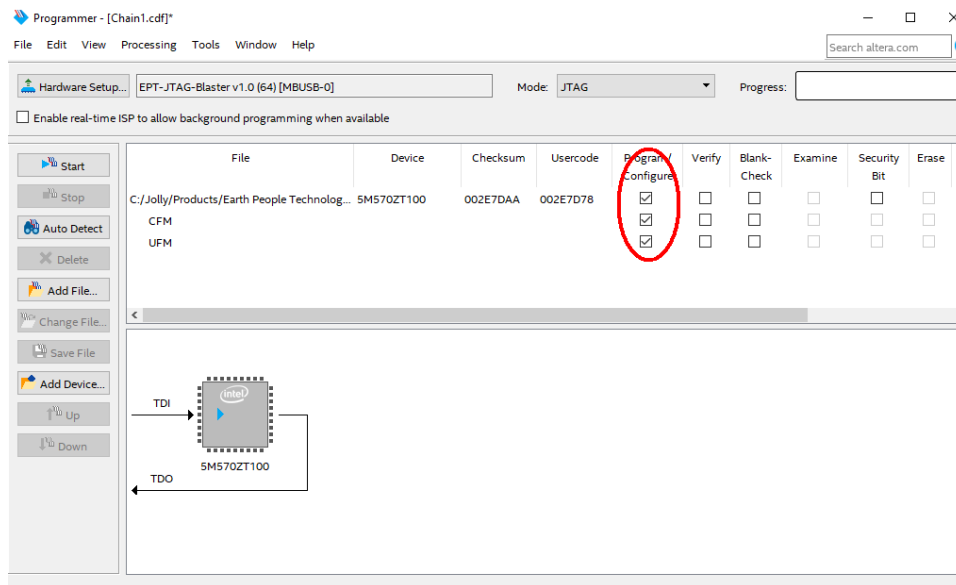
Click on the EPT_5M57_AP_M4_Top.pof file to select it.

UnoMax CPLD Development System User Manual



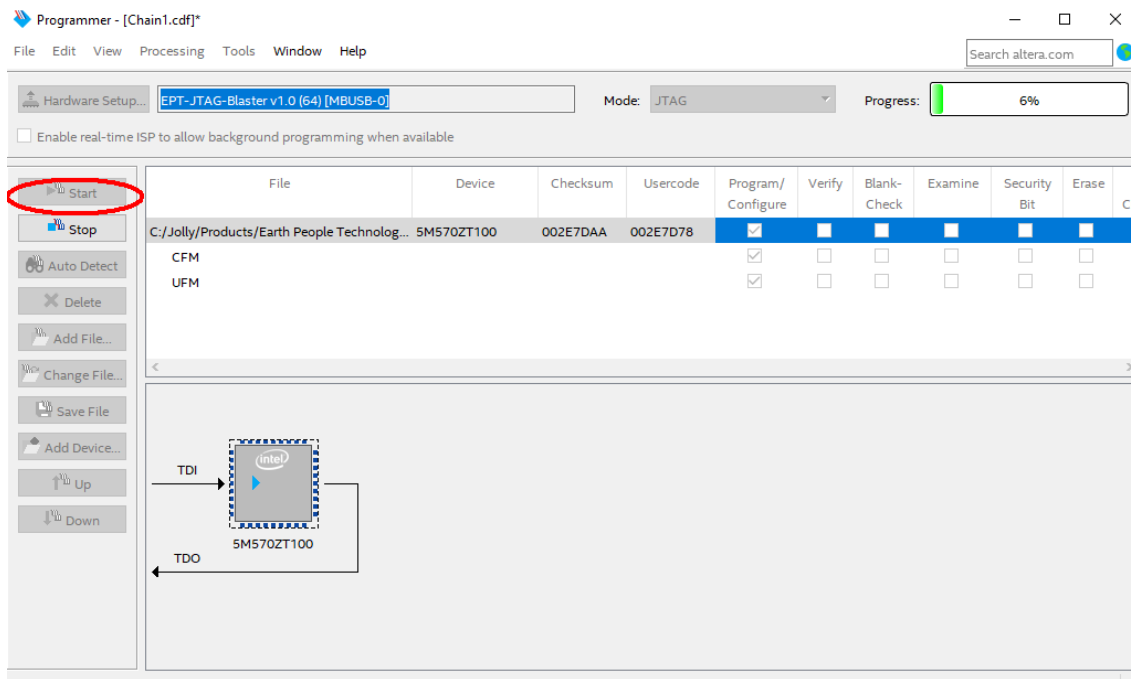
Click the Open button in the lower right corner.

Next, select the checkbox under the “Program/Configure” of the Programmer Tool. The checkboxes for the CFM and UFM will be selected automatically.



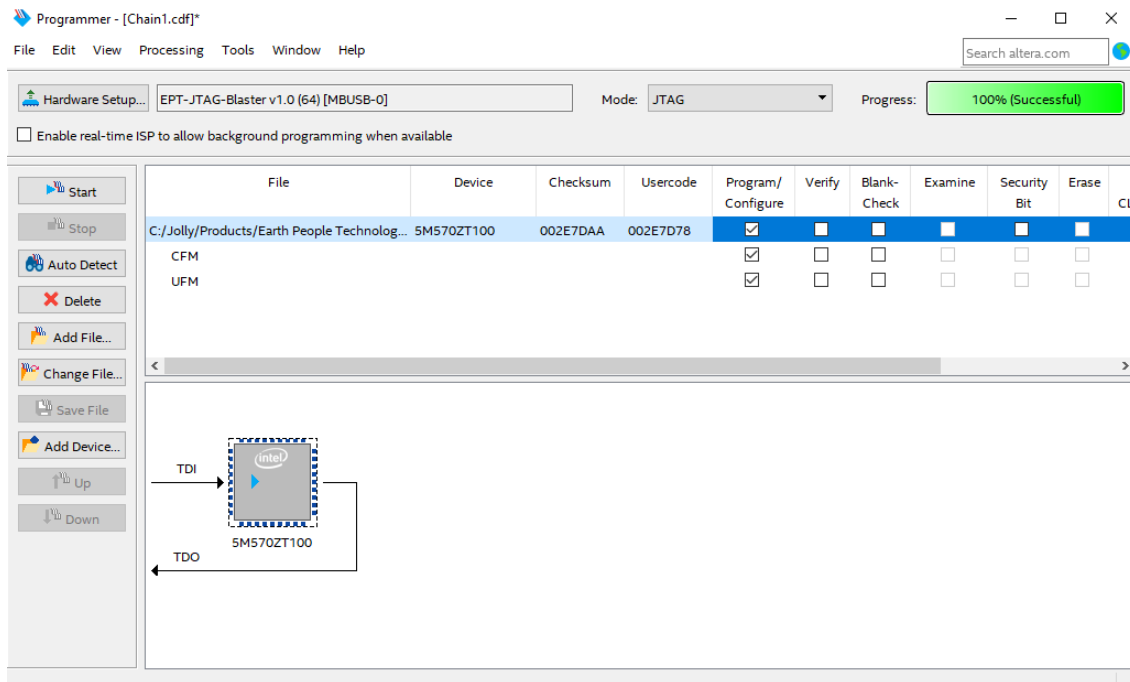
Click on the Start button to start programming the CPLD. The Progress bar will indicate the progress of programming.

UnoMax CPLD Development System User Manual



When the programming is complete, the Progress bar will indicate success.

UnoMax CPLD Development System User Manual



At this point, the UnoMax is programmed and ready for use. To test that the CPLD is properly programmed, bring up the Active Host Test Tool. Click on one of the LED's and verify that the LED selected lights up. Press one of the switches on the board and ensure that the switch is captured on the Active Host Test Tool. Now you are ready to write some code to transfer data between CPLD and PC.

5 Active Host Application

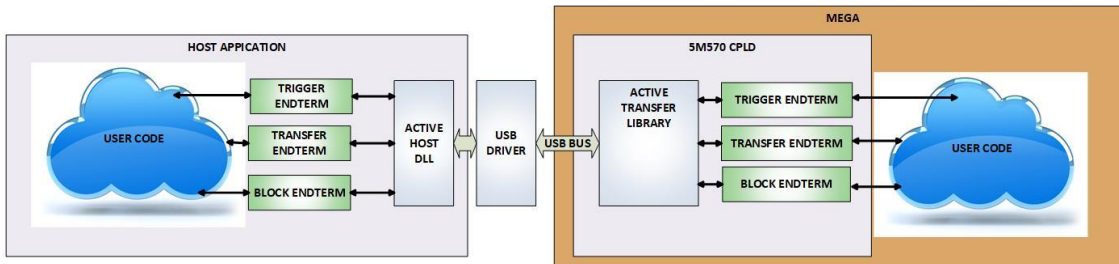
The Active Host SDK is provided as a dll which easily interfaces to application software written in C#, C++ or C. It runs on the PC and provides transparent connection from PC application code through the USB driver to the user CPLD code. The user code connects to "Endterms" in the Active Host dll. These host "Endterms" have complementary HDL "Endterms" in the Active Transfer Library. Users have seamless bi-directional communications at their disposal in the form of:

- Trigger Endterm
- Transfer Endterm
- Block Endterm

User code writes to the Endterms as function calls. Just include the address of the individual module (there are eight individually addressable modules of each Endterm).

UnoMax CPLD Development System User Manual

Immediately after writing to the selected Endterm, the value is received at the HDL Endterm in the CPLD. The Trigger Endterms are used as “switches”. The user code can set a Trigger bit in the CPLD and cause an event to occur. The Transfer Endterm sends one byte to the CPLD. The Block Endterm sends a block of bytes. By using one of the Active Host Endterms, the user can create a dynamic, bi-directional, and configurable data transfer design.



5.1 Trigger EndTerm

The Trigger EndTerm is a software component that provides a direct path from the users application to the commensurate Trigger EndTerm in the CPLD. The Trigger has eight bits and is intended to be used to provide a switch at the opposite EndTerm. They are fast acting and are not stored or buffered by memory. When the user code sets a Trigger, it is immediately passed through to the opposite EndTerm via the USB driver. When receiving Trigger, the user application is required to respond to a callback from the Active Host dll.

5.2 Transfer(Byte) EndTerm

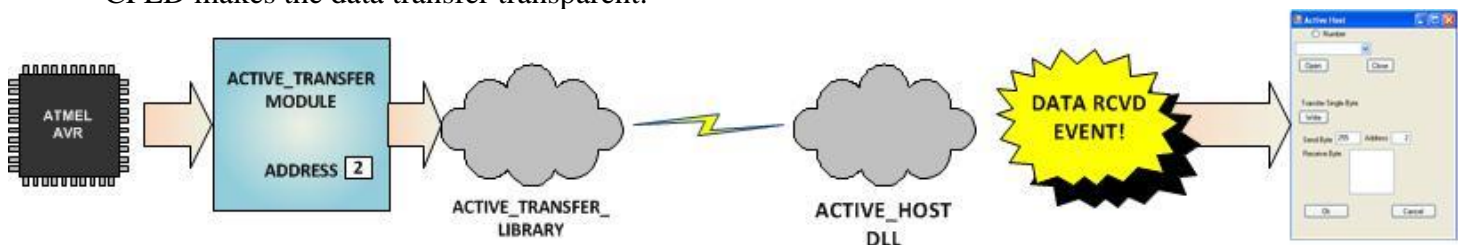
The Transfer EndTerm is a software component that provides a direct path from the users application to the commensurate Transfer EndTerm in the CPLD. It is used to transfer a byte to and from the CPLD. Eight separate Transfer EndTerm modules can be instantiated in the CPLD. Each module is addressed by the user application. Sending a byte is easy, just use the function call with the address and byte value. The byte is immediately sent to the corresponding EndTerm in the CPLD. Receiving a byte is just as easy, a callback function is registered at initialization. When the CPLD transmits a byte using its EndTerm, the callback function is called in the user application. The user code must store this byte in order to use it. The incoming Transfers are stored in a circular buffer in memory. This allows the user code to fetch the transfers with out losing bytes.

5.3 Block EndTerm

The Block EndTerm is a software component that provides a direct path from the users application to the commensurate Block EndTerm in the CPLD. The Block EndTerm is used to transfer a complete block to the CPLD. Block size is limited to 1 to 256 bytes. Eight separate Block EndTerm modules can be instantiated in the CPLD. Each module is addressed by the user application. Sending a block is easy, just use the function call with the address, block length, byte array. The block is buffered into a circular buffer in memory then transmitted via the USB bus to the Block EndTerm in the CPLD. Receiving a block is just as easy, a callback function is registered at initialization. When the CPLD transmits a block using its EndTerm, the callback function is called in the user application. The incoming Transfers are stored in a circular buffer in memory. This allows the user code to fetch the transfers with out losing bytes.

5.4 Active Host DLL

The Active_Host DLL is designed to transfer data from the CPLD when it becomes available. The data will be stored into local memory of the PC, and an event will be triggered to inform the user code that data is available from the addressed module of the CPLD. This method of automatically moving data from the user code Endterm in the CPLD makes the data transfer transparent.



The data seamlessly appears in Host PC memory from the UnoMax. The user code will direct the data to a control such as a textbox on a Windows Form. The transparent receive transfer path is made possible by a Callback mechanism in the Active Host dll. The dll calls a registered callback function in the user code. The user code callback can be designed to generate any number of events to handle the received data.

The user application will access the CPLD by use of functions contained in the Active Host dll. The functions to access the CPLD are:

- EPT_AH_SendTrigger ()
- EPT_AH_SendByte ()
- EPT_AH_SendBlock ()



UnoMax CPLD Development System User Manual

- EPT_AH_SendTransferControlByte()

5.4.1 Active Host Open Device

To use the library functions for data transfer and triggering, an Earth People Technology device must be opened. The first function called when the Windows Form loads up is the `<project_name>_Load()`. This function is called automatically upon the completion of the Windows Form, so there is no need to do anything to call it. Once this function is called, it in turn calls the `ListDevices()`. Use the function `List Devices()` to detect all EPT devices connected to the PC.

UnoMax CPLD Development System User Manual

```
private void EPT_Transfer_Test_Load(object sender, System.EventArgs e)
{
    //String buffer
    String PortText = "";

    //Index registers
    int Index = 0, EPTgroupNumber = 0;

    // Call the List Devices function
    List<string> names = ComPortNames("0403", "6010");

    // Get a list of serial port names.
    string[] ports;
    ports = SerialPort.GetPortNames();

    if (names.Count > 0)
    {
        foreach (String port in ports)
        {
            //Compare port name with the found VID/PID
            //combinations. Add them to Matching port list
            //and comboDevList
            if (names.Contains(port))
            {
                MatchingComPortList[Index] = port;

                if (Index == 0)
                {
                    PortText = "EPT JTAG Blaster " + EPTgroupNumber;
                    Index++;
                }
                else
                {
                    PortText = "EPT Serial Communications " + EPTgroupNumber++;
                    Index++;
                }
                cmbDevList.Items.Add(PortText);
            }
        }
    }
    else
        MessageBox.Show("No EPT Devices found!");

    //SetButtonEnables_Close();
}
```

The ListDevices() function calls the

```
ports = SerialPort.GetPortNames();
```

UnoMax CPLD Development System User Manual

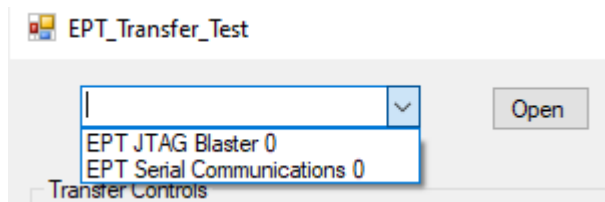
to determine the Serial devices attached to the PC. Next,

```
if (names.Contains(port))
```

is called inside a for loop to return the ASCII name of each Serial device attached to the PC. It will automatically populate the combo box, cmbDevList with all the EPT devices it finds.

```
cmbDevList.Items.Add(PortText);
```

The user will select the device from the drop down combo box. This can be seen when the Windows Form is opened and the cmbDevList combo box is populated with all the devices. The selected device will be stored as an index number in the variable device_index.



In order to select the device, the user will click on the “Open” button which calls the

```
OpenSerialPort1()
```

function. The device_index is passed into the function. If the function is successful, the device name is displayed in the label, labelDeviceCnt. Next, the Open button is grayed out and the Close button is made active.

UnoMax CPLD Development System User Manual

```

1 reference
public bool OpenSerialPort1()
{
    try
    {
        //Set the serial port parameters
        serialPort_AH.PortName = PortName;
        serialPort_AH.BaudRate = Convert.ToInt32(BaudRate);
        serialPort_AH.Parity = (Parity)Enum.Parse(typeof(Parity), vParity);
        serialPort_AH.DataBits = Convert.ToInt16(DataBits);
        serialPort_AH.StopBits = (StopBits)Enum.Parse(typeof(StopBits), StopBits);
        serialPort_AH.Handshake = (Handshake)Enum.Parse(typeof(Handshake), pHandshake);

        if (!serialPort_AH.IsOpen)
        {
            serialPort_AH.Open();
            btnOpenDevice.Enabled = false;
            btnCloseDevice.Enabled = true;
            //textBox1.ReadOnly = false;
            return true;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    return false;
}

```

5.4.2 Active Host Triggers

The user application can send a trigger to the CPLD by using the EPT_AH_SendTrigger() function. First, open the EPT device to be used with OpenSerialPort1(). Call the function with the bit or bits to assert high on the trigger byte as the parameter. Then execute the function, the trigger bit or bits will momentarily assert high in the user code on the CPLD.

```

private void btnTrigger1_Click(object sender, EventArgs e)
{
    EPT_AH_SendTrigger((char) 1);
}

```

To detect a trigger from the CPLD, the user application must subscribe to the event created when the incoming trigger has arrived at the Read Callback function. The Read Callback must store the incoming trigger in a local variable. A switch statement is used to decode which event should be called to handle the incoming received data.

UnoMax CPLD Development System User Manual

- TRIGGER_IN
- TRANSFER_IN
- BLOCK_IN

```

<#endif>
public void EPT_AH_Receive(byte[] receiveBytes)
{
    uint c, a, p, l, index;
    //Compare first byte to the incoming message code
    string s = String.Empty;
    foreach (byte b in receiveBytes)
    {
        s += String.Format("{0:x2}", (int)System.Convert.ToInt32(b.ToString()));
        s += "\r\n";
    }
    //tbBlockRcv.AppendText(s);
    //this.Invoke(new MethodInvoker(delegate () { textBox1.AppendText(s); }));

    //Write the command into the EPTReceiveDevice
    c = (uint)receiveBytes[0];
    c = c & 0xf8;

    //Write the address into EPTReceiveDevice
    a = (uint)receiveBytes[1];
    a = a & 0x07;
    EPTReceiveDevice.Address = a;
    //Display Address to text box
    string r = String.Empty;
    r += String.Format("EPTReceiveDevice Address= {0:x2}", EPTReceiveDevice.Address);
    r += "\r\n";
    //this.Invoke(new MethodInvoker(delegate () { textBox1.AppendText(r); }));

    switch (c)
    {
        case 0xc8:
            //this.Invoke(new MethodInvoker(delegate () { textBox1.AppendText("Trigger Recieved\r\n"); }));
            EPTReceiveDevice.Command = TRIGGER_IN_COMMAND;
            break;
        case 0xd0:
            //this.Invoke(new MethodInvoker(delegate () { textBox1.AppendText("Transfer Byte Recieved\r\n"); }));
            EPTReceiveDevice.Command = TRANSFER_IN_COMMAND;
            break;
        case 0xe0:
            //this.Invoke(new MethodInvoker(delegate () { textBox1.AppendText("Block Recieved\r\n"); }));
            EPTReceiveDevice.Command = BLOCK_IN_COMMAND;
    }
}

```



UnoMax CPLD Development System User Manual

```
private void EPTParseReceive(object sender, System.EventArgs e)
{
    switch (EPTReceiveData.Command)
    {
        case TRIGGER_OUT_COMMAND:
            TriggerOutReceive();
            break;
        case TRANSFER_OUT_COMMAND:
            TransferOutReceive();
            break;
        case BLOCK_OUT_COMMAND:
            BLockOutReceive();
            break;
        default:
            break;
    }
}
```

The event handler function for the TRIGGER_IN's uses a switch statement to determine which trigger was asserted and what to do with it.

```
public void Receive_Trigger_In(object sender, EventArgs e)
{
    switch (ept_data.Payload)
    {
        case 0x01:
            lLabelSwitch1.Text = "Switch 1\n Pressed";
            break;
        case 0x02:
            lLabelSwitch2.Text = "Switch 2\n Pressed";
            break;
        case 0x04:
            lLabelSwitch1.Text = "";
            lLabelSwitch2.Text = "";
            break;
    }
}
```

The receive callback method is complex, however, Earth People Technology has created several projects which implement callbacks. Any part of these sample projects can be copied and pasted into a user's project.

5.4.3 Active Host Byte Transfers

The Active Host Byte Transfer EndTerm is designed to send/receive one byte to/from the EPT Device. To send a byte to the Device, the appropriate address must be selected



UnoMax CPLD Development System User Manual

for the Transfer module in the CPLD. Up to eight modules can be instantiated in the user code on the CPLD. Each module has its own address.

```
private void btnWriteByte_Click(object sender, EventArgs e)
{
    int ibyte, address_to_device;
    ibyte = Convert.ToInt32(tbNumBytes.Text);
    address_to_device = Convert.ToInt32(tbAddress.Text);
    EPT_AH_SendByte(address_to_device, (char)ibyte);
}
```

Use the function `EPT_AH_SendByte()` to send a byte to the selected module. First, open the EPT device to be used with `openSerialPort1()`. Then add the address of the transfer module as the first parameter of the `EPT_AH_SendByte()` function. Enter the byte to be transferred in the second parameter. Then execute the function, the byte will appear in the ports of the Active Transfer module in the user code on the CPLD.

To transfer data from the CPLD Device, a polling technique is used. This polling technique is because the Bulk Transfer USB is a Host initiated bus. The Device will not transfer any bytes until the Host commands it to. If the Device has data to send to the Host in an asynchronous manner (meaning the Host did not command the Device to send data), the Host must periodically check the Device for data in its transmit FIFO. If data exists, the Host will command the Device to send its data. The received data is then stored into local memory and register bits are set that will indicate data has been received from a particular address.

To receive a byte transfer from the Active host dll, user code must subscribe to the event created when the incoming byte transfer has arrived at the Read Callback function. The Read Callback must store the incoming transfer payload and module address in a local memory block. A switch statement is used to decode which event should be called to handle the incoming received data. The event handler function will check for any bytes read for that address.



UnoMax CPLD Development System User Manual

```
private void EPTParseReceive(object sender, System.EventArgs e)
{
    switch (EPTReceiveData.Command)
    {
        case TRIGGER_OUT_COMMAND:
            TriggerOutReceive();
            break;
        case TRANSFER_OUT_COMMAND:
            TransferOutReceive();
            break;
        case BLOCK_OUT_COMMAND:
            BLockOutReceive();
            break;
        default:
            break;
    }
}
```

The EventHandler function EPTParseReceive() is called by the Read function. The EPTParseReceive() function will examine the command of the incoming byte transfer and determine which receive function to call.

```
public void TransferOutReceive()
{
    string WriteRcvChar = "";
    WriteRcvChar = String.Format("{0}", (int)EPTReceiveData.Payload);
    tbDataBytes.AppendText(WriteRcvChar + ' ');
    tbAddress.Text = String.Format("{0:x2}", (uint)System.Convert.ToUInt32(EPTReceiveData.Address.ToString()));
}
```

For our example project, the TransferOutReceive() function writes the Transfer byte received to a text block. The receive callback method is complex, however, Earth People Technology has created several projects which implement callbacks. Any part of these sample projects can be copied and pasted into a user's project.

5.4.4 Active Host Block Transfers

The Active Host Block Transfer is designed to transfer blocks of data between Host and CPLD and vice versa through the Block EndTerm. This allows buffers of data to be transferred with a minimal amount of code. The Active Host Block module (in the User Code) is addressable, so up to eight individual modules can be instantiated and separately addressed. The length of the block to be transferred must also be specified. The Block EndTerm is limited to 1 to 256 bytes.



UnoMax CPLD Development System User Manual

To send a block, first, open the EPT device to be used with `EPT_AH_OpenDeviceByIndex()`. Next, use the `EPT_AH_SendBlock()` function to send the block. Add the address of the transfer module as the first parameter. Next, place the pointer to the buffer in the second parameter of `EPT_AH_SendBlock()`. Add the length of the buffer as the third parameter. Then execute the function, the entire buffer will be transferred to the USB chip. The data is available at the port of the Active Block module in the user code on the CPLD.

```
public unsafe void BlockCompare(object data)
{
    int BlockAddress = (int)data;
    byte[] cBuf = new Byte[device[BlockAddress].Length];

    if ((device[BlockAddress].Repetitions > 0) &
        !device[BlockAddress].TransferPending & !BlockTransferStop)
    {
        device[BlockAddress].TransferPending = true;
        Buffer.BlockCopy(block_8_in_payload, 0, cBuf, 0,
            device[BlockAddress].Length);
        fixed (byte* pBuf = cBuf)
        {
            EPT_AH_SendBlock(device[BlockAddress].Address,
                (void*)pBuf, (uint)device[BlockAddress].Length);
        }
        Thread.Sleep(1);
        EPT_AH_SendTransferControlByte((char)2, (char)2);
        Thread.Sleep(1);
        EPT_AH_SendTrigger((char)128);
        Thread.Sleep(1);
        EPT_AH_SendTransferControlByte((char)2, (char)0);

        if (BlockTransferInfinite)
            device[BlockAddress].Repetitions = 1;
        else
            device[BlockAddress].Repetitions--;
    }
}
```

To receive a block transfer from the CPLD Device, a polling technique is used by the Active Host dll. This is because the Bulk Transfer USB is a Host initiated bus. The Device will not transfer any bytes until the Host commands it to. If the Device has data to send to the Host in an asynchronous manner (meaning the Host did not command the Device to send data), the Host must periodically check the Device for data in its transmit FIFO. If data exists, the Host will command the Device to send its data. The



UnoMax CPLD Development System User Manual

received data is then stored into local memory and register bits are set that will indicate data has been received from a particular address. The receive callback function is then called from the Active Host dll. This function start a thread to do something with the block data.

To receive a byte transfer from the callback function, user code must subscribe to the event created when the incoming byte transfer has arrived at the Read Callback function. The Read Callback must store the incoming transfer payload and module address in a local memory block. A switch statement is used to decode which event should be called to handle the incoming received data. The event handler function will check for any bytes read for that address.

```
private void EPTParseReceive(object sender, System.EventArgs e)
{
    switch (EPTReceiveData.Command)
    {
        case TRIGGER_OUT_COMMAND:
            TriggerOutReceive();
            break;
        case TRANSFER_OUT_COMMAND:
            TransferOutReceive();
            break;
        case BLOCK_OUT_COMMAND:
            BLockOutReceive();
            break;
        default:
            break;
    }
}
```

The EPTParseReceive() function will examine the command of the incoming byte transfer and determine which receive function to call.



UnoMax CPLD Development System User Manual

```
public void Receive_Block_In(object sender, EventArgs e)
{
    device[ept_data.Address].TransferPending = false;
    Thread.Sleep(5);
    if (device[ept_data.Address].ContinuosCountTest == false)
    {
        Thread t = new Thread(new ParameterizedThreadStart(BlockCompare));
        t.Start(ept_data.Address);
    }
    if (device[ept_data.Address].Repititions == 0)
    {
        Thread u = new Thread(new ParameterizedThreadStart(Display_Block_In));
        u.Start(BlockCount);
    }
    else if (BlockTransferInfinite | device[ept_data.Address].ContinuosCountTest)
    {
        if ((BlockCount % 100) == 0)
        {
            Thread u = new Thread(new ParameterizedThreadStart(Display_Block_In));
            u.Start(BlockCount);
        }
    }
}
```

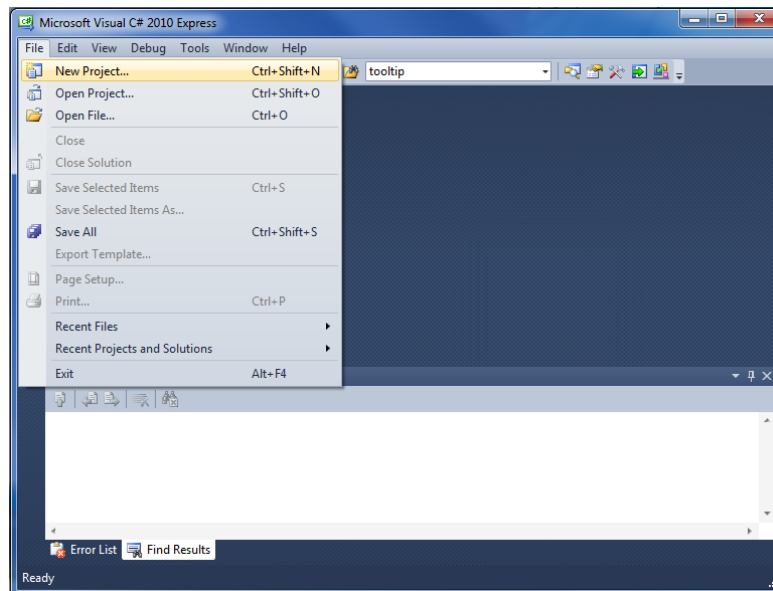
For our example project, the `Receive_Block_In()` function writes the Transfer block received to a text block. Assembling, Building, and Executing a .NET Project on the PC The Active Host Application DLL is used to build a custom standalone executable on the PC that can perform Triggers and Transfer data to/from the UnoMax. A standalone project can be range from a simple program to display and send data from the user to/from the CPLD. Or it can more complex to include receiving data, processing it, and start or end a process on the CPLD. This section will outline the procedures to take an example project and Assemble it, Build it, and Execute it.

This guide will focus on writing a Windows Forms application using the C# language for the Microsoft Visual Studio with .NET Framework. This is due to the idea that beginners can write effective Windows applications with the C# .NET Framework. They can focus on a subset of the language which is very similar to the C language. Anything that deviates from the subset of the C language, such as events and controls, will be explained as the explanation progresses. Any language can be used with the Active Host Application DLL.

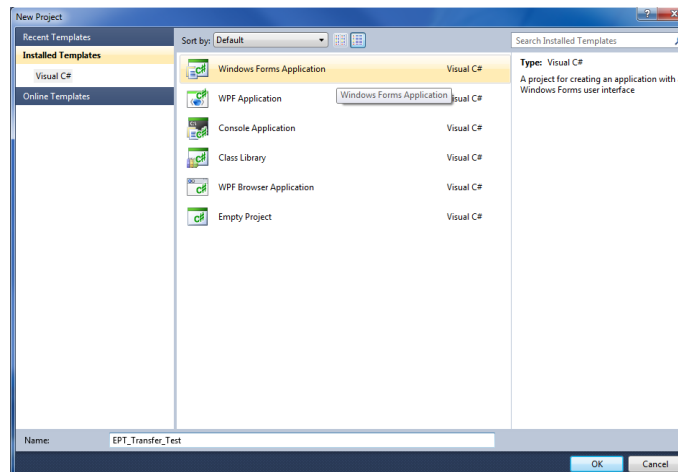
5.5 Creating a Project

Once the application is installed, open it up. Click on File->New Project.

UnoMax CPLD Development System User Manual

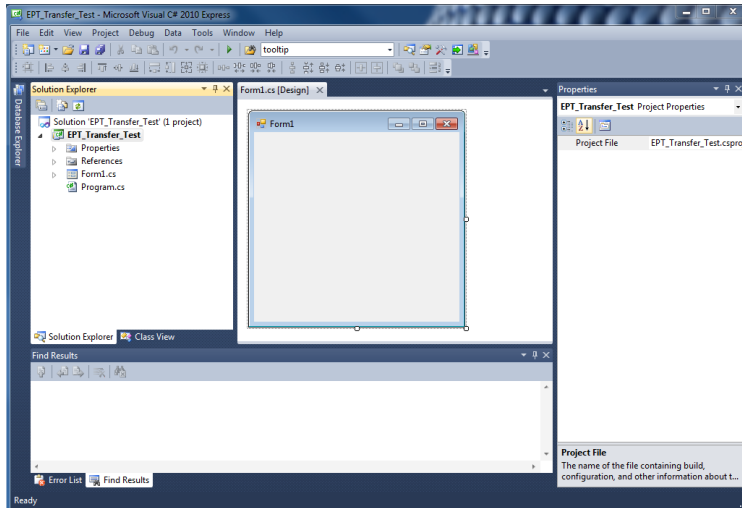


At the New Project window, select the Windows Forms Application. Then, at the Name: box, type in EPT_Transfer_Test

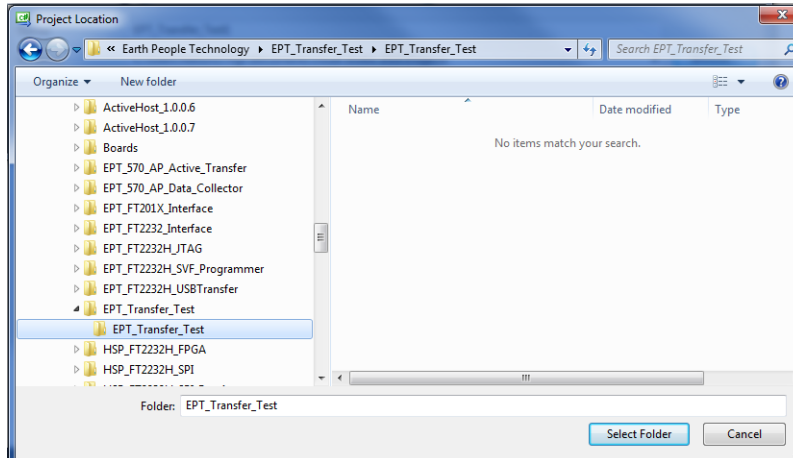


The project creation is complete.

UnoMax CPLD Development System User Manual



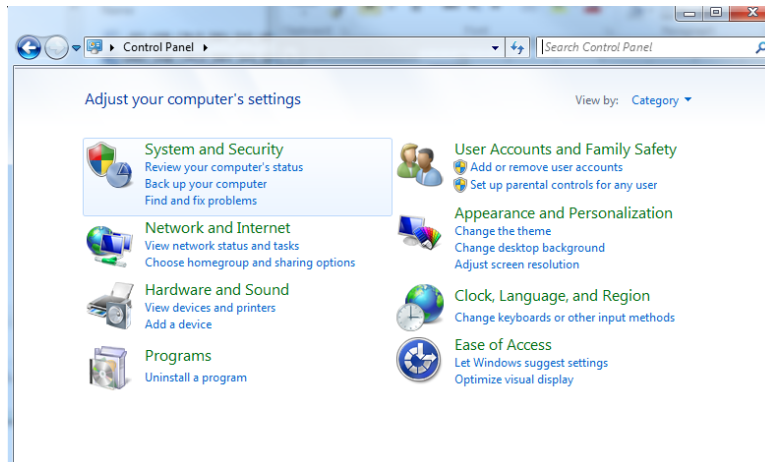
Save the project, go to File->Save as, browse to a folder to create EPT_Transfer_Test folder. The default location is c:\Users\<<Users Name>\documents\visual studio 2010\Projects.



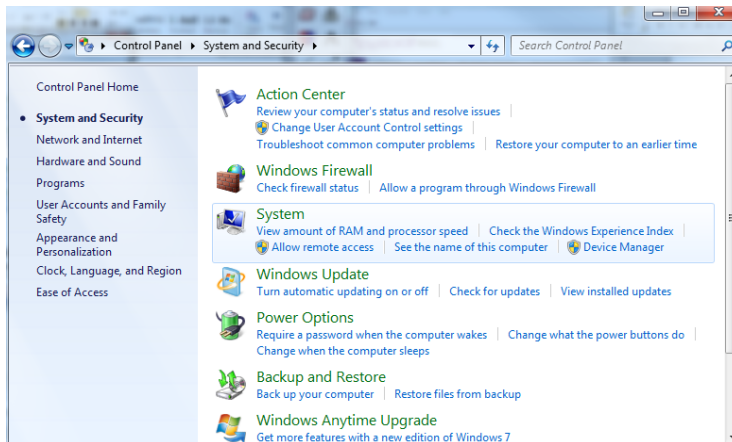
5.5.1 Setting up the C# Express Environment x64 bit

The project environment must be set up correctly in order to produce an application that runs correctly on the target platform. If your system supports 64 bit operation, perform the following steps. Otherwise if your system is 32 bit skip to the Section, Assembling Files into the Project. Visual C# Express defaults to 32 bit operation. If you are unsure if your system supports, you can check it by going to Start->Control Panel->System and Security->System

UnoMax CPLD Development System User Manual

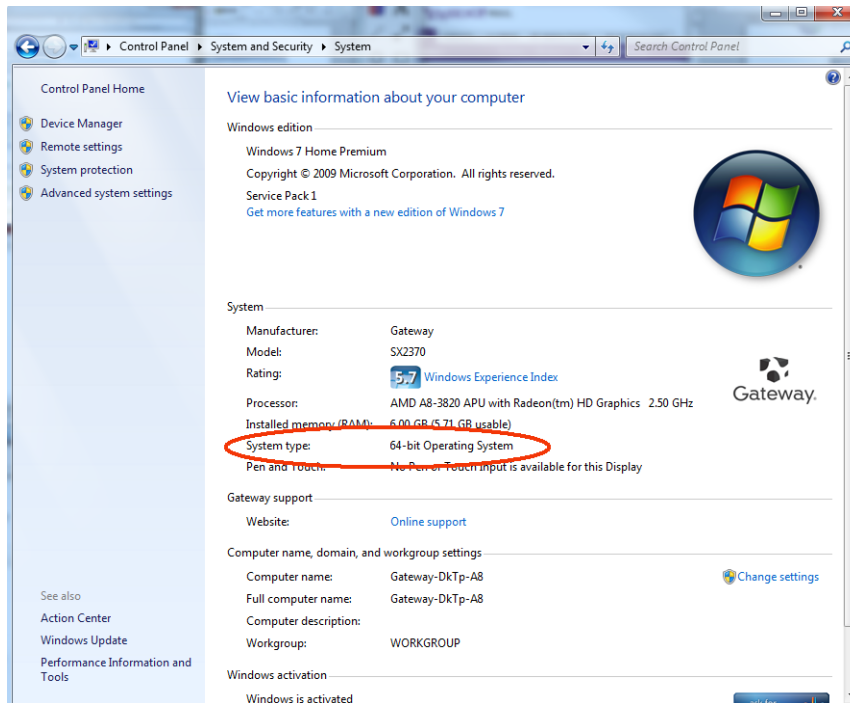


Click on System.

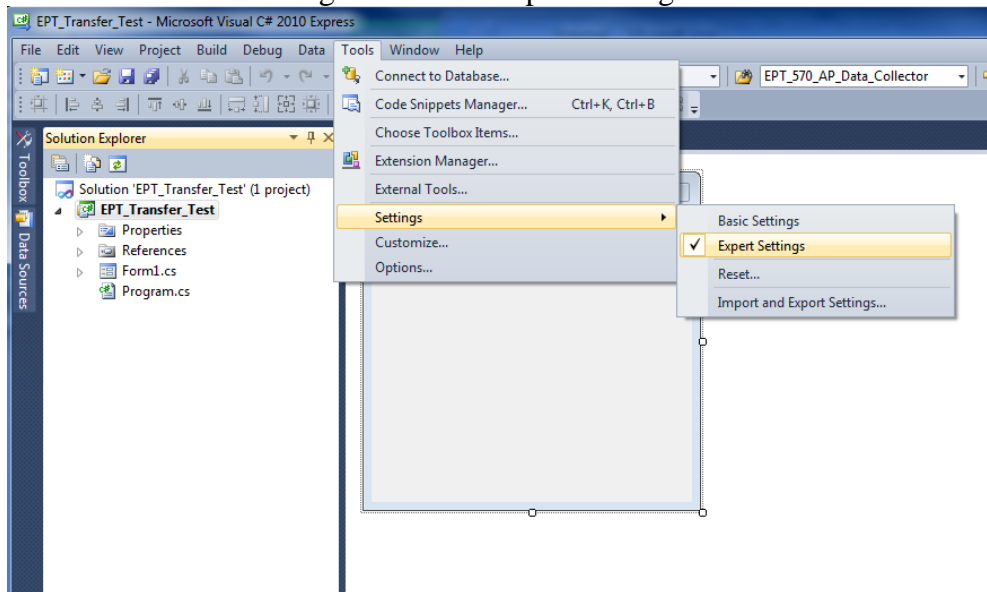


Check under System\System type:

UnoMax CPLD Development System User Manual

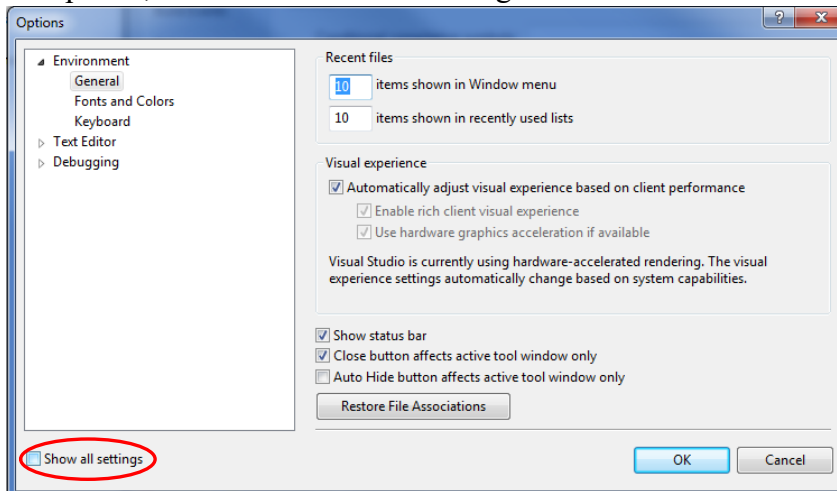


First, we need tell C# Express to produce 64 bit code if we are running on a x64 platform. Go to Tools->Settings and select Expert Settings

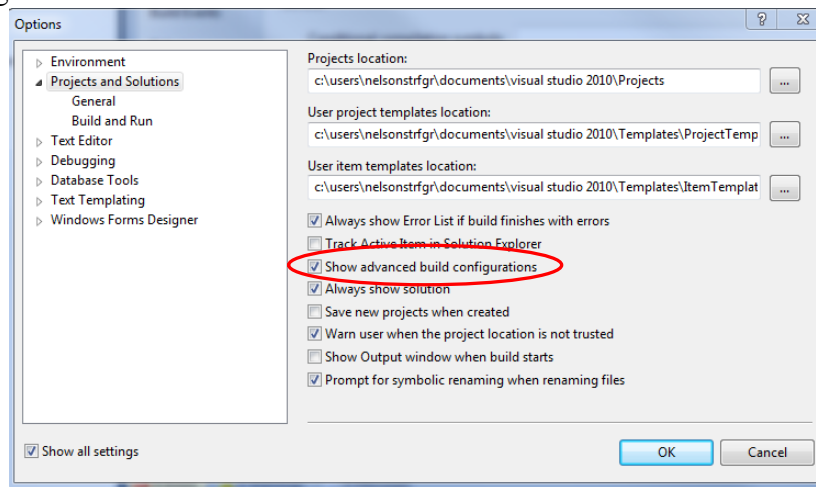


UnoMax CPLD Development System User Manual

Go to Tools->Options, locate the “Show all settings” check box. Check the box.

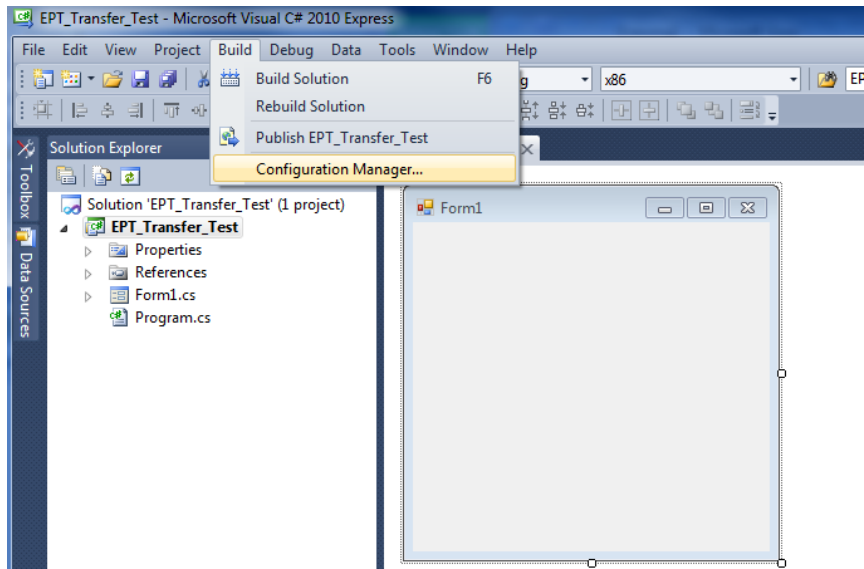


In the window on the left, go to “Projects and Solutions”. Locate the “Show advanced build configurations” check box. Check the box.

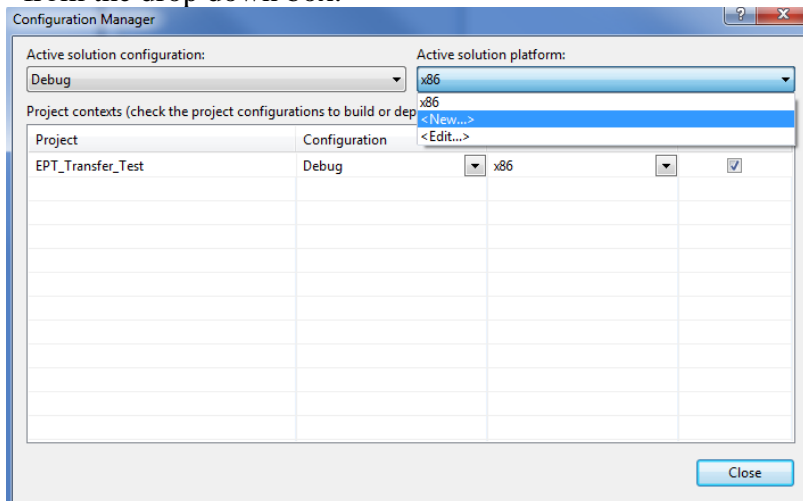


Go to Build->Configuration Manager.

UnoMax CPLD Development System User Manual

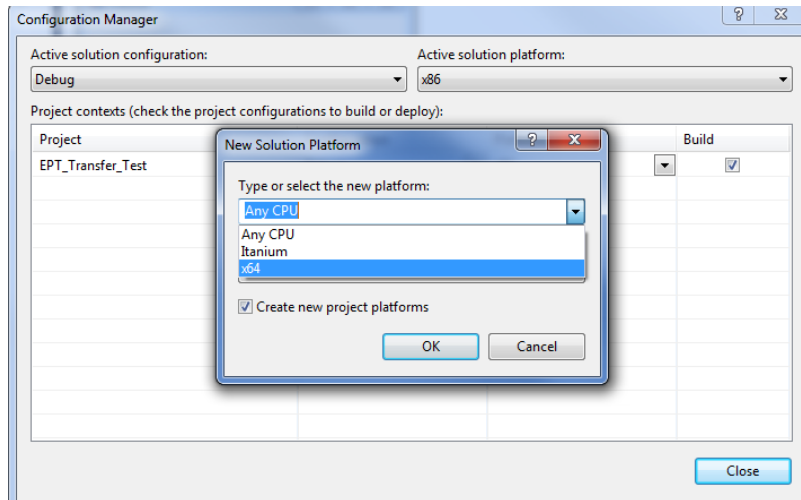


In the Configuration Manager window, locate the “Active solution platform:” label, select “New” from the drop down box.

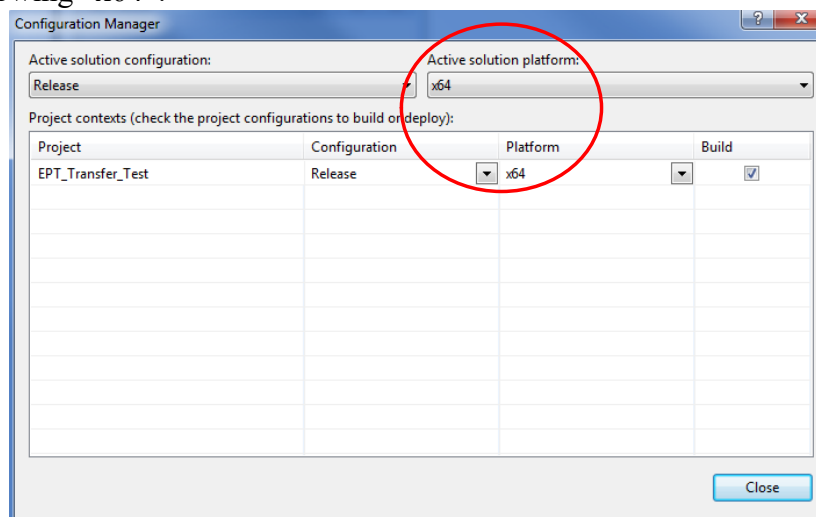


In the New Solution Platform window, click on the drop down box under “Type or select the new platform:”. Select “x64”.

UnoMax CPLD Development System User Manual

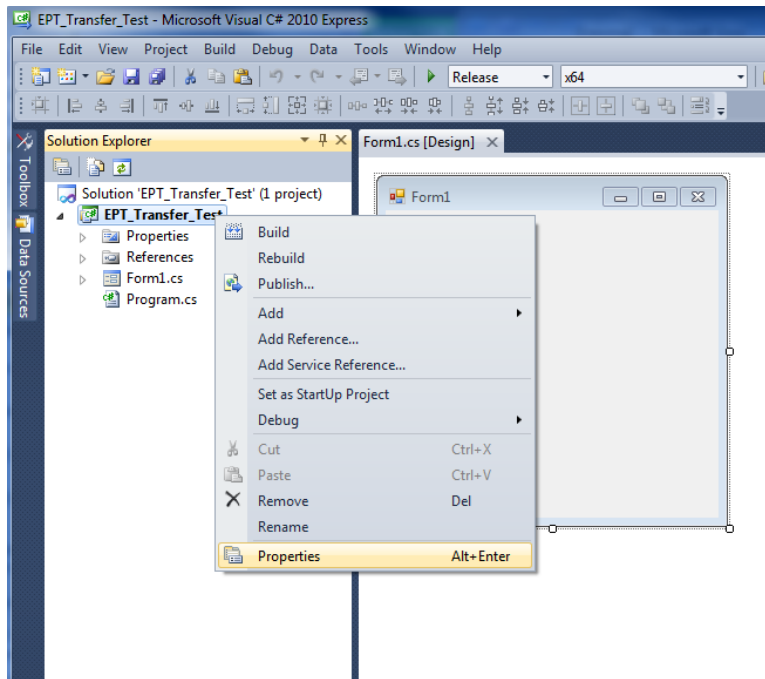


Click the Ok button. Verify that the “Active Solution Platform” and the “Platform” tab are both showing “x64”.

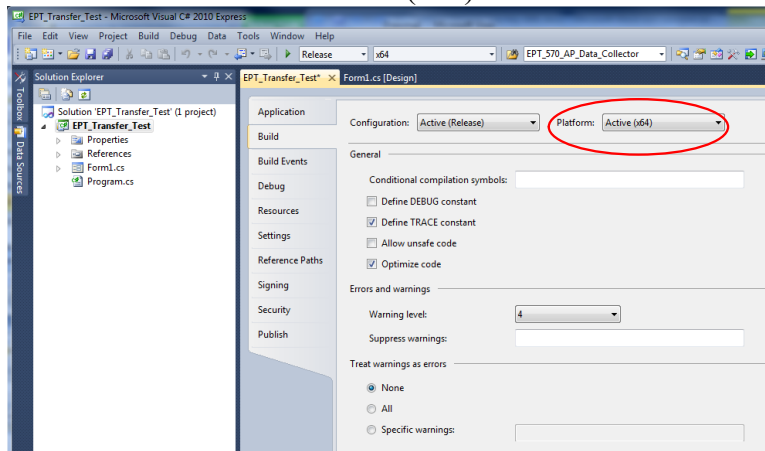


Also, select “Release” under “Active solution configuration”. Click Close. Then, using the Solution Explorer, you can right click on the project, select Properties and click on the Build tab on the right of the properties window.

UnoMax CPLD Development System User Manual



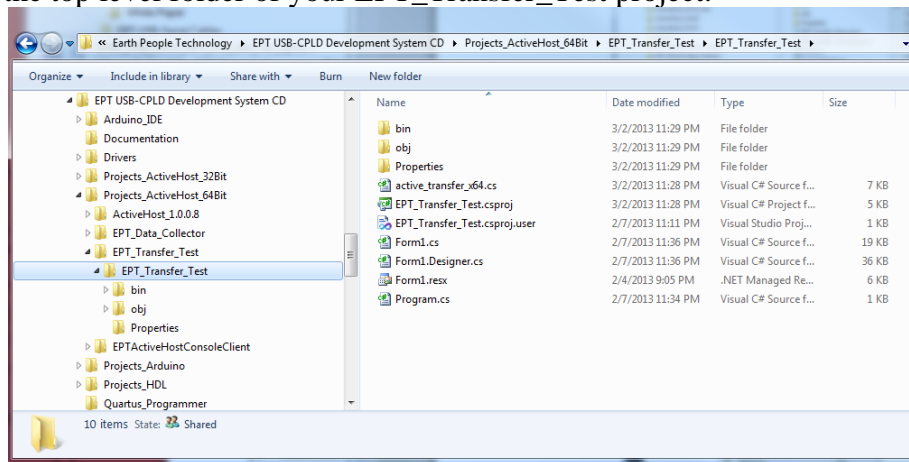
Verify that the “Platform:” label has “Active (x64)” selected from the drop down box.



Click on the Save All button on the tool bar. The project environment is now setup and ready for the project files. Close the Project.

5.6 Assembling Files into the Project

Locate the UnoMax Development System CD installed on your PC. Browse to the EPT_Transfer_Test folder where the Project files reside, copy the*.cs files, and install them in the top level folder of your EPT_Transfer_Test project.



5.6.1 Changing Project Name

NOTE

If you named your project something other than EPT_Transfer_Test, you will have to make changes to the *.cs files above. This is because Visual C# Express links the project files and program files together. These changes can be made by modifying the following:

1. Change namespace of Form1.cs to new project name.
2. Change class of Form1.cs to new project name.
3. Change constructor of Form1.cs to new project name.

UnoMax CPLD Development System User Manual

```

EPT_FT2232_Interface.EPT_FT2232_Interface
using System;
using System.Drawing;
using System.Collections;
using System.Windows.Forms;
using System.Data;
using System.Threading;
using System.Runtime.InteropServices;
using System.Diagnostics;

namespace EPT_FT2232_Interface
{
    public partial class EPT_FT2232_Interface : System.Windows.Forms.Form
    {
        public EPT_FT2232_Interface()
        {
            InitializeComponent();

            for (int i = 0; i < device.Length; ++i)
            {
                device[i] = new Transfer();
            }
        }
    }
}

```

4. Change EPT_Transfer_Test_Load of Form1.cs to new <project name>_Load

```

// Main object loader
private void EPT_FT2232_Interface_Load(object sender, System.EventArgs e)
{
    // Call the List Devices function
    ListDevices();
}

```

5. Change namespace of Form1.Designer.cs to new project name.
6. Change class of Form1.Designer.cs to new project name.

```

EPT_FT2232_Interface.EPT_FT2232_Interface InitializeComponent
namespace EPT_FT2232_Interface
{
    partial class EPT_FT2232_Interface
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }
    }
}

```

7. Change the this.Name and this.Text in Form1Designer.cs to new project name.
8. Change this.Load in Form1Designer.cs to include new project name.

UnoMax CPLD Development System User Manual

```

this.Controls.Add(this.btnTrigger3);
this.Controls.Add(this.btnTrigger2);
this.Controls.Add(this.btnTrigger1);
this.Controls.Add(this.btnCloseDevice);
this.Controls.Add(this.btnOpenDevice);
this.Controls.Add(this.cmbDevList);
this.Controls.Add(this.LEDBox);
this.Controls.Add(this.gbTriggerOut);
this.Controls.Add(this.gbTransferControl);
this.Controls.Add(this.groupBox1);
this.Name = "EPT_FT2232_Interface";
this.Text = "EPT_FT2232_Interface";
this.Load += new System.EventHandler(this.EPT_FT2232_Interface_Load);
this.LEDBox.ResumeLayout(false);
this.LEDBox.PerformLayout();
this.gbTriggerOut.ResumeLayout(false);
this.gbTransferControl.ResumeLayout(false);
this.gbTransferControl.PerformLayout();
this.groupBox1.ResumeLayout(false);
this.groupBox1.PerformLayout();
this.ResumeLayout(false);
this.PerformLayout();

```

9. Change namespace in Program.cs to new project name
10. Change Application.Run() in Program .cs to new projectname.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

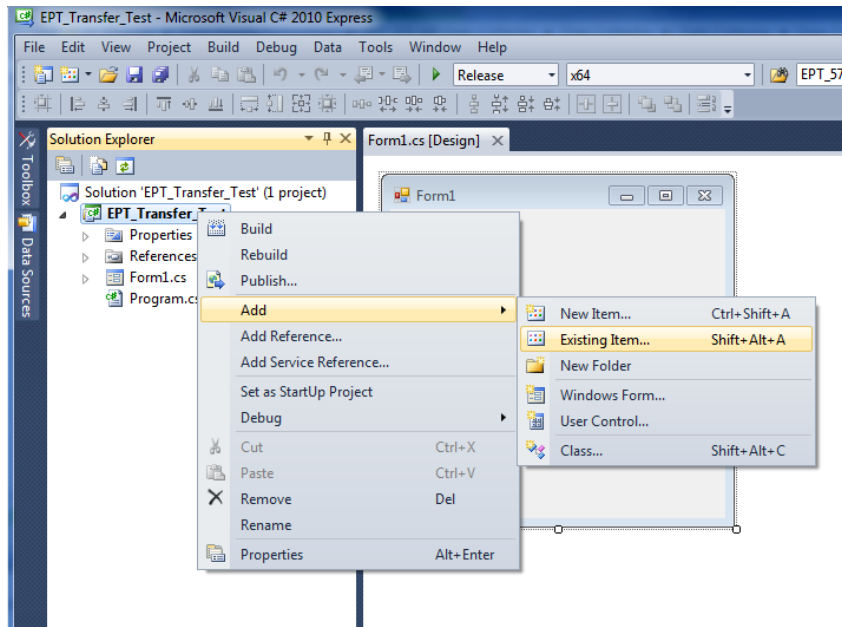
namespace EPT_FT2232_Interface
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new EPT_FT2232_Interface());
        }
    }
}

```

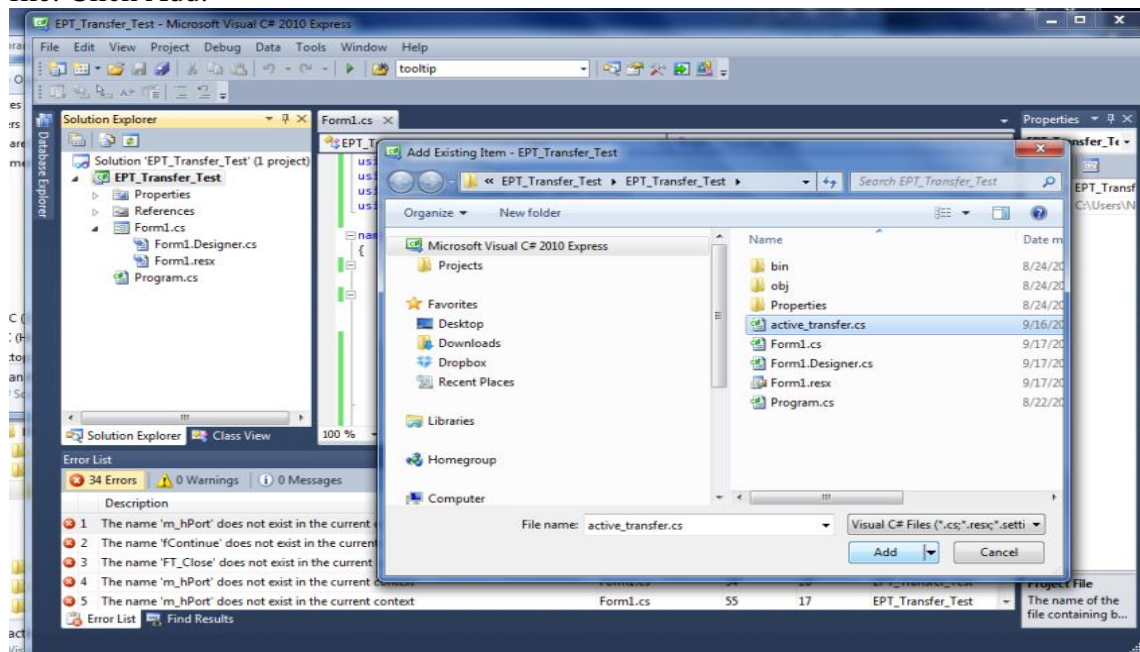
5.6.2 Add Files to Project

Open the EPT_Transfer_Test project. Right click on the project in the Solutions Explorer. Select Add->Existing Item.

UnoMax CPLD Development System User Manual

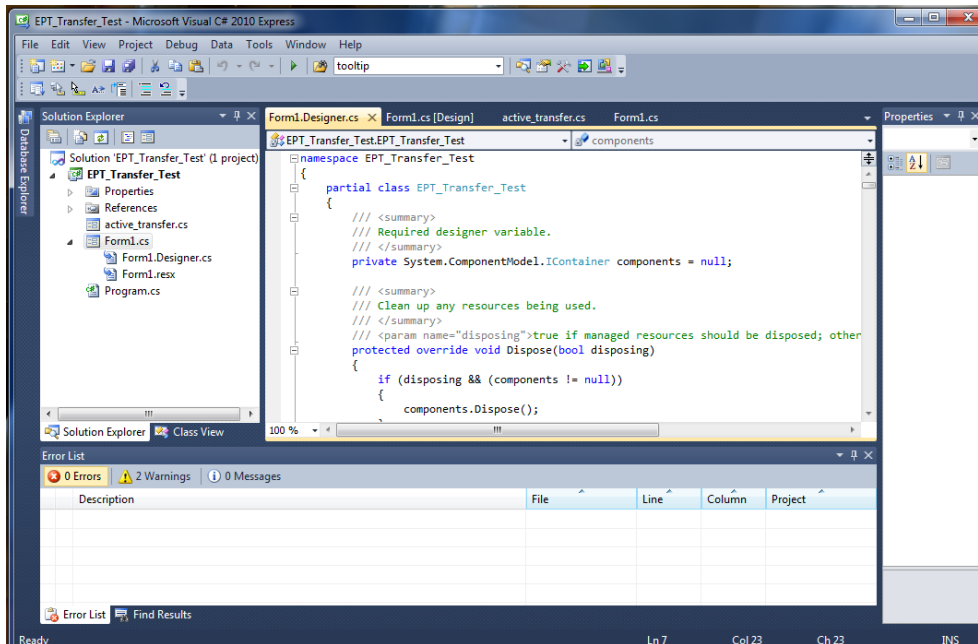


Browse to the EPT_Transfer_Test project folder and select the active_transfer_xx.cs file. Click Add.



UnoMax CPLD Development System User Manual


In the C# Express Solution Explorer, you should be able to browse the files by clicking on them. There should be no errors noted in the Error List box.



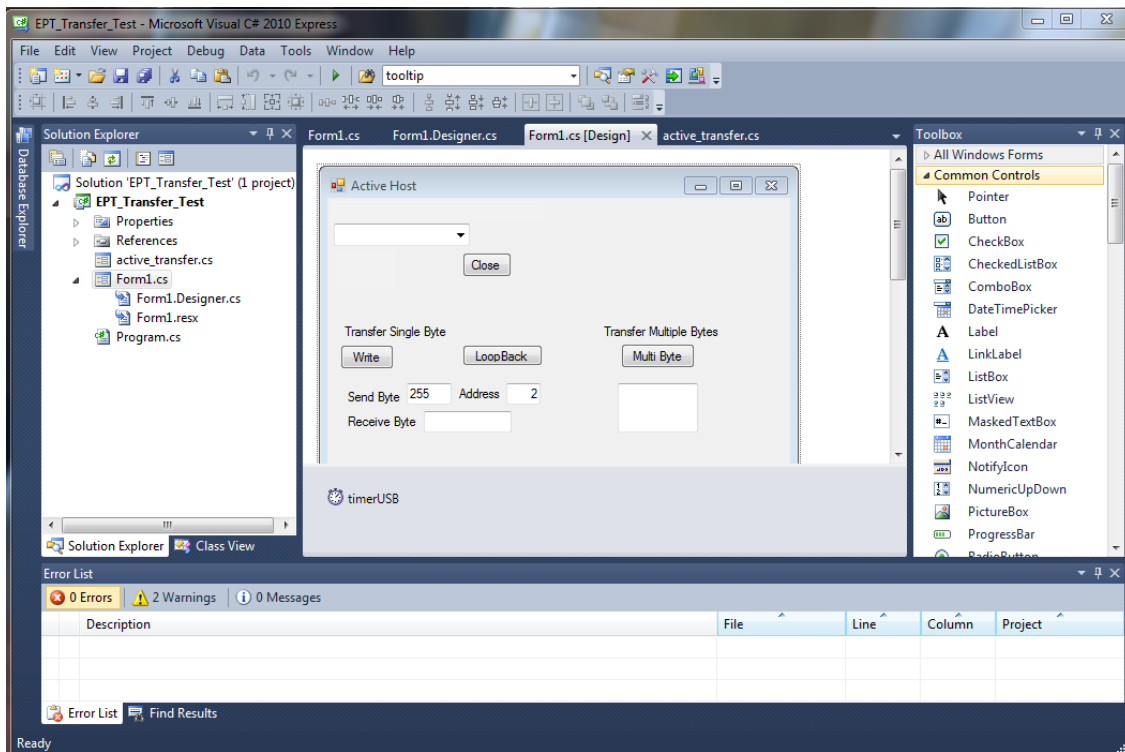
5.6.3 Adding Controls to the Project

Although, the C# language is very similar to C Code, there are a few major differences. The first is C# .NET environment is event based. A second is C# utilizes classes. This guide will keep the details of these items hidden to keep things simple. However, a brief introduction to events and classes will allow the beginner to create effective programs.

Event based programming means the software responds to events created by the user, a timer event, external events such as serial communication into PC, internal events such as the OS, or other events. The events we are concerned with for our example program are user events and the timer event. The user events occur when the user clicks on a button on the Windows Form or selects a radio button. We will add a button to our example program to show how the button adds an event to the Windows Form and a function that gets executed when the event occurs.

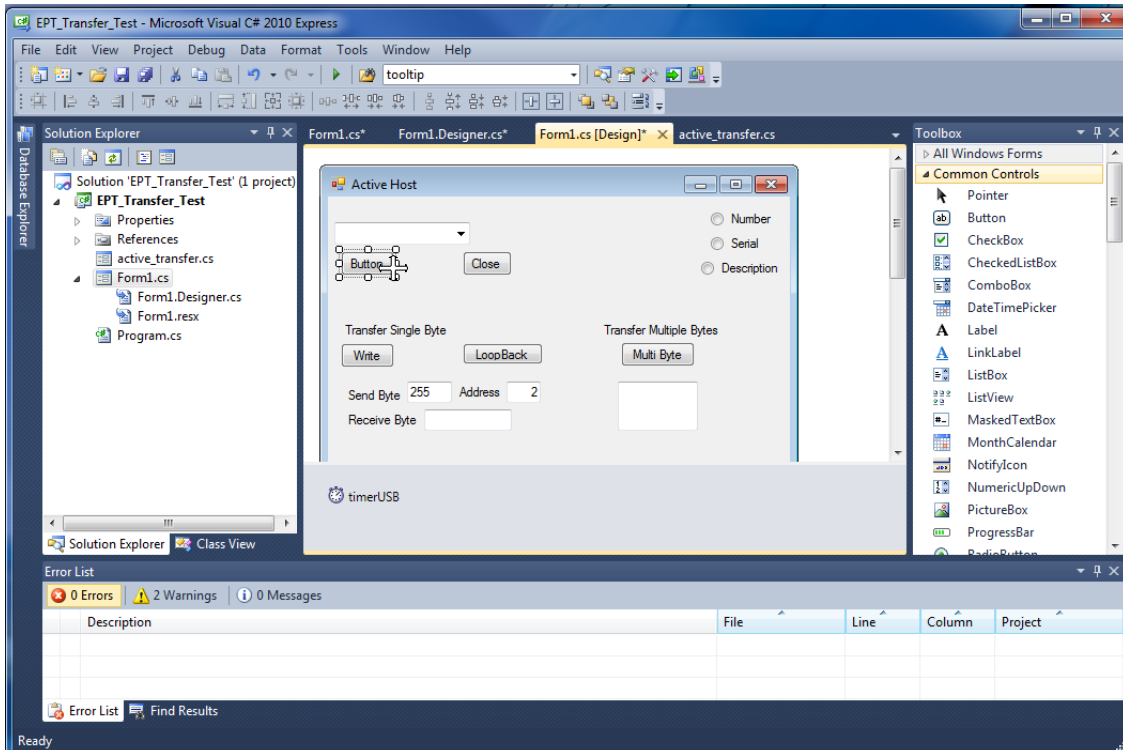
The easiest way to add a button to a form is to double click the Form1.cs in the Solution Explorer. Click on the  button to launch the Toolbox.

UnoMax CPLD Development System User Manual



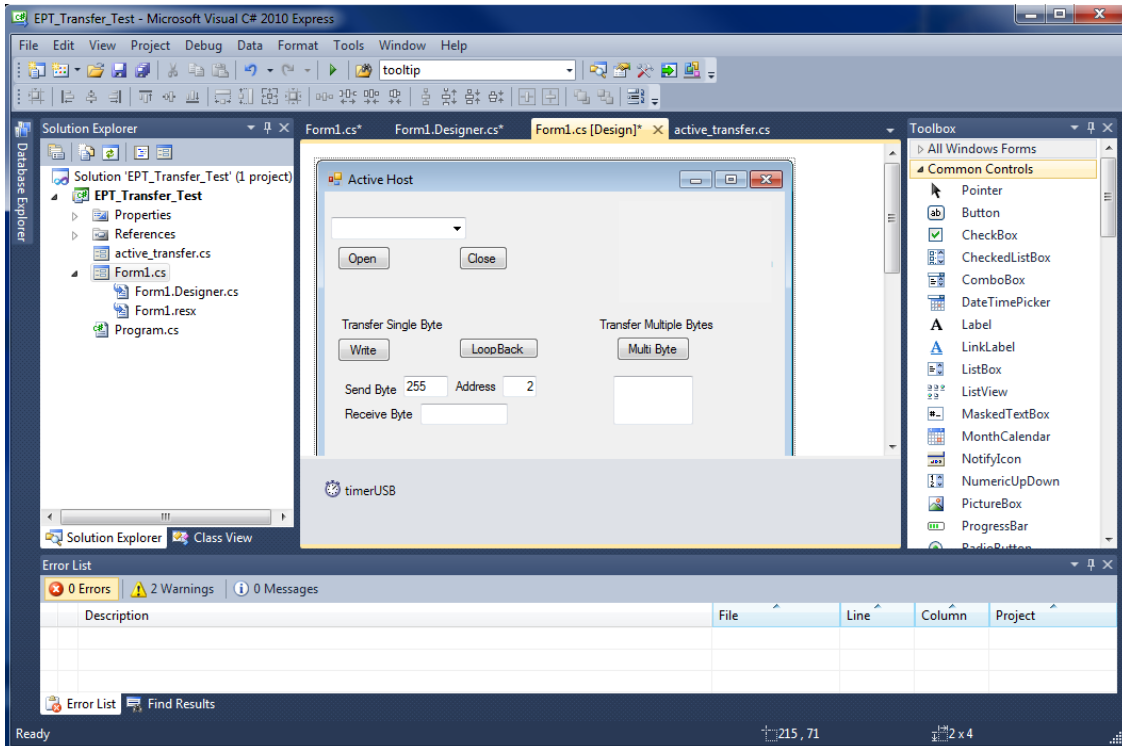
Locate the button on the Toolbox, grab and drag the button onto the Form1.cs [Design] and drop it near the top.

UnoMax CPLD Development System User Manual



Go to the Properties box and locate the (Name) cell. Change the name to “btnOpenDevice”. Locate the Text cell, and change the name to Open.

UnoMax CPLD Development System User Manual



Double click on the Open button. The C# Explorer will automatically switch to the Form1.cs code view. The callback function will be inserted with the name of the button along with “_click” appended to it. The parameter list includes (object sender, System.EventArgs e). These two additions are required for the callback function to initiate when the “click” event occurs.

Private void btnOpenDevice_click(object sender, System.EventArgs e)

There is one more addition to the project files. Double click on the Form1.Designer.cs file in the Solution Explorer. Locate the following section of code.



UnoMax CPLD Development System User Manual

```
//  
// btnOpenDevice  
//  
this.btnOpenDevice.Location = new System.Drawing.Point(240, 13);  
this.btnOpenDevice.Name = "btnOpenDevice";  
this.btnOpenDevice.Size = new System.Drawing.Size(50, 23);  
this.btnOpenDevice.TabIndex = 2;  
this.btnOpenDevice.Text = "Open";  
this.btnOpenDevice.UseVisualStyleBackColor = true;  
this.btnOpenDevice.Click += new System.EventHandler(this.btnOpenDevice_Click);
```

This code sets up the button, size, placement, and text. It also declares the “System.EventHandler()”. This statement sets the click method (which is a member of the button class) of the btnOpenDevice button to call the EventHandler – btnOpenDevice_Click. This is where the magic of the button click event happens.

```
private void btnOpenDevice_Click(object sender, EventArgs e)  
{  
    //Open the Device  
    OpenDevice();  
}  
  
private void btnCloseDevice_Click(object sender, EventArgs e)  
{  
    if (EPT_AH_CloseDeviceByIndex(device_index) != 0)  
    {  
        btnBlkCompare8.Enabled = false;  
        btnBlkCompare16.Enabled = false;  
        btnTrigger1.Enabled = false;  
        btnTrigger2.Enabled = false;  
        btnTrigger3.Enabled = false;  
        btnTrigger4.Enabled = false;  
        btnLEDReset.Enabled = false;  
    }  
    btnOpenDevice.Enabled = true;  
    btnCloseDevice.Enabled = false;  
}
```

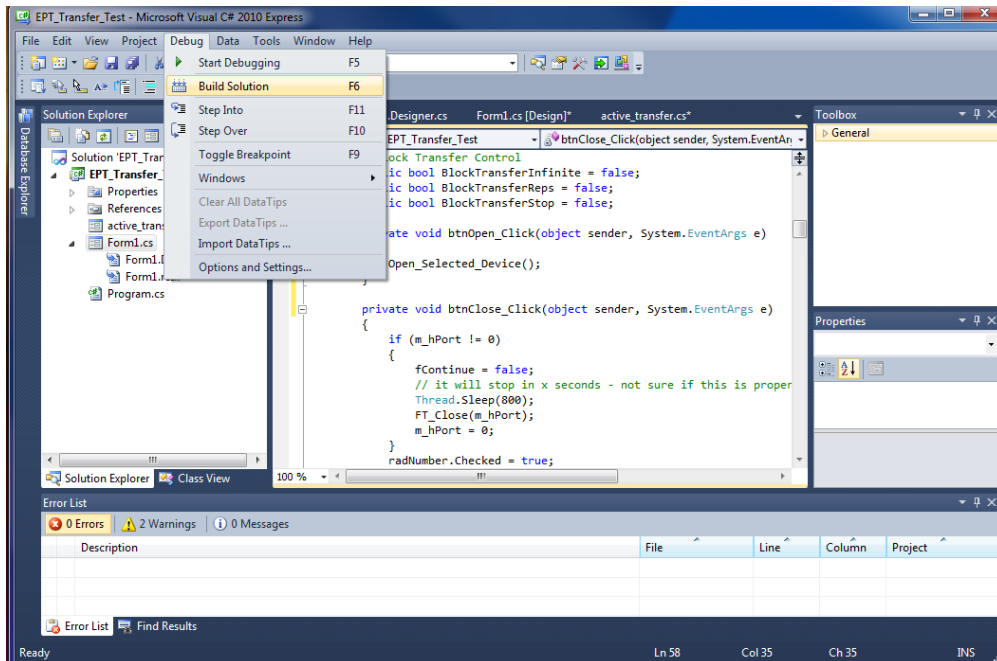
When btnOpenDevice_Click is called, it calls the function “OpenDevice()”. This function is defined in the dll and will connect to the device selected in the combo box. This is a quick view of how to create, add files, and add controls to a C# project. The user is encouraged to spend some time reviewing the online tutorial at <http://www.homeandlearn.co.uk/csharp/csharp.html> to become intimately familiar with Visual C# .NET programming. In the meantime, follow the examples from the Earth

UnoMax CPLD Development System User Manual

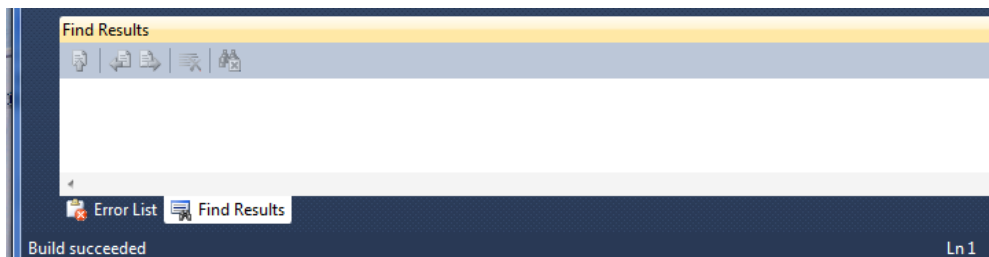
People Technology to perform some simple reads and writes to the EPT USB-CPLD Development System.

5.6.4 Building the Project

Building the EPT_Transfer_Test project will compile the code in the project and produce an executable file. To build the project, go to Debug->Build Solution.



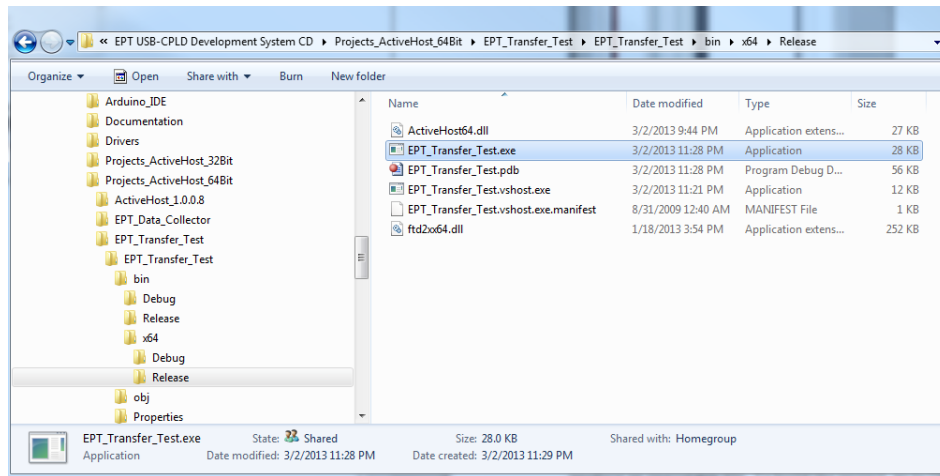
The C# Express compiler will start the building process. If there are no errors with code syntax, function usage, or linking, then the environment responds with “Build Succeeded”.



UnoMax CPLD Development System User Manual

5.6.5 Testing the Project

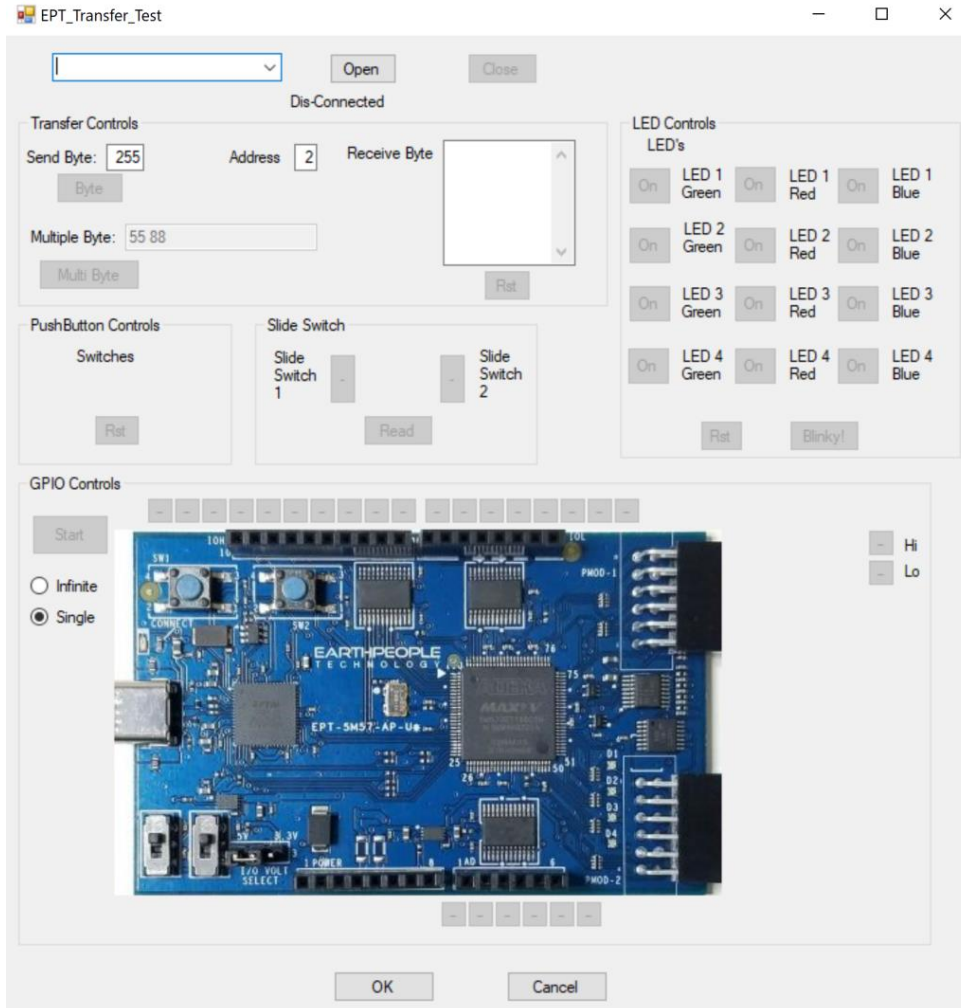
Once the project has been successfully built, it produces an *.exe file. The file will be saved in the Release or Debug folders.



The EPT_Transfer_Text.exe file can now be tested using the UnoMax board. To test the file, connect the UnoMax to the Windows PC using Type A to Type Micro B USB cable. Make sure the driver for the board loads. If the USB driver fails to load, the Windows OS will indicate that no driver was loaded for the device. Go to the folder where the EPT_Transfer_Text.exe file resides, and double click on the file. The application should load with a Windows form.

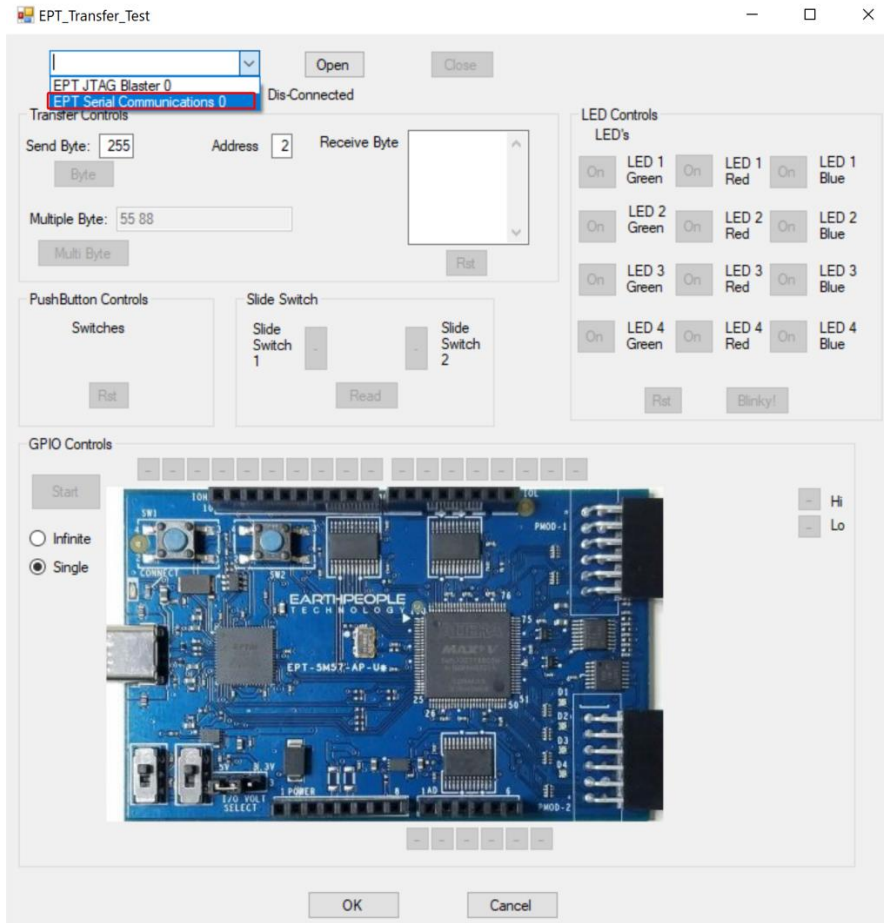


UnoMax CPLD Development System User Manual



With the application loaded, select the USB-CPLD board from the dropdown combo box and click on the “Open” button.

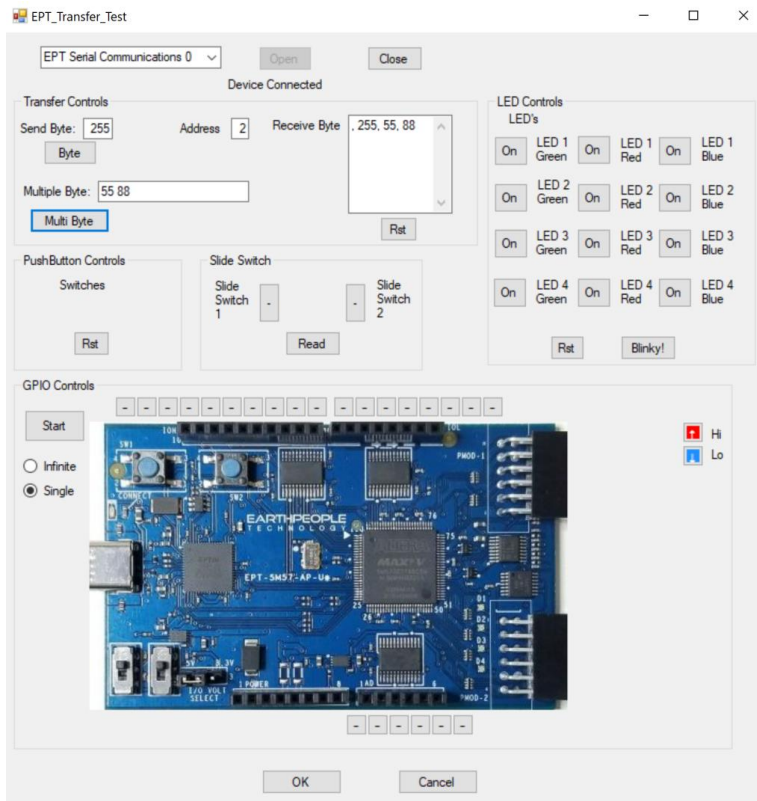
UnoMax CPLD Development System User Manual



Click on one of the LED buttons in the middle of the window. The corresponding LED on the UnoProLogic-U2 board should light up.

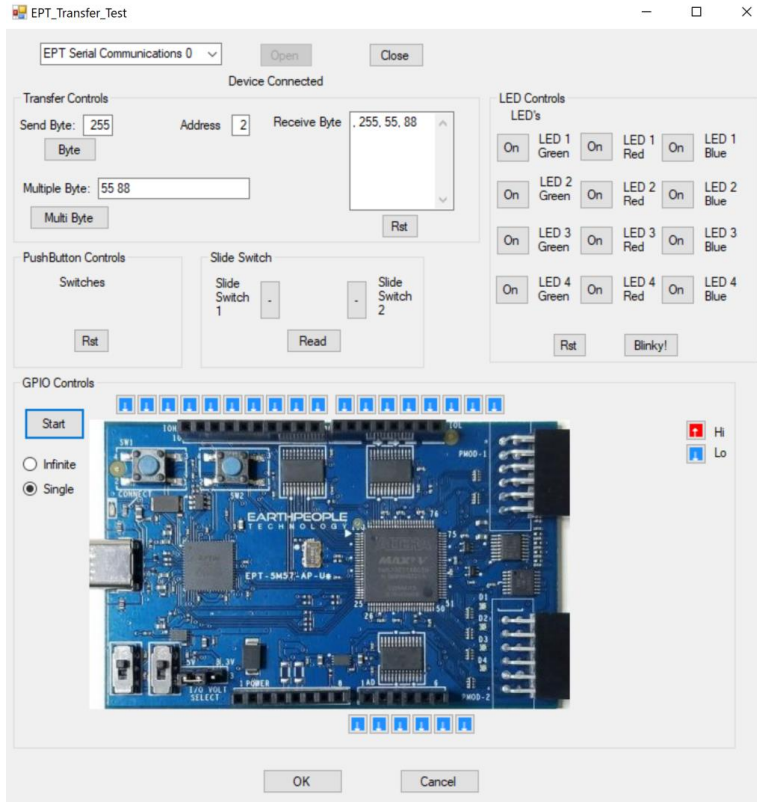
To exercise the Single Byte Transfer EndTerm, click the “Byte” button in the Transfer Controls group. Type in several numbers separated by a space and less 256 into the Multiple Byte textbox. Then hit the Multi Byte button. The numbers appear in the Receive Byte textbox.

UnoMax CPLD Development System User Manual



To exercise the Block Transfer EndTerm, click the “Start” button in the GPIO Controls group. The UnoProLogic will sample the state of each Input pin from around the board and display the result of each pin on the window.

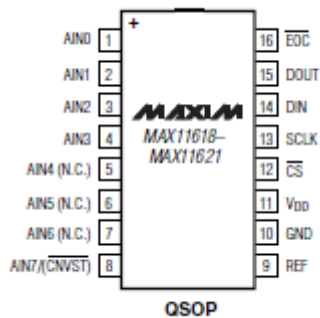
UnoMax CPLD Development System User Manual



Press the PCB switches on the UnoProLogic to view the Switch Controls in action.

6 Using the Analog to Digital Converter

The EPT 5M57-AP-U2 has an onboard Four Channel, 10 Bit, 300 KSamples/second Analog to Digital Converter. It has a serial SPI communications that allow the host to send setup commands and retrieve the sampled data.



PIN	NAME	FUNCTION
1–4	AIN0–AIN3	Analog Inputs
5, 6, 7	N.C.	No Connection
8	CONVST	Active – low Conversion Start Input
9	REF	Reference Input
10	GND	Ground
11	VDD	Power Input
12	CS	Active Low Chip Select Input. When CS is Low the interface is enabled. When CS is high MOSI is high impedance
13	SCLK	Serial Clock input. Clocks data in and out of the serial interface.
14	MISO	Serial Data input. MISO data is latched into the interface on the rising edge of SCLK
15	MOSI	Serial Data Output. Data is clocked out on the falling edge of SCLK. High impedance when CS is connected to VDD.
16	EOC	End of Conversion Output. Data is valide after EOC pulls low.

6.1 Register Descriptions

The MAX11618 communicate between the internal registers and the external circuitry through the SPI-/QSPI-compatible serial interface. Table 1 details the registers and the bit names. Tables 2–5 show the various functions within the conversion register, setup register, averaging register, and reset register.

Table 1. Input Data Byte (MSB First)

REGISTER NAME	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Conversion	1	CHSEL3	CHSEL2	CHSEL1	CHSEL0	SCAN1	SCAN0	X
Setup	0	1	CKSEL1	CKSEL0	REFSEL1	REFSEL0	X	X
Averaging	0	0	1	AVGON	NAV1	NAV0	NSCAN1	NSCAN0
Reset	0	0	0	1	$\overline{\text{RESET}}$	X	X	X

X = Don't care.

6.2 Conversion Register

Select active analog input channels per scan and scan modes by writing to the conversion register. Table 2 details channel selection and the four scan modes. Request a scan by writing to the conversion register when in clock mode 10 or 11, or by applying a low pulse to the CNVST pin when in clock mode 00 or 01.

A conversion is not performed if it is requested on a channel that has been configured as CNVST. Select scan mode 00 or 01 to return one result per single-ended channel within the requested range. Select scan mode 10 to scan a single input channel numerous times, depending on NSCAN1 and NSCAN0 in the averaging register (Table 4). Select scan mode 11 to return only one result from a single channel.

Table 2. Conversion Register*

BIT NAME	BIT	FUNCTION
—	7 (MSB)	Set to 1 to select conversion register.
CHSEL3	6	Analog input channel select.
CHSEL2	5	Analog input channel select.
CHSEL1	4	Analog input channel select.
CHSEL0	3	Analog input channel select.
SCAN1	2	Scan mode select.
SCAN0	1	Scan mode select.
—	0 (LSB)	Don't care.

UnoMax CPLD Development System User Manual

CHSEL1	CHSEL0	SELECTED CHANNEL (N)
0	0	AIN0
0	1	AIN1
1	0	AIN2
1	1	AIN3

SCAN1	SCAN0	SCAN MODE (CHANNEL N IS SELECTED BY BITS CHSEL3–CHSEL0)
0	0	Scans channels 0 through N.
0	1	Scans channels N through the highest numbered channel.
1	0	Scans channel N repeatedly. The averaging register sets the number of results.
1	1	No scan. Converts channel N once only.

6.3 Setup Register

Write a byte to the setup register to configure the clock, reference, and power-down modes. Table 3 details the bits in the setup register. Bits 5 and 4 (CKSEL1 and CKSEL0) control the clock mode, acquisition and sampling, and the conversion start. Bits 3 and 2 (REFSEL1 and REFSEL0) control internal or external reference use.

UnoMax CPLD Development System User Manual

Table 3. Setup Register*

BIT NAME	BIT	FUNCTION
—	7 (MSB)	Set to zero to select setup register.
—	6	Set to 1 to select setup register.
CKSEL1	5	Clock mode and $\overline{\text{CNVST}}$ configuration. Resets to 1 at power-up.
CKSELO	4	Clock mode and $\overline{\text{CNVST}}$ configuration.
REFSEL1	3	Reference mode configuration.
REFSELO	2	Reference mode configuration.
—	1	Don't care.
—	0 (LSB)	Don't care.

*See below for bit details.

CKSEL1	CKSELO	CONVERSION CLOCK	ACQUISITION/SAMPLING	$\overline{\text{CNVST}}$ CONFIGURATION
0	0	Internal	Internally timed	$\overline{\text{CNVST}}$
0	1	Internal	Externally timed through $\overline{\text{CNVST}}$	$\overline{\text{CNVST}}$
1	0	Internal	Internally timed	AIN15/AIN11/AIN7**
1	1	External (4.8MHz max)	Externally timed through SCLK	AIN15/AIN11/AIN7**

**For the MAX11618/MAX11619, $\overline{\text{CNVST}}$ has its own dedicated pin.

REFSEL1	REFSELO	VOLTAGE REFERENCE	AutoShutdown
0	0	Internal	Reference off after scan; need wake-up delay.
0	1	External single ended	Reference off; no wake-up delay.
1	0	Internal	Reference always on; no wake-up delay.
1	1	Reserved	Reserved. Do not use.

6.4 Averaging Register

Write to the averaging register to configure the ADC to average up to 32 samples for each requested result, and to independently control the number of results requested for single-channel scans. Table 2 details the four scan modes available in the conversion register. All four scan modes allow averaging as long as the AVGO bit, bit 4 in the averaging register, is set to 1. Select scan mode 10 to scan the same channel multiple times. Clock mode 11 disables averaging.

UnoMax CPLD Development System User Manual

Table 4. Averaging Register*

BIT NAME	BIT	FUNCTION
—	7 (MSB)	Set to 0 to select averaging register.
—	6	Set to 0 to select averaging register.
—	5	Set to 1 to select averaging register.
AVGON	4	Set to 1 to turn averaging on. Set to zero to turn averaging off.
NAV1	3	Configures the number of conversions for single-channel scans.
NAV0	2	Configures the number of conversions for single-channel scans.
NSCAN1	1	Single-channel scan count. (Scan mode 10 only.)
NSCAN0	0 (LSB)	Single-channel scan count. (Scan mode 10 only.)

*See below for bit details.

AVGON	NAV1	NAV0	FUNCTION
0	X	X	Performs 1 conversion for each requested result.
1	0	0	Performs 4 conversions and returns the average for each requested result.
1	0	1	Performs 8 conversions and returns the average for each requested result.
1	1	0	Performs 16 conversions and returns the average for each requested result.
1	1	1	Performs 32 conversions and returns the average for each requested result.

X = Don't care.

NSCAN1	NSCAN0	FUNCTION (APPLIES ONLY IF SCAN MODE 10 IS SELECTED)
0	0	Scans channel N and returns 4 results.
0	1	Scans channel N and returns 8 results.
1	0	Scans channel N and returns 12 results.
1	1	Scans channel N and returns 16 results.

6.5 Reset Register

Write to the reset register (as shown in Table 5) to clear the FIFO or to reset all registers to their default states. Set the RESET bit to 1 to reset the FIFO. Set the reset bit to zero to return the MAX11618 to the default power-up state.



Table 5. Reset Register

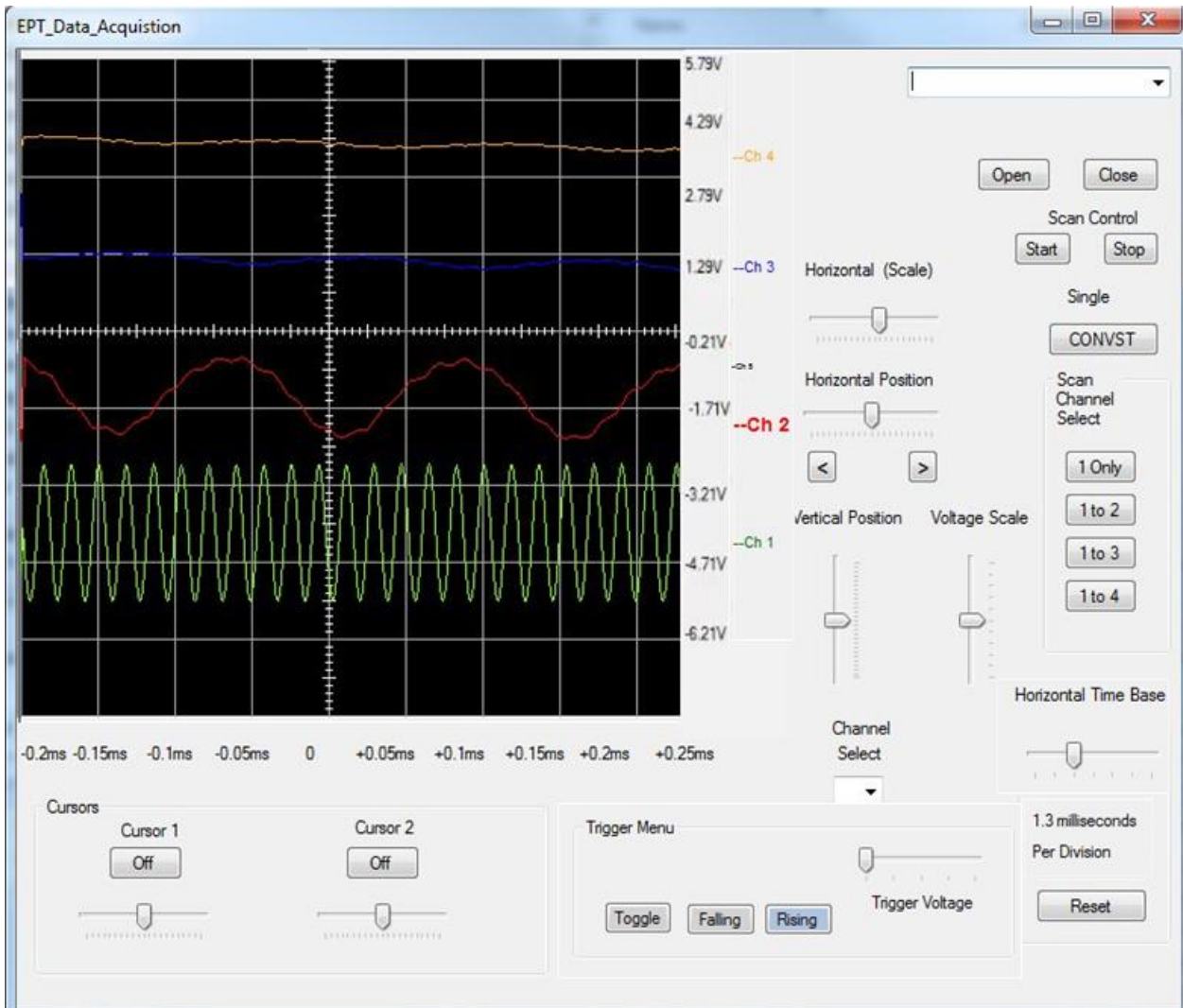
BIT NAME	BIT	FUNCTION
—	7 (MSB)	Set to 0 to select reset register.
—	6	Set to 0 to select reset register.
—	5	Set to 0 to select reset register.
—	4	Set to 1 to select reset register.
RESET	3	Set to zero to reset all registers. Set to 1 to clear the FIFO only.
X	2	Don't care.
X	1	Don't care.
X	0 (LSB)	Don't care.

7 The UnoProLyzer Application

The source DVD for the UnoMax2 comes with the UnoProLyzer application project. This project allows the user to display 1 to 4 channels of analog input in a graphing application. The project utilizes the PC to perform all data storage and graphing. The PC sends commands to and receives the data from the UnoMax2 and stores each channel data in its own separate buffer in memory. The UnoProLyzer collects all samples from each channel by streaming across up to four dedicated communication “pipes”.

The UnoMax2 commands the ADC to start a conversion on the channels selected by the user. It then waits for the ADC to complete the conversion on all channels. It transfers the data for each channel across its own dedicated communication pipe. Then starts the process over again. The UnoProLyzer application will accept each data word and decode the pipe number it came across. It stores each word into a separate buffer for each channel. The UnoProLyzer then performs post processing on each data word. It performs trigger detection, smoothing, sorting, scaling and searching. It then displays the data set in 500 data point segments.

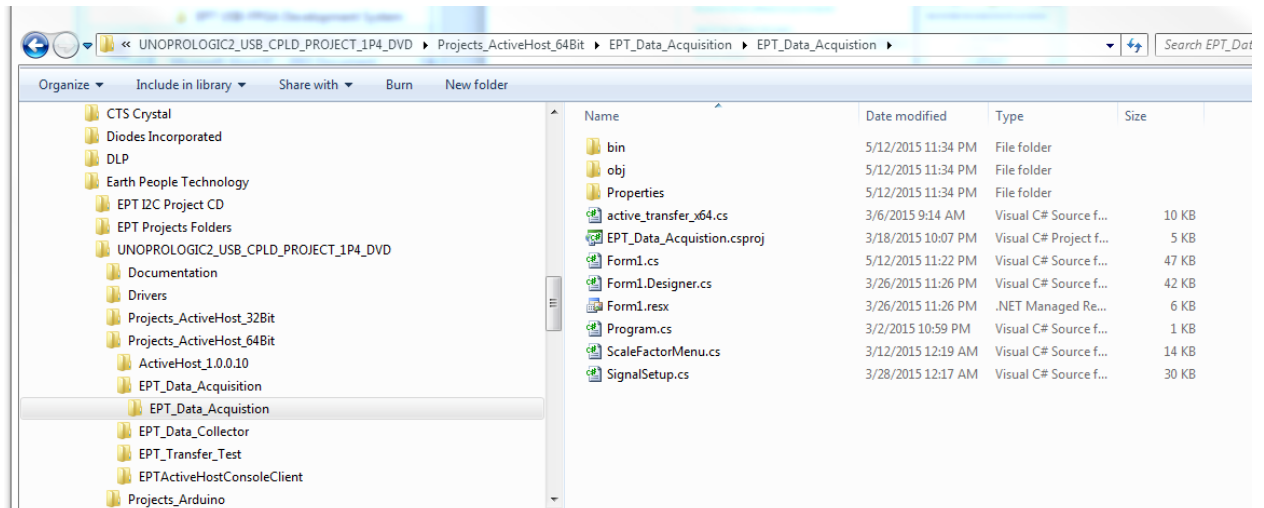
UnoMax CPLD Development System User Manual



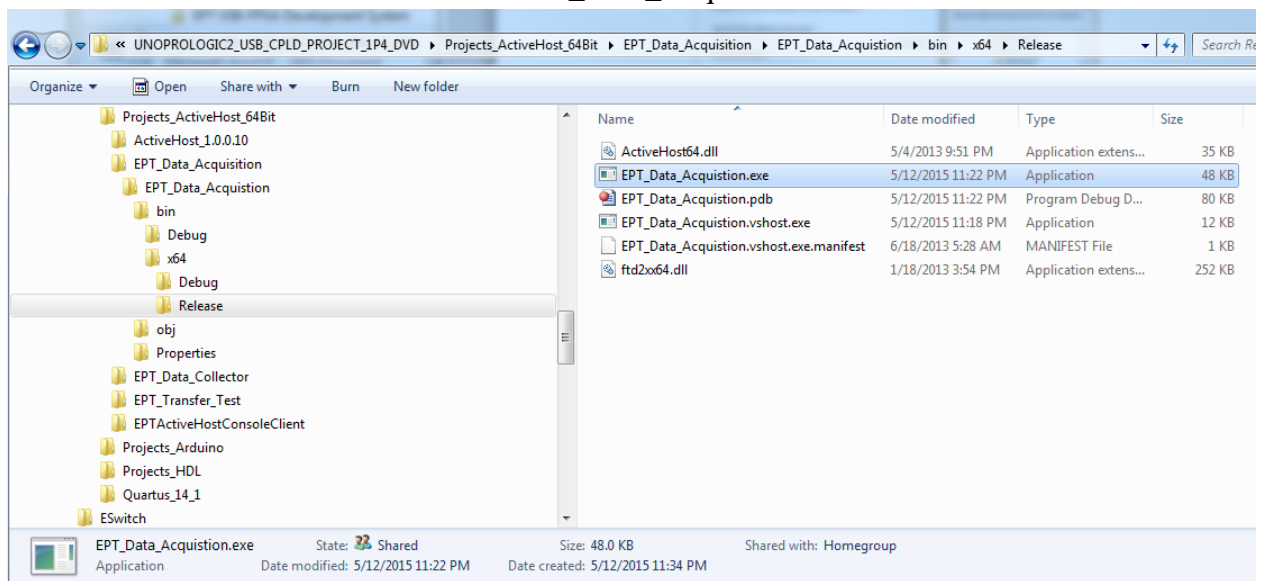
7.1 Accessing the UnoProLyzer Application

Locate the EPT_Data_Acquisition folder in the Drivers folder of the EPT USB-CPLD Development System DVD using Windows Explorer.

UnoMax CPLD Development System User Manual

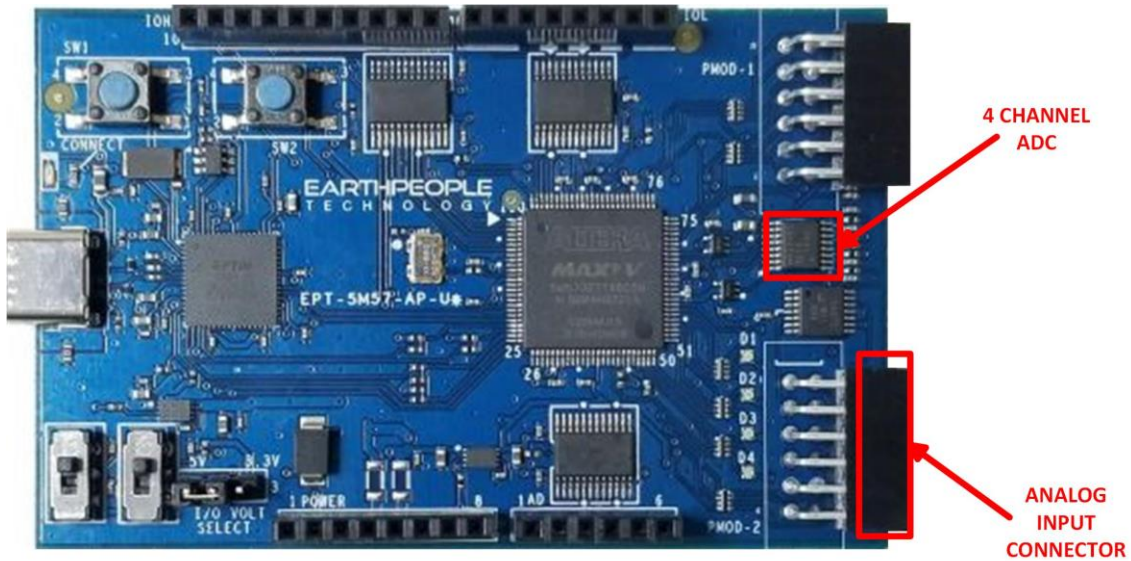


Go to the Release folder and locate the EPT_Data_Acquisition.exe file.



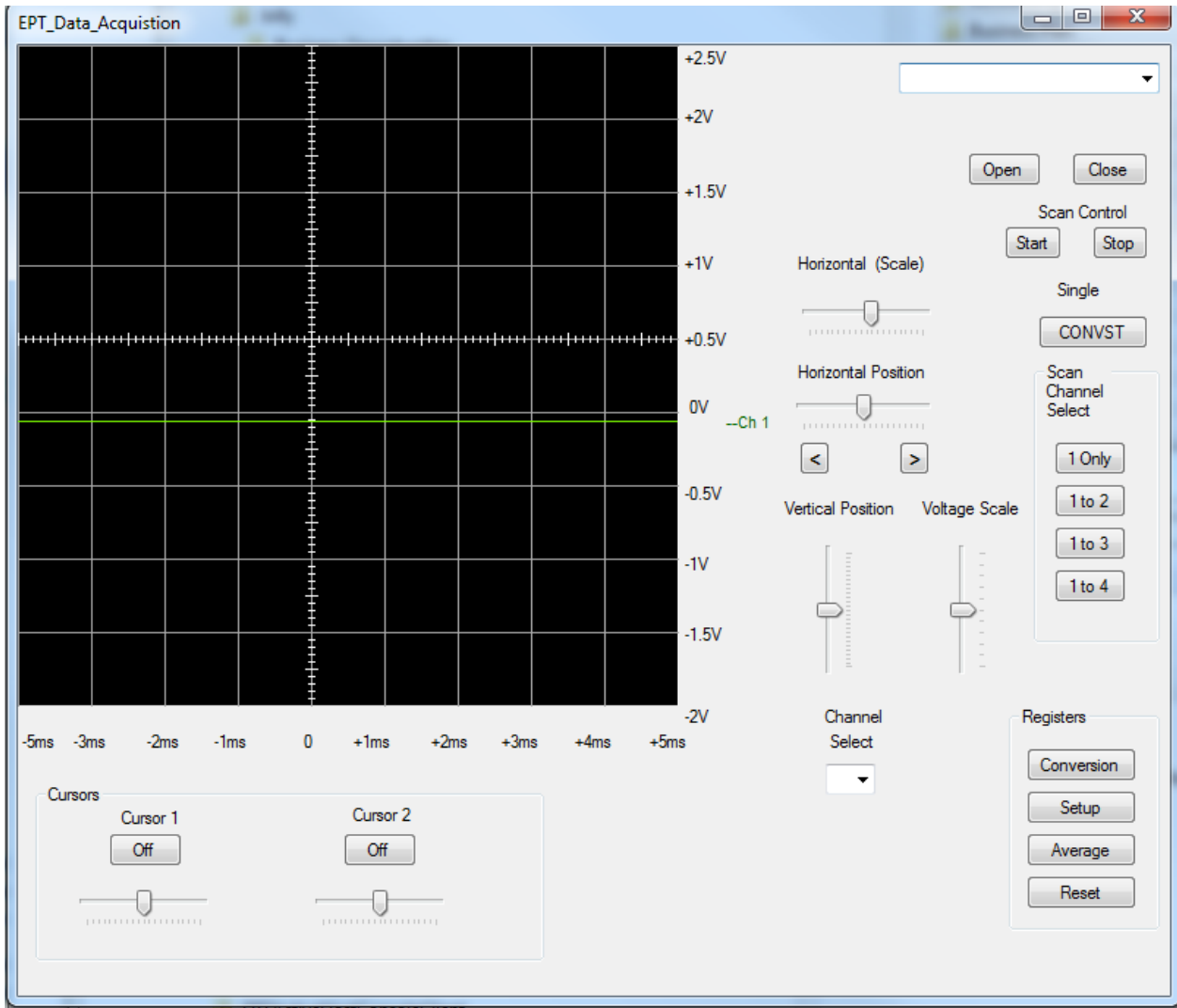
Make sure the UnoMax2 is installed and the USB driver has been loaded.

UnoMax CPLD Development System User Manual



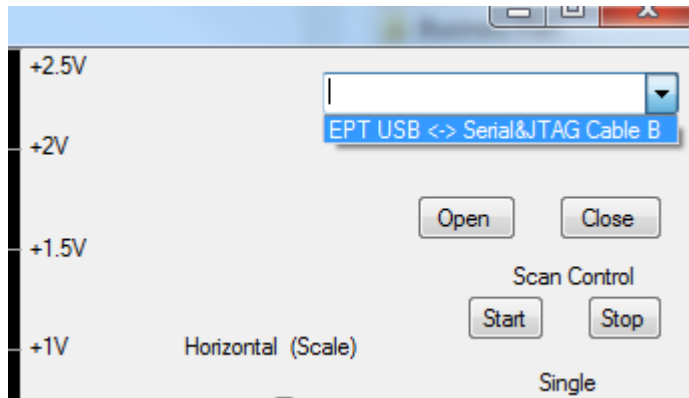
Double click on the EPT_Data_Acquisition.exe file and the UnoProLyzer Application will open.

UnoMax CPLD Development System User Manual

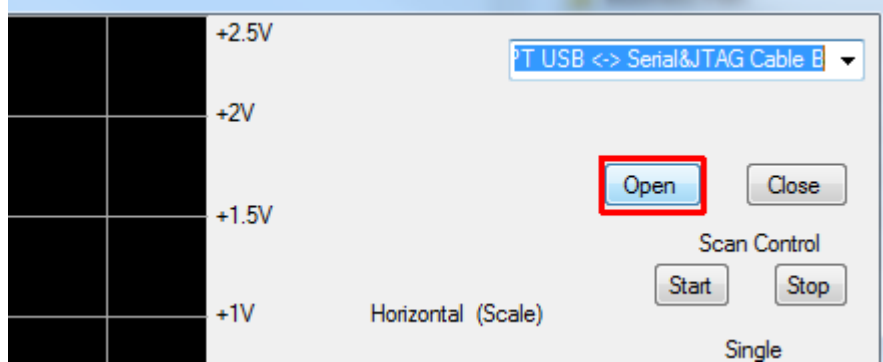


Go to the upper right of the window and click on the drop down box.

UnoMax CPLD Development System User Manual

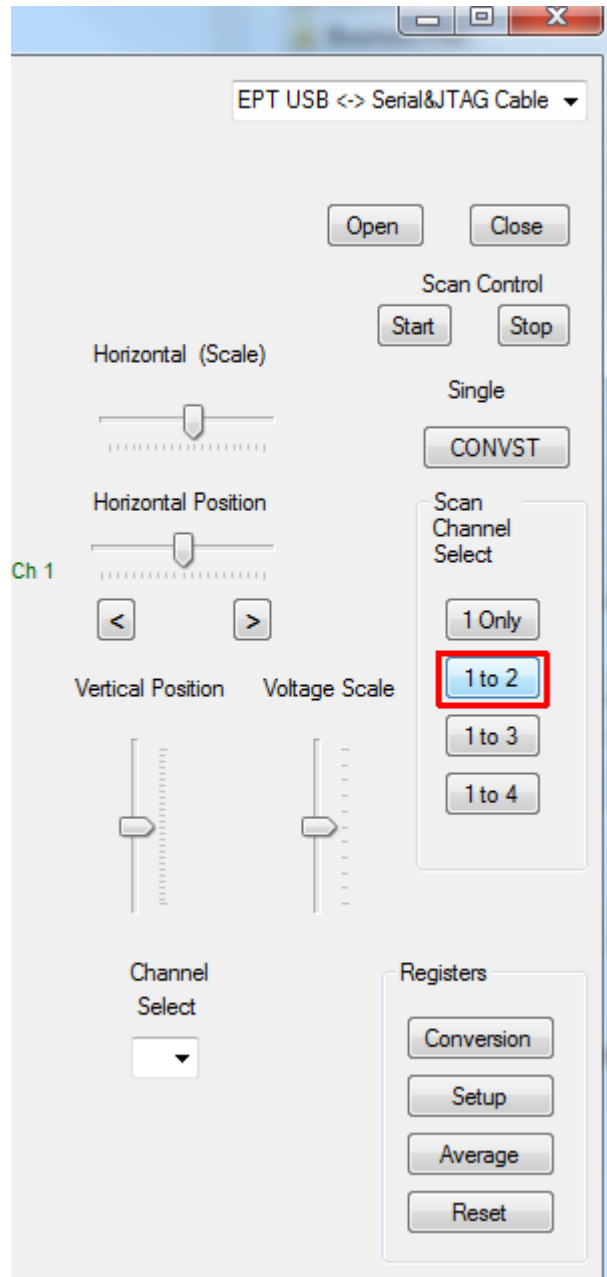


Select the “EPT USB<-> Serial&JTAG Cable B. Then click on the Open button



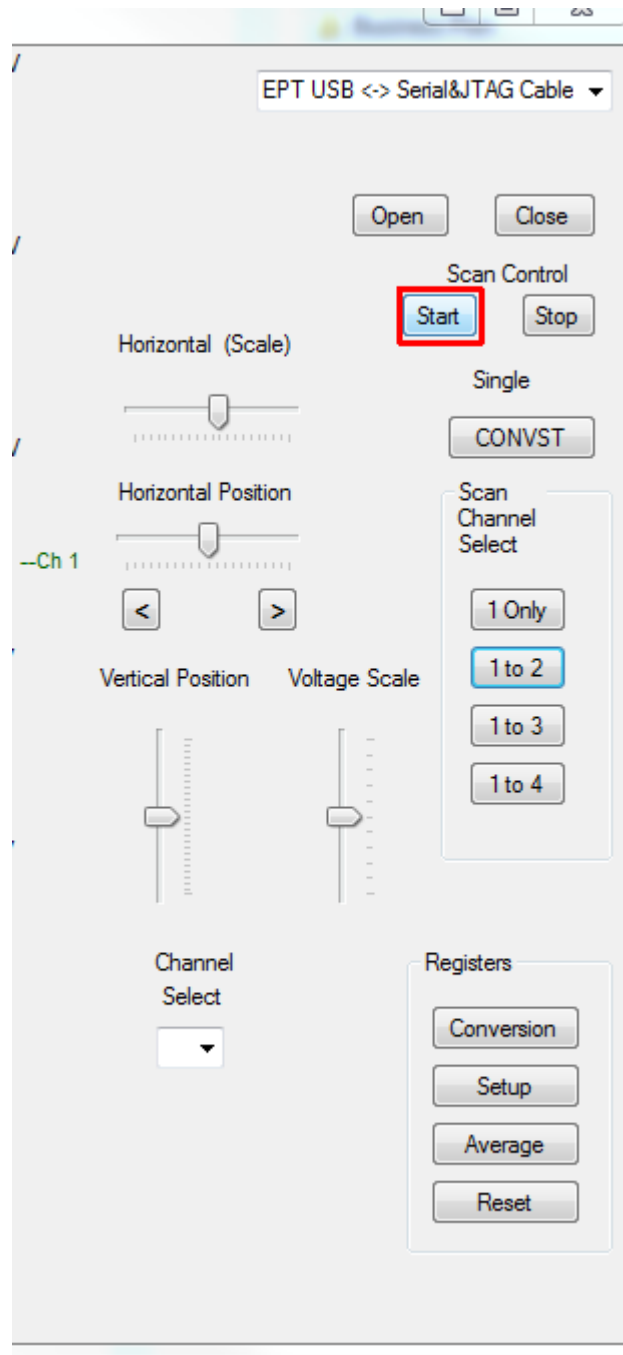
Next, select the number of channels to display. The channels have to be selected in sequential order, you cannot pick out single channel (except for channel 1). So for two channels, click on the “1 to 2” button. This will display the data from both channel 1 and 2.

UnoMax CPLD Development System User Manual



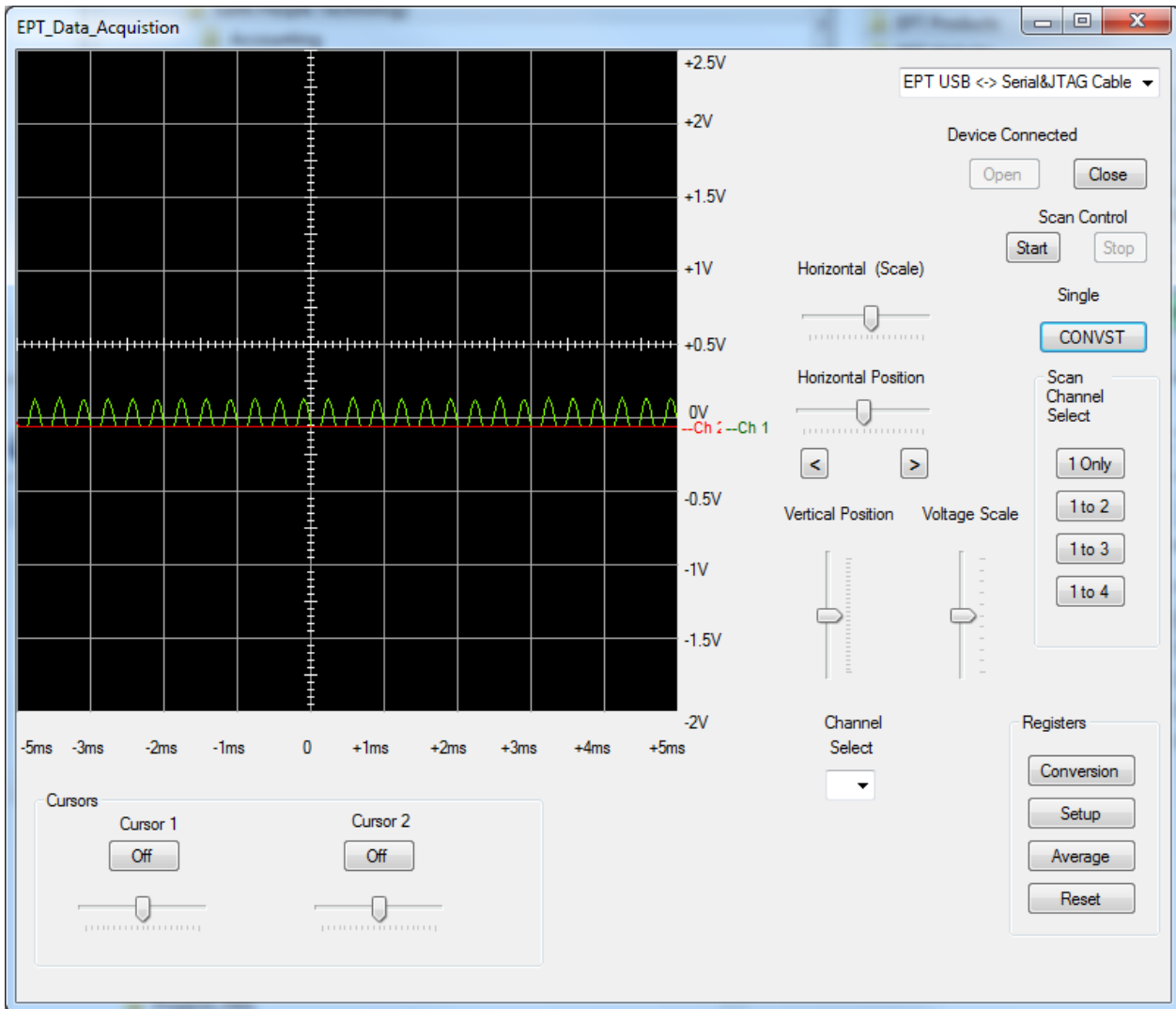
Next, click on the Start button.

UnoMax CPLD Development System User Manual



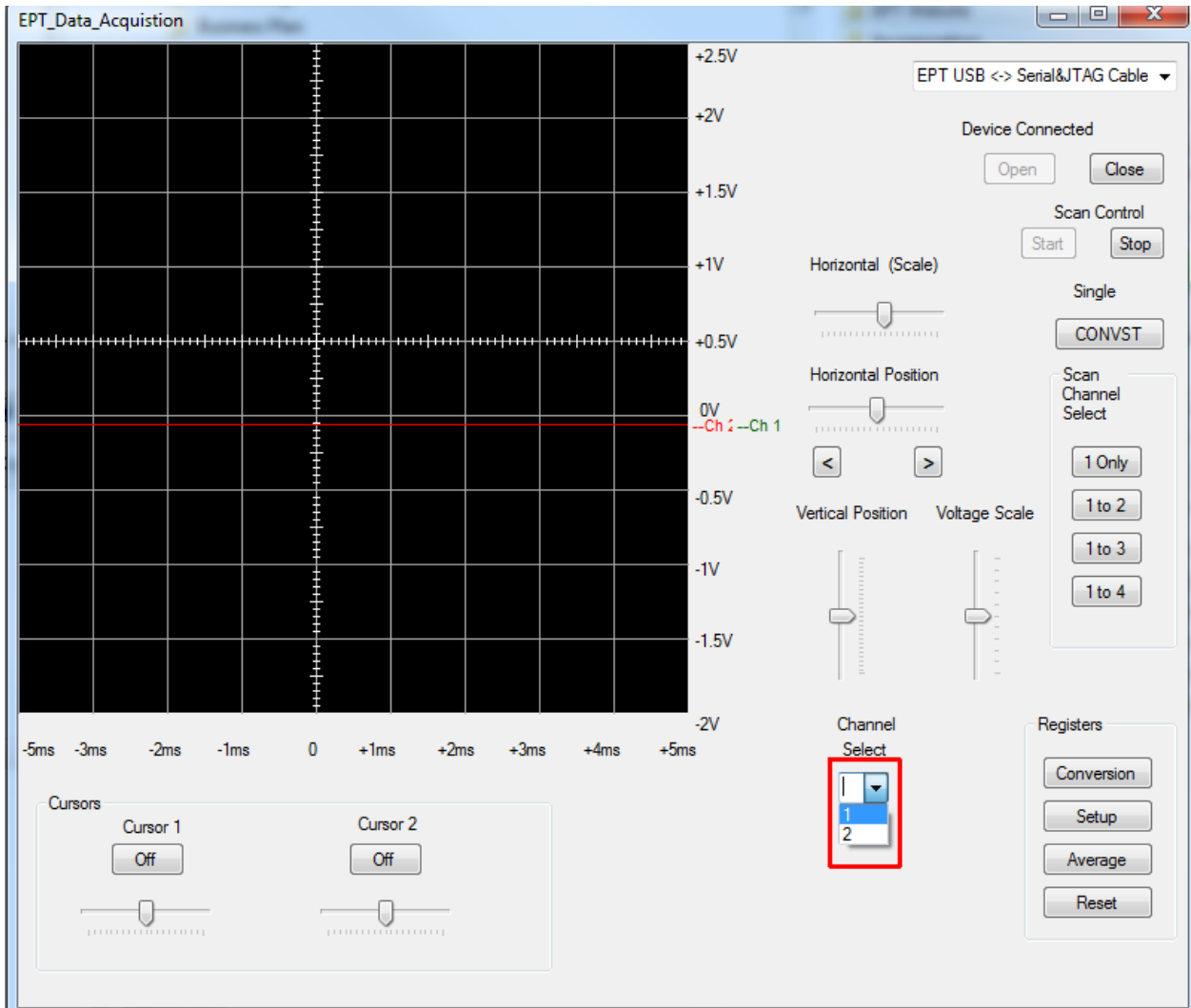
The data from the two channels will appear at the same latitude on the graph.

UnoMax CPLD Development System User Manual



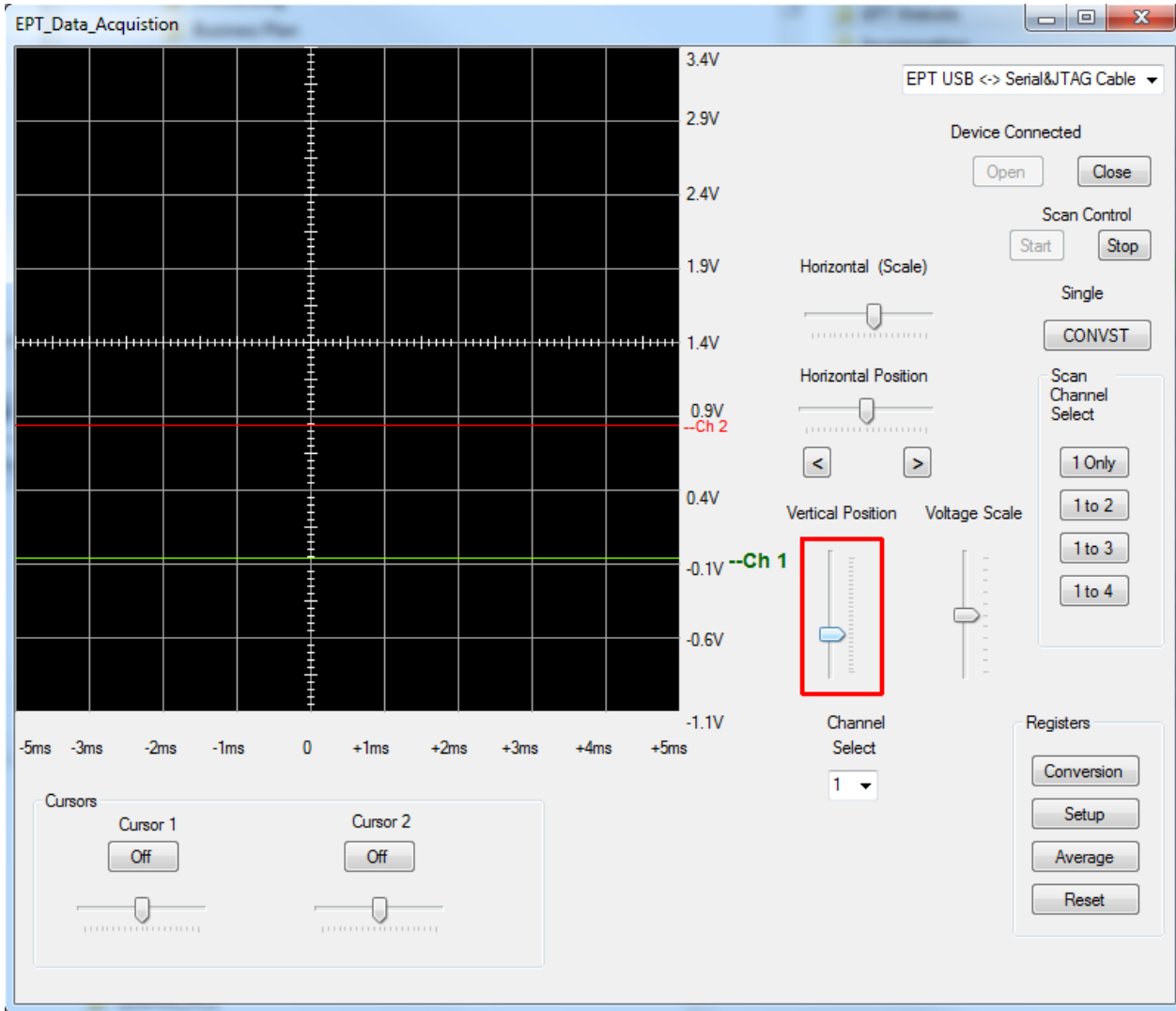
Next, locate the Channel Select drop down box and click on channel 1.

UnoMax CPLD Development System User Manual



Locate the Vertical Position slider and pull it down. The channel 1 data will change position in the graph depending on where you move the slider. The voltage magnitude data also adjusts to indicate the magnitude of the data relative to the position of channel 1 data.

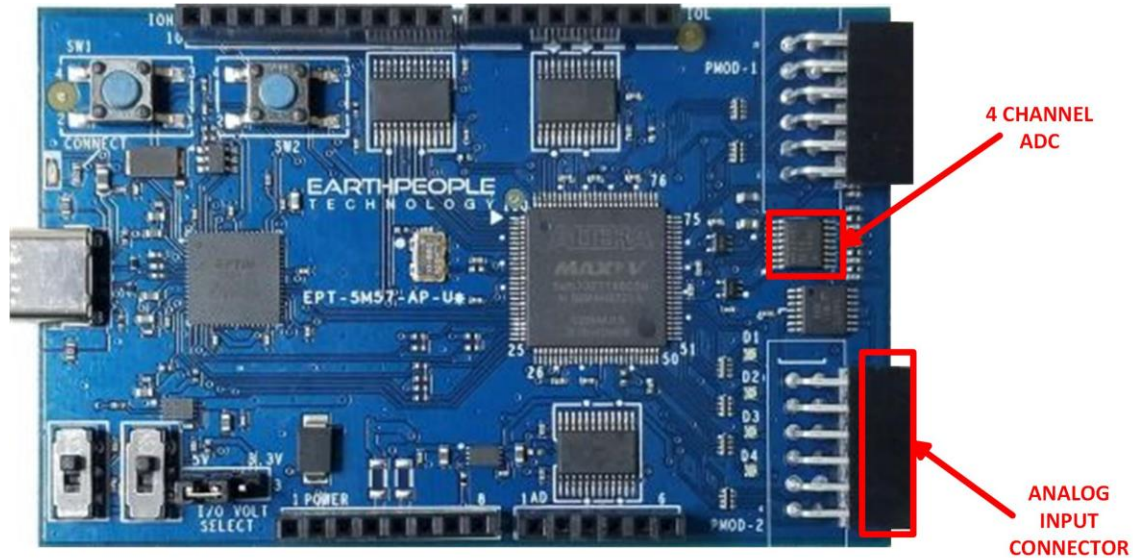
UnoMax CPLD Development System User Manual



The selected channel will show up as a large icon. Its position indicates the zero position of the data. The magnitude information along the y-axis is only for the selected channel.

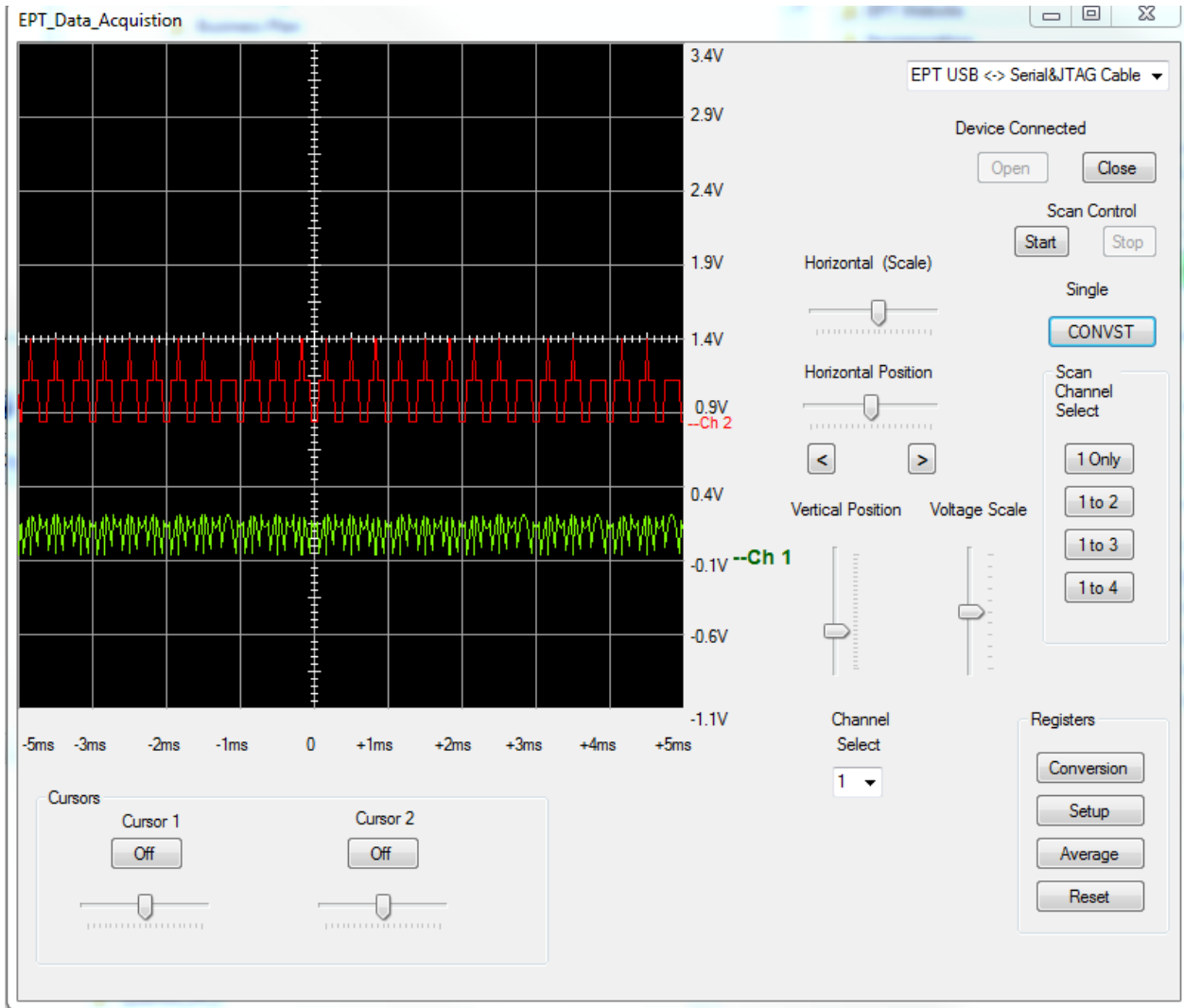
UnoMax CPLD Development System User Manual

Then connect a signal to the channel 1 input on the UnoMax2.



If you don't have a 0-5 Volt signal to connect to the UnoMax2, you can use your finger and touch it to the bottom of the Analog Input Connector. The ambient electricity from your body has just enough current to give the Analog inputs a deflection from zero.

UnoMax CPLD Development System User Manual



Now the UnoMax and UnoProLyzzer are ready to measure an 0-5VDC signals.

8 APPENDIX I

List of Abbreviations and Acronyms

EPT	Earth People Technology
FIFO	First In – First Out



UnoMax CPLD Development System User Manual

FTDI	Future Technology Device International
HSP	Hyper Serial Port
I2C	Inter-Integrated Circuit
JTAG	Joint Test Action Group
PC	Personal Computer
CPLD	Complex Programmable Logic Device
USB	Universal Serial Bus

APPENDIX II

Details of the Altera EPM570 CPLD