USB Blaster III FPGA Development Cable User Guide

Online Version
Send Feedback

864466
2026.03.11

Contents

1. Introduction **4**
1.1. USB Blaster III FPGA Development Cable Kit Contents 4
1.2. Supported Devices and Systems 5

2. Setting Up the USB Blaster III FPGA Development Cable **8**
2.1. Power Source Requirements 8
2.2. Software Requirements and Supported Operating Systems 8
2.3. Connecting the USB Blaster III FPGA Development Cable 10
2.4. Installing the USB Blaster III FPGA Development Cable on Windows Systems 10
2.5. Installing the USB Blaster III FPGA Development Cable on Linux Systems 12

3. Programming with Your USB Blaster III FPGA Development Cable **13**
3.1. Programming Your FPGA with Quartus Prime Programmer and the USB Blaster III FPGA Development Cable 13
3.1.1. Changing TCK Frequency with Quartus Prime Programmer 15
3.1.2. TCK Frequency Auto-Adjust with Quartus Prime Programmer 16
3.2. Command Line Programming Your FPGA with the USB Blaster III FPGA Development Cable 18
3.2.1. Changing TCK Frequency with Command Line Programming 18
3.2.2. TCK Frequency Auto-Adjust with Command Line Programming 20

4. Functional Overview **21**
4.1. Block Diagram 21
4.2. Signal Names and Programming Modes 22
4.3. LED Description 23

5. USB Blaster III FPGA Development Cable Specifications **24**
5.1. Dimensions of the USB Blaster III FPGA Development Cable 24
5.2. Voltage Requirements by Device 30
5.3. Operating Conditions 31
5.4. JTAG Timing Constraints and Waveforms 33
5.4.1. Creating JTAG Timing Constraints in Quartus Prime Software 35

6. Document Revision History for the USB Blaster III FPGA Development Cable User Guide **37**

A. Additional Information..... 38
A.1. Manually Installing the USB Blaster III FPGA Development Cable Drivers on Windows..... 38
A.2. Binding the FTDI_SIO Driver on Older Linux Kernels..... 39
A.3. Certification Statements..... 41
A.3.1. RoHS Compliance..... 41
A.3.2. CE EMI Conformity Caution..... 41

1. Introduction

The USB Blaster III FPGA Development Cable provides a quick and convenient method for programming and configuring Altera FPGAs, CPLDs, and configuration devices. This cable interfaces a USB port on a host computer and sends data to a 1.27 mm micro or 2.54 mm legacy 10-pin header connected to the FPGA, CPLD, or configuration device.

You can use the USB Blaster III FPGA Development Cable for the following:

- Iteratively download configuration data to a system during prototyping
- Program data into the system during production
- Advanced Encryption Standard (AES) key and fuse programming
- JTAG debugging of system level designs

Caution: To prevent damage to the USB Blaster III FPGA Development Cable and your target devices, always observe proper ESD handling procedures. Use a grounded wrist strap and work on an ESD-safe mat. Avoid touching exposed conductive parts of the cables and connectors when the system is powered.

Related Information

Cables and Adapter Drivers Information

1.1. USB Blaster III FPGA Development Cable Kit Contents

Verify that the following items are included in your USB Blaster III FPGA Development Cable kit.

Table 1. USB Blaster III FPGA Development Cable Kit Contents

Item	Description
Programmer Module	The central unit responsible for USB communication, signal translation, and voltage level adaptation. It features status LEDs for power and activity, a USB Type-C port for PC connection, and a micro cable programming socket for cable attachment.
USB 2.0 Cable, USB Type-A to USB Type-C, 1 meter	A shielded cable connecting the programmer module to the host computer, supporting USB 2.0 speeds for fast, reliable data transfer.
2" Performance Programming Cable	An ultra-short, high-performance cable engineered for minimal signal loss and optimal integrity. This cable is specifically designed to support reliable JTAG operation at clock rates up to 30 MHz, where the standard cable may not meet timing constraints due to increased propagation delay and signal degradation. This cable is ideal for high-speed programming, use in test fixtures, and environments where cable length must be minimized.
6" Programming Cable	The standard cable for connecting the programmer module to the target board via a micro connector. This cable is suitable for most bench and prototype setups, supporting JTAG clock rates up to 15 MHz.
Micro-to-Legacy Adapter Board	A small PCB that converts the 1.27 mm 10-pin micro programming connector to the 2.54 mm legacy 10-pin programming header, ensuring compatibility with legacy and current development boards.

Related Information

Dimensions of the USB Blaster III FPGA Development Cable on page 24

1.2. Supported Devices and Systems

You can use the USB Blaster III FPGA Development Cable with the following devices:

Table 2. Supported Devices for Configuration Data

Device Families	
Agilex™ FPGA portfolio	Agilex 3
	Agilex 5
	Agilex 7
	Agilex 9

continued...

Device Families	
Stratix® FPGA portfolio	Stratix III
	Stratix IV
	Stratix V
	Stratix 10
Arria® FPGA portfolio	Arria V
	Arria 10
Cyclone® FPGA portfolio	Cyclone II
	Cyclone III
	Cyclone IV
	Cyclone V
	Cyclone 10
	Cyclone 10 LP
	Cyclone 10 GX
MAX® series CPLDs	MAX II
	MAX V
	MAX 10

You can use the USB Blaster III FPGA Development Cable to perform in-system programming of the following serial flash devices:

Table 3. Supported Serial Flash Devices for In-System Programming

Device Families	
EPCQA	EPCQ16A
	EPCQ32A
	EPCQ64A
	EPCQ128A
	continued...

Device Families	
Micron MT25QL	MT25QL128
	MT25QL256
	MT25QL512
	MT25QL01G
	MT25QL02G
Micron MT25QU	MT25QU128
	MT25QU256
	MT25QU512
	MT25QU01G
	MT25QU02G
Macronix MX25L	MX25L128
	MX25L256
	MX25L512
Macronix MX25U	MX25U128
	MX25U256
	MX25U512
Macronix MX66U	MX66U512
	MX66U1G
	MX66U2G
Infineon S25FL	S25FL128
	S25FL256
	S25FL512

2. Setting Up the USB Blaster III FPGA Development Cable

The following sections provide you with the necessary information and steps to correctly set up your USB Blaster III FPGA Development Cable.

2.1. Power Source Requirements

The USB Blaster III FPGA Development Cable's programmer module is powered directly via the host PC's USB port, requiring no external power supply. The target device's I/O voltage reference (VREF) is supplied by the target board through the programming connector.

- USB supply voltage of 5.0 V from the USB Blaster III FPGA Development Cable
- Between 1.5 V and 3.3 V from the target circuit board

For more details, refer to the *Voltage Requirements by Device* section.

Related Information

Voltage Requirements by Device on page 30

2.2. Software Requirements and Supported Operating Systems

Ensure you have the necessary software and suitable operating systems installed before proceeding with setting up the USB Blaster III FPGA Development Cable.

Supported operating systems:

- Windows 10 (64-bit)
- Windows 11 (64-bit)
- Windows Server 2019 (64-bit)
- Windows Server 2022 (64-bit)
- (Linux) Red Hat Enterprise 8.6 and above

Required software:

- Quartus® Prime version 25.1 or later

  Note: Install the latest version and patches (if any) to ensure full compatibility with the USB Blaster III FPGA Development Cable.

The USB Blaster III FPGA Development Cable also supports the following tools:

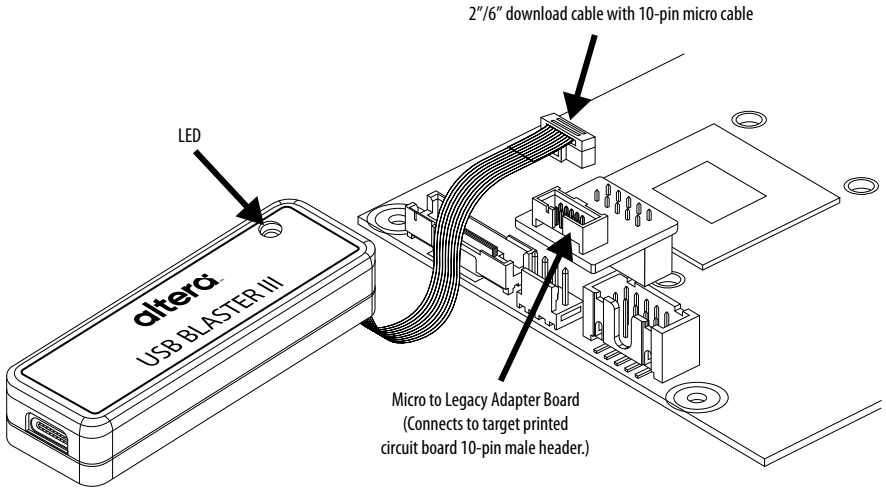
- Quartus Prime Programmer (and standalone version)
- Quartus Prime Signal Tap Logic Analyzer (and standalone version)
- JTAG and debug tools supported by the JTAG server, for example: System Console, Nios® V debugger.

Related Information

Quartus Prime Pro Edition User Guide: Programmer

2.3. Connecting the USB Blaster III FPGA Development Cable

Figure 1. USB Blaster III FPGA Development Cable



Follow these steps to connect your cable.

1. Disconnect the power from the circuit board.
2. Connect the USB 2.0 cable to the USB Type-A port on your computer and to the USB Type-C port on the programmer module.
3. Connect either the 2" or 6" programming cable to the micro programming connector on the programmer module and to the:
   - 1.27 mm micro socket on the device board.
   - (for legacy sockets) 1.27 mm micro socket on the micro-to-legacy adapter board.
4. Reconnect the power to reapply power to the circuit board.

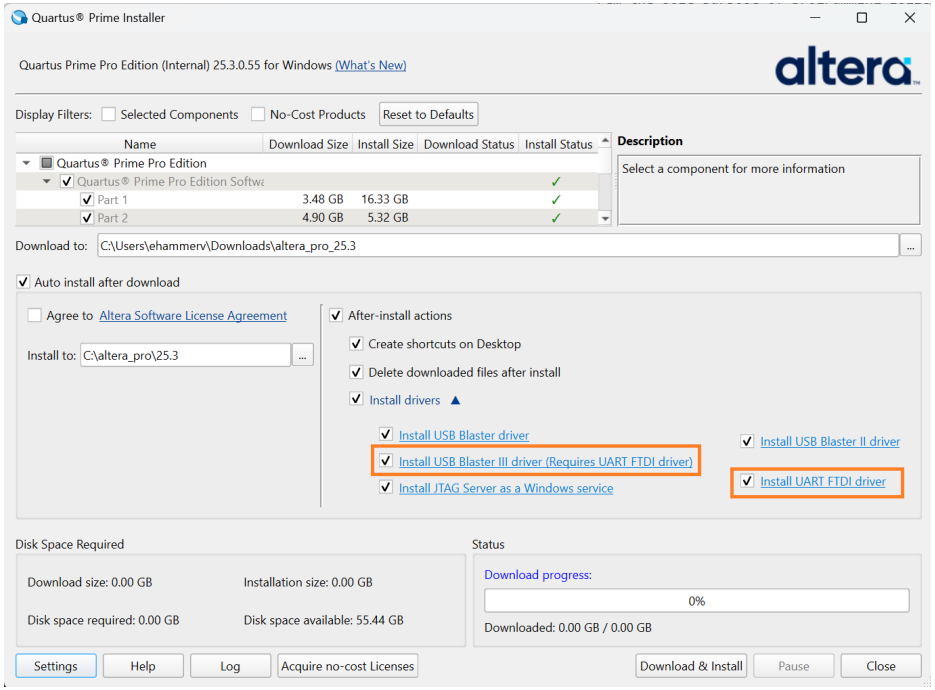
2.4. Installing the USB Blaster III FPGA Development Cable on Windows Systems

The USB Blaster III FPGA Development Cable drivers are included in the Quartus Prime software installation.

During the Quartus Prime software installation, in the after-install actions section of the installation menu, ensure the following items are selected:

- **Install USB Blaster III driver (Requires UART FTDI driver)**
- **Install UART FTDI driver**

Figure 2. After-Install Actions on Software Installation Menu



After the installation is complete, verify that the driver is located in your directory:

```
\<Quartus Prime system directory>\drivers\USB Blaster-iii
```

If the driver is not in your directory, repeat the Quartus Prime software installation to bring up the installation menu and attempt the USB Blaster III FPGA Development Cable driver installation again.

2.5. Installing the USB Blaster III FPGA Development Cable on Linux Systems

To access the cable, the Quartus Prime software uses the built-in Red Hat USB drivers, the USB file system (`usbfs`). By default, **root** is the only user allowed to use usbfs. You must have system administration (root) privileges to configure the USB Blaster III FPGA Development Cable drivers.

1. Create a file named `/etc/udev/rules.d/51-usbblaster.rules` and add the following lines to it.

```
# USB Blaster III
SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device", ATTR{idVendor}=="09fb", ATTR{idProduct}=="6023", MODE="0666",
NAME="bus/usb/$env{BUSNUM}/$env{DEVNUM}", RUN+="/bin/chmod 0666 %c"
```

   Important: Do not add extra line breaks to the `.rules` file.

   Note: The `.rules` file may already exist if you have installed an earlier USB Blaster version.

2. After the `usbblaster.rules` file has been updated a reboot or reload of the udev rules is required. To reload, root access is required to run the command:

```
udevadm control --reload-rules && udevadm trigger
```

3. Complete your installation by setting up the programming hardware in the Quartus Prime software. Refer to the *Programming with Your USB Blaster III FPGA Development Cable* section.

For more information about USB Blaster III FPGA Development Cable driver installation, refer to the *Cable and Adapter Drivers Information* webpage.

Related Information

- Cables and Adapter Drivers Information
- Programming with Your USB Blaster III FPGA Development Cable on page 13

3. Programming with Your USB Blaster III FPGA Development Cable

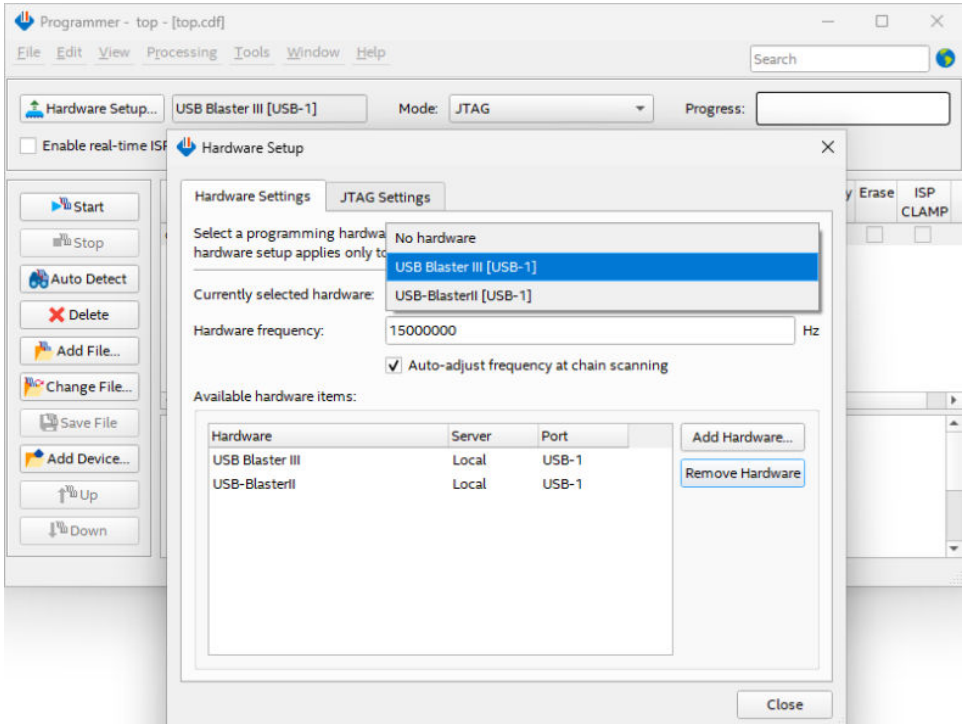
This section describes the procedures for programming your FPGA using the USB Blaster III FPGA Development Cable, covering both graphical and command-line workflows in the Quartus Prime software, as well as instructions for adjusting the JTAG TCK frequency for optimal performance.

3.1. Programming Your FPGA with Quartus Prime Programmer and the USB Blaster III FPGA Development Cable

Configure your hardware setup to begin programming your FPGA with Quartus Prime Programmer and your USB Blaster III FPGA Development Cable:

1. In the Quartus Prime software, navigate to **Tools ➤ Programmer**. The **Programmer** window appears.
2. Click **Hardware Setup**.
3. Click the **Hardware Settings** tab.
4. In the **Currently selected hardware** list, select **USB Blaster III**.

Figure 3. Select the Cable



5. Click **Close** and return to the **Programmer** window.
6. In the **Mode** list, choose an appropriate programming mode, referencing the table below.

Figure 4. Select Programming Mode

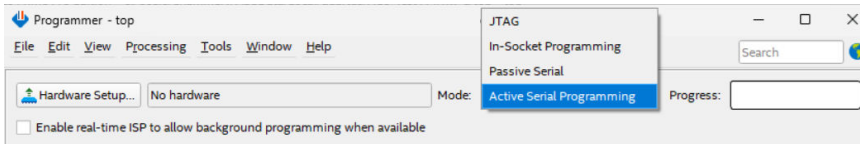


Table 4. Supported Programming Modes

Programming Mode	Description
JTAG	Programs or configures all Altera devices supported by Quartus Prime software via JTAG programming.
In-Socket Programming	Not supported with the USB Blaster III FPGA Development Cable
Passive Serial	Not supported with the USB Blaster III FPGA Development Cable
Active Serial Programming	Programs a single flash device.

Related Information

- Quartus Prime Pro Edition User Guide: Programmer
- Command Line Programming Your FPGA with the USB Blaster III FPGA Development Cable on page 18

3.1.1. Changing TCK Frequency with Quartus Prime Programmer

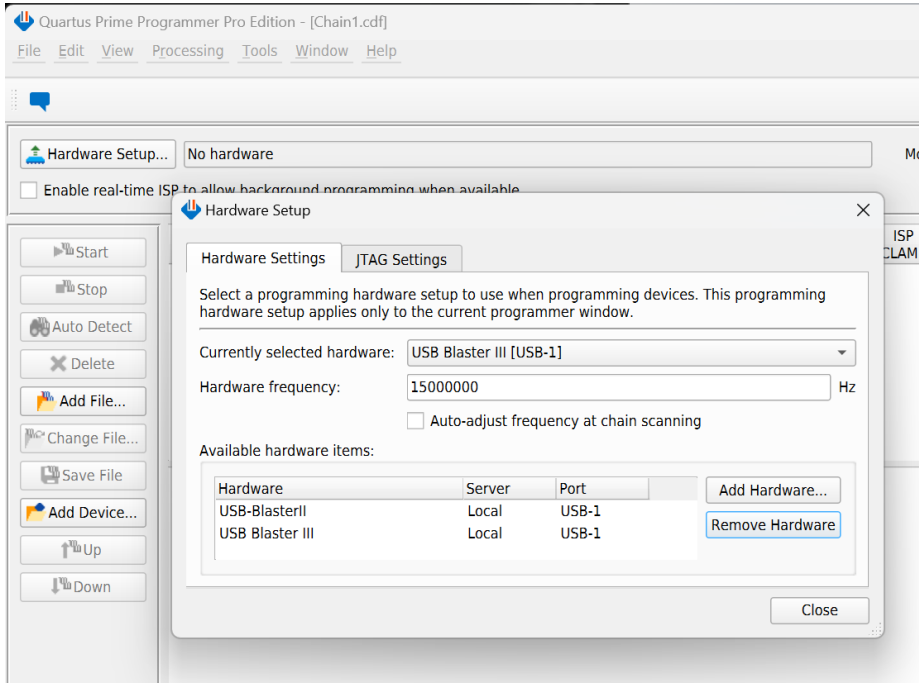
The USB Blaster III FPGA Development Cable has a default TCK frequency of 15 MHz but can be run at 30 MHz if signal integrity and board timing requirements are met.

See *JTAG Timing Constraints and Waveforms* for more information.

To change the TCK frequency of the USB Blaster III FPGA Development Cable with Quartus Prime Programmer:

1. In Quartus Prime Programmer, click on **Hardware Setup**.
2. Under the **Hardware Settings** tab, uncheck **Auto-adjust frequency at chain scanning** to disable the feature.
3. Type in one of the recommended frequencies in the **Hardware frequency** field.
4. Close the **Hardware Setup** dialog box to save the settings.

Figure 5. Quartus Prime Programmer - Hardware Setup



Related Information

- JTAG Timing Constraints and Waveforms on page 33
- Changing TCK Frequency with Command Line Programming on page 18

3.1.2. TCK Frequency Auto-Adjust with Quartus Prime Programmer

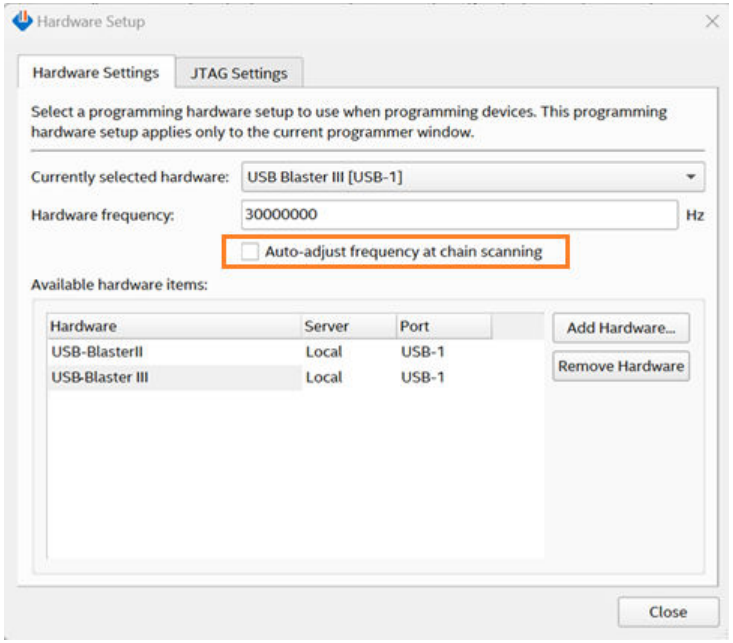
The TCK Frequency Auto-Adjust feature ensures correct TCK frequency settings to prevent device failures during JTAG operations.

It is turned on by default and can be toggled on and off via either the command line interface. The feature resets to default when the cable is reconnected or the JTAG server is restarted.

The TCK Frequency Auto-Adjust feature always applies the optimal frequency supported by the current JTAG chain based on the self-tests. If you specify a TCK frequency, the auto-adjust feature stops at that frequency, provided the self- tests pass at that frequency. Otherwise, the auto-adjust feature continues and stops at a lower frequency where the self-tests have passed.

Note: The TCK frequency auto-adjust is designed based on the hard JTAG scan chain. For virtual JTAG scan chain, the TCK frequency after auto-adjust may still require further adjustment from the user for successful JTAG operation.

Figure 6. Auto-Adjust Frequency in Programmer GUI



A checkbox is added in the Quartus Prime Programmer's **Hardware Setup** dialog box to toggle the auto-adjust frequency feature on and off. When the auto-adjust frequency is enabled, the new adjusted TCK frequency value is shown under the message box of the Programmer GUI.

This checkbox is grayed out when the auto-adjust frequency feature is unavailable.

Related Information

TCK Frequency Auto-Adjust with Command Line Programming on page 20

3.2. Command Line Programming Your FPGA with the USB Blaster III FPGA Development Cable

Follow these steps to begin command line programming.

1. Use the Quartus Prime Programmer to list the available programming cables, using the `quartus_pgm -l` command.

   Example of what you should see:

   ```
   Info: Command: quartus_pgm -l
   1) USB BlasterIII [USB-1]
   ```

2. Use the `quartus_pgm -c 1 –auto` command to auto-detect devices on the JTAG chain.

   Example of what you should see:

   ```
   Info: Command: quartus_pgm -c 1 --auto
   Info (213045): Using programming cable "AG3C_DK [USB-1]"
   1) USB Blaster III [USB-1]
     4369B0DD   A3C(Y135BM16A|Z135BM16A)
   ```

3. Use the following command line to program your FPGA with the connected USB Blaster III FPGA Development Cable:
   `quartus_pgm –m jtag –o "p;<path_to_file.sof>@<device_number>”`

   Example: assuming the cable index is 1, device number is 1, and file is output_file.sof:

   ```
   quartus_pgm -c 1 -m jtag -o "p;output_file.sof@1”
   ```

Related Information

Programming Your FPGA with Quartus Prime Programmer and the USB Blaster III FPGA Development Cable on page 13

3.2.1. Changing TCK Frequency with Command Line Programming

The USB Blaster III FPGA Development Cable has a default TCK frequency of 15 MHz but can be run at 30 MHz if signal integrity and board timing requirements are met.

Refer to the *JTAG Timing Constraints and Waveforms* section for more information.

To change the TCK frequency of the USB Blaster III FPGA Development Cable with command line programming:

1. Open the command line interface with the USB Blaster III FPGA Development Cable bin directory in your path.
   Example:
   ```
   C:\<Quartus Prime system directory>\25.3\quartus\bin64
   ```
2. Enter the following command to change the TCK frequency:
   ```
   jtagconfig --setparam <cable_number> JtagClock <frequency><unit_prefix>
   ```
   where:
   - `cable_number` is the USB Blaster III FPGA Development Cable to be modified.
   - `frequency` is the desired TCK frequency, from one the following supported rates:
     - `30M`
     - `15M`
     - `10M`
     - `7500K`
     - `6M`
     - `5M`
     - `3M`
     - `2M`
     - `1M`
     - <30/n>M (between 10 kHz and 6 MHz, where n represents an integer value number)
   - `unit_prefix` is the unit prefix for the frequency (for example, M for MHz).

   An example for setting TCK frequency to 6 MHz:
   ```
   jtagconfig --setparam 1 JtagClock 6M
   ```

Related Information

- JTAG Timing Constraints and Waveforms on page 33

- Changing TCK Frequency with Quartus Prime Programmer on page 15

3.2.2. TCK Frequency Auto-Adjust with Command Line Programming

The TCK Frequency Auto-Adjust feature ensures correct TCK frequency settings to prevent device failures during JTAG operations.

It is turned on by default and can be toggled on and off via either the command line interface. The feature resets to default when the cable is reconnected or the JTAG server is restarted.

The TCK Frequency Auto-Adjust feature always applies the optimal frequency supported by the current JTAG chain based on the BYPASS tests. If you specify a TCK frequency, the auto-adjust feature stops at that frequency, provided the BYPASS tests pass at that frequency. Otherwise, the auto-adjust feature continues and stops at a lower frequency where the BYPASS tests has passed.

Note: The TCK frequency auto-adjust is designed based on the hard JTAG scan chain. For virtual JTAG scan chain, the TCK frequency after auto-adjust may still require further adjustment from the user for successful JTAG operation.

To turn off the auto-adjust frequency feature, use the following command:

```
jtagconfig --setparam <cable number> JtagClockAutoAdjust 0
```

To turn on the auto-adjust frequency feature, use the following command:

```
jtagconfig --setparam <cable number> JtagClockAutoAdjust 1
```

You may use the following command to check whether the auto-adjust frequency is enabled. The expected output is `0` if auto-adjust is disabled and `1` if the auto-adjust is enabled.

```
Jtagconfig --getparam <cable number> JtagClockAutoAdjust
```

Related Information

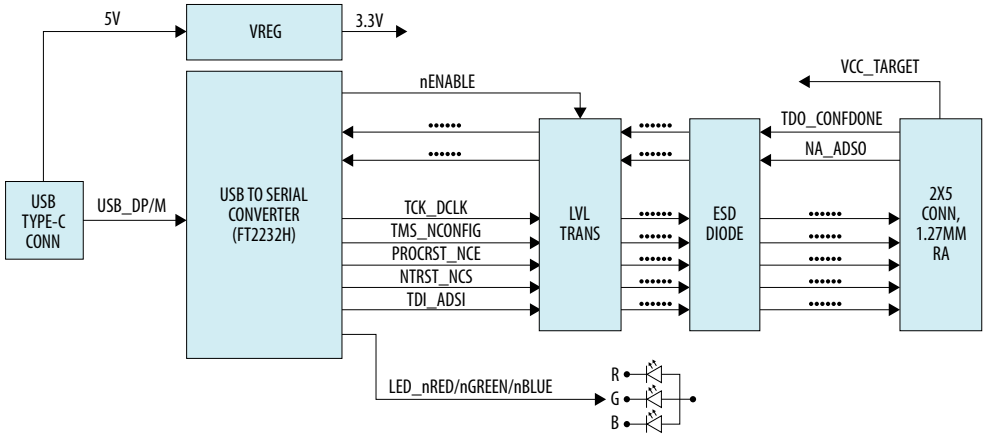
TCK Frequency Auto-Adjust with Quartus Prime Programmer on page 16

4. Functional Overview

This section provides a functional overview of the USB Blaster III FPGA Development Cable's programmer module, including an internal block diagram, descriptions of key signal names, supported programming modes, and detailed information on the status LED indicators. A summary table outlines LED color states and their corresponding operational meanings.

4.1. Block Diagram

Figure 7. USB Blaster III FPGA Development Cable Block Diagram



The USB Blaster III FPGA Development Cable's programmer module features a compact design that connects to a host PC via a USB Type-C port, supplying both power and JTAG/Active Serial data.

At its core, the FTDI FT2232 USB interface controller efficiently translates USB signals into JTAG and Active Serial (AS) programming signals. These are passed through a level translator circuit, which automatically adjusts voltage levels for broad device compatibility and incorporates ESD protection for signal reliability.

Status LEDs offer clear visual feedback for power and activity. The programming signals are delivered through a 2 × 5 micro connector, supporting both standard and performance cables, as well as an adapter for legacy headers, ensuring robust and flexible device programming across development and production environments.

4.2. Signal Names and Programming Modes

This section summarizes the signal names and programming modes for both the 1.27 mm and 2.54 mm headers used with the USB Blaster III FPGA Development Cable. It includes visual references for pin numbering and a table mapping pin numbers to signal names, supporting straightforward identification and reliable connections across different header formats.

Figure 8. Adapter Board 10-Pin Male 1.27 mm Micro Socket

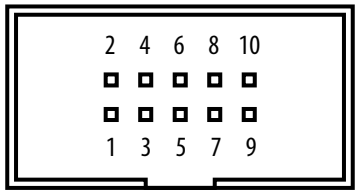


Figure 9. Adapter Board 10-Pin Female 2.54 mm Legacy Header

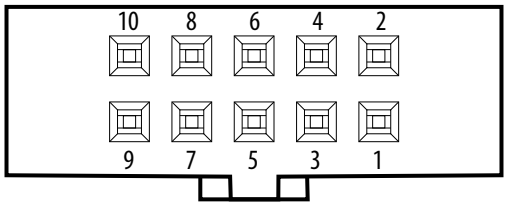


Table 5. Pin Number vs Signal Name

1.27 mm Micro Header	2.54 mm Legacy Header	Driven by	JTAG Function	Signal Description	AS Function	Signal Description
1	4	DUT	VCC_TARGET	Target power supply	VCC_TARGET	Target power supply
2	5	Cable	TMS	Test mode select input	nCONFIG	Configuration control
3	2	Cable	GND	Signal ground	GND	Signal ground
4	1	Cable	TCK	Clock signal	DCLK	Clock signal
5	7	DUT	–	–	ASDO	Active serial data out
6	3	DUT	TDO	Test data output	CONFDONE	Configuration done
7	8	Cable	nTRST	Test reset	nCS	Serial configuration device chip select
8	9	Cable	TDI	Test data input	ASDI	Active serial data in
9	10	GND	GND	Signal ground	GND	Signal ground
10	6	Cable	nPROCRST	Processor reset	nCE	Target chip enable

4.3. LED Description

The LED on the USB Blaster III FPGA Development Cable's programmer module indicates different conditions.

Table 6. LED Indicator

LED Color	State	Description
None	Off	Not connected or is suspended
Blue	Solid	Connected, not in use
Green	Solid	JTAG port open, idle
Green	Blinking	JTAG port open, active
Purple	Blinking	Occurs when JTAG command Identify(1) is triggered

(1) `jtagconfig --setparam <cable number> Identify <1=on, 0=off>`

5. USB Blaster III FPGA Development Cable Specifications

This section provides the specifications for the USB Blaster III FPGA Development Cable.

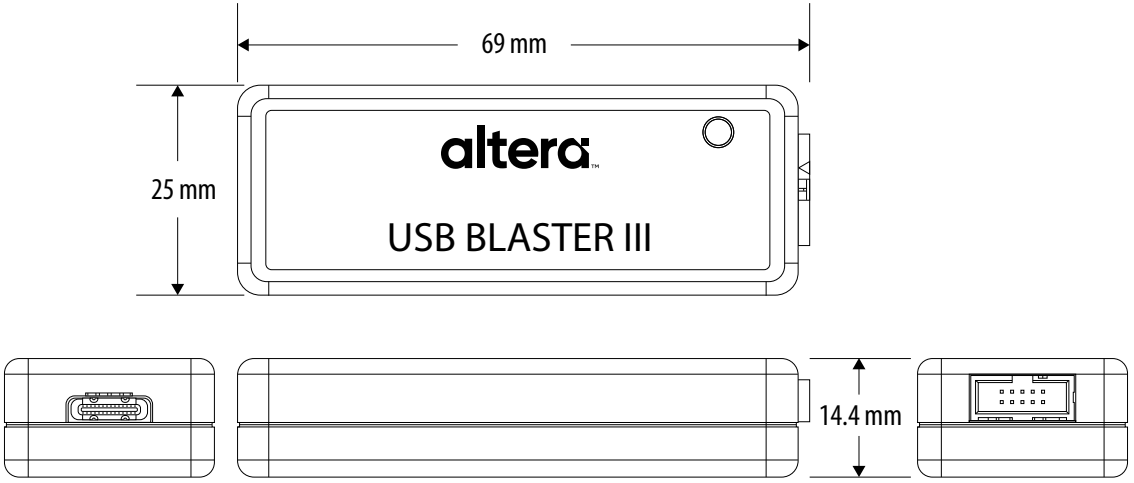
5.1. Dimensions of the USB Blaster III FPGA Development Cable

This sections provides the physical dimensions for various components of the USB Blaster III FPGA Development Cable.

All connectors are keyed to prevent incorrect insertion.

Programmer Module

Figure 10. Programmer Module Dimensions



The programmer module features a USB Type-C port and a micro programming connector.

Programming Cables

Figure 11. Programming Cables Dimensions

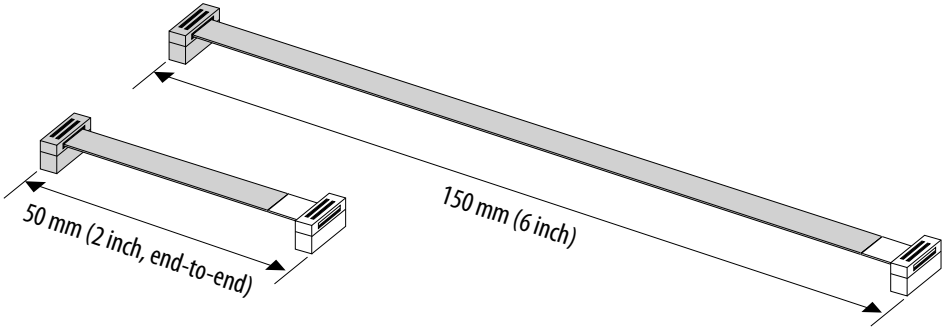
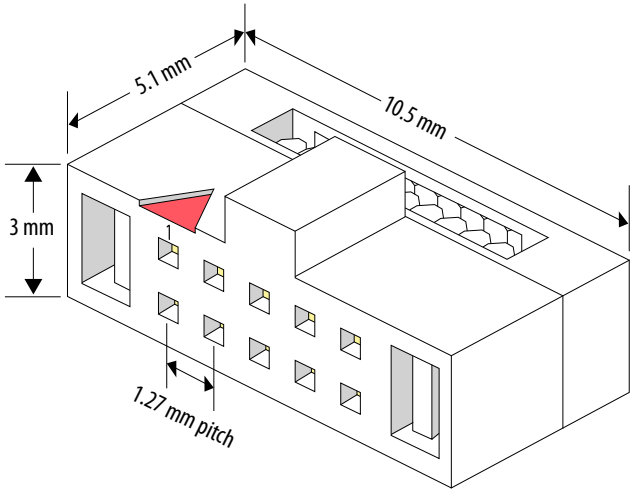


Figure 12. Micro Connectors on Programming Cables

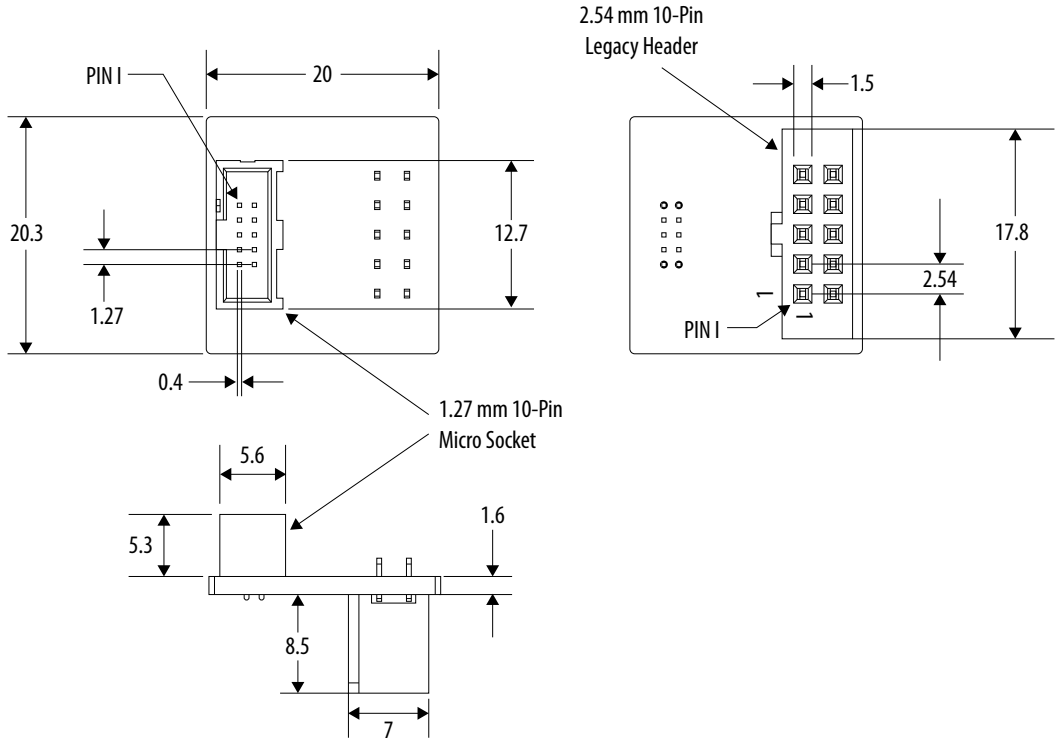


The programming cables utilizes a flat ribbon design to maintain a compact profile and facilitate neat, organized routing in benchtop or prototype environments. Both ends are equipped with 2×5 micro connectors ensuring secure and reliable connections to the programmer module and adapter board.

The ribbon cable's low-profile construction allows it to fit easily in tight spaces, while its flexibility and lightweight nature support convenient handling and consistent signal transmission.

Micro-to-Legacy Adapter Board

Figure 13. Micro-to Legacy Adapter Board

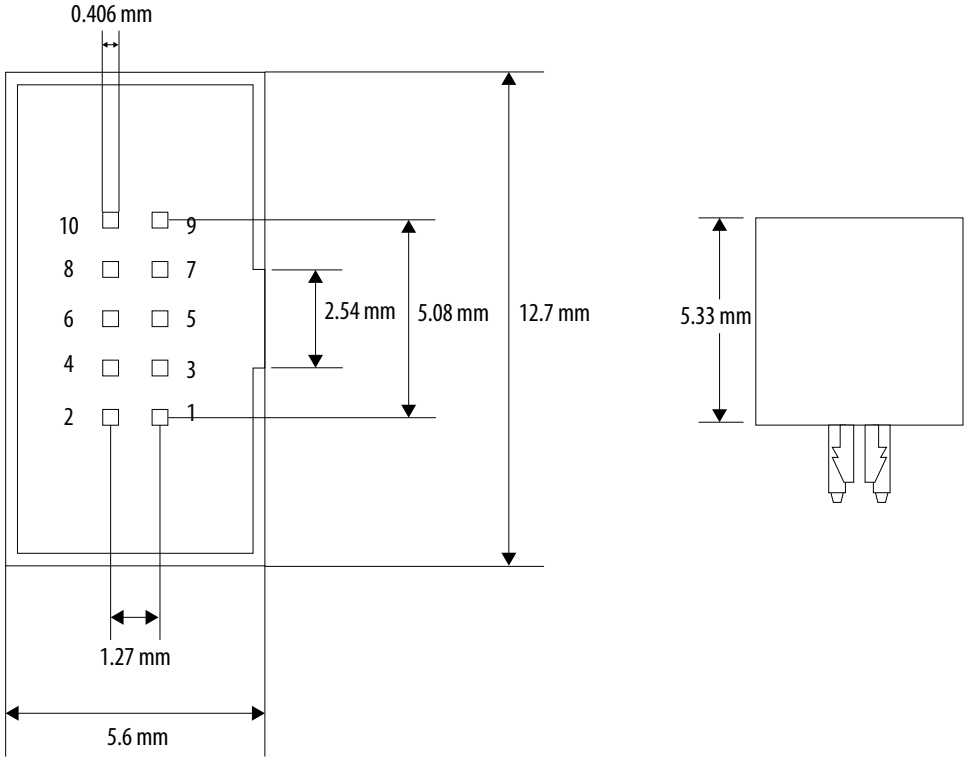


The adapter board allows the programming cable to interface with 2.54 mm legacy 10-pin connectors commonly found on development boards of earlier Altera FPGA devices. This board converts a 10-pin female Micro Socket connection with 1.27 mm pitch to a 10-pin male Legacy header with 2.54 mm pitch.

Note: The signal connections between the micro header and the legacy header are not identical. Refer to the *Signal Names and Programming Modes* section for correct signal conversion.

Circuit Board 1.27 mm Micro Socket Connection (Recommended)

Figure 14. 1.27 mm 10-Pin Male Micro Socket



The 10-pin micro socket on the programmer module features a compact 2×5 pin configuration and is designed for reliable, space-saving connections. Its small footprint allows for direct, streamlined cable attachment, reducing clutter and improving durability compared to traditional larger headers.

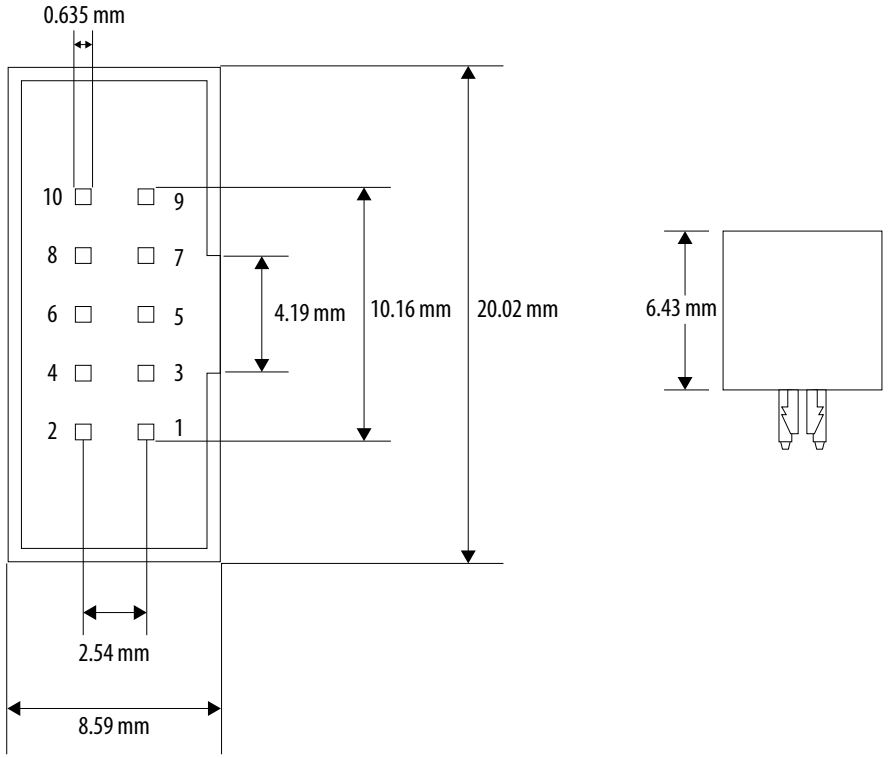
For new designs, customers are encouraged to incorporate this micro socket directly onto their target boards, eliminating the need for an adapter board and enabling a more robust, modern, and efficient programming interface over legacy solutions.

Note: If the header connection on the circuit board is a male receptacle, it must have a key notch. The 10-pin female plug cannot connect to it without a key notch.

Although a 10-pin surface mount header can be used for the download cable, Altera recommends using a through-hole connector. Through-hole connectors hold up better under repeated insertion and removal.

Circuit Board 2.54 mm Legacy Socket Connection

Figure 15. 2.54 mm 10-Pin Male Legacy Socket



The 10-pin male header, which connects to the adapter board's 10-pin female plug, has two rows of five pins. The pins are connected to the device's programming or configuration pins.

Note: If the header connection on the circuit board is a male receptacle, it must have a key notch. The 10-pin female plug cannot connect to it without a key notch.

Although a 10-pin surface mount header can be used for the download cable, Altera recommends using a through-hole connector. Through-hole connectors hold up better under repeated insertion and removal.

Related Information

Signal Names and Programming Modes on page 22

5.2. Voltage Requirements by Device

The USB Blaster III FPGA Development Cable's programmer module automatically adapts its signal voltage levels to match your target device's I/O voltage reference (VREF), ensuring safe, reliable programming across a wide range of devices without manual configuration. This simplifies board design, reduces the risk of electrical incompatibility, and streamlines integration into both development and production workflows.

The $V_{CC(TRGT)}$ pin on the programmer module must be connected to a specific voltage for the device being programmed. Connect pull-up resistors to the same power supply as the programmer module: $V_{CC(TRGT)}$.

Table 7. $V_{CC(TRGT)}$ Pin Voltage Requirements

Device	VCC Voltage Required
Agilex 3	As specified by VCCIO_SDM
Agilex 5	As specified by VCCIO_SDM
Agilex 7	As specified by VCCIO_SDM
Agilex 9	As specified by VCCIO_SDM
Stratix IV	As specified by V_{CCPGM} or V_{CCPD}
Stratix V	As specified by V_{CCPD} Bank 3A
Stratix 10	As specified by VCCIO_SDM
Arria V	As specified by V_{CCPD} Bank 3A
Arria 10	As specified by V_{CCPGM} or V_{CCIO}
Cyclone II	As specified by V_{CCIO}
Cyclone III	As specified by V_{CCA} or V_{CCIO}
Cyclone IV	As specified by V_{CCIO}.

continued...

Device	VCC Voltage Required
	• Bank 9 for Cyclone IV GX devices • Bank 1 for Cyclone IV E devices
Cyclone V	As specified by V_{CCPD} Bank 3A
Cyclone 10 LP	As specified by V_{CCA} or V_{CCIO}
Cyclone 10 GX	As specified by V_{CCPGM}
MAX II	As specified by V_{CCIO} Bank 1
MAX V	As specified by V_{CCIO} Bank 1
MAX 10	As specified by V_{CCIO}
EPC4, EPC8, EPC16	3.3 V
EPCS1, EPCS4, EPCS16, EPCS64, EPCS128	3.3 V
EPCQ256, EPCQ512	3.3 V
EPCQ-L	1.8 V

5.3. Operating Conditions

The following tables summarize the maximum ratings, recommended operating conditions, and DC operating conditions for the USB Blaster III FPGA Development Cable.

Table 8. USB Blaster III FPGA Development Cable Absolute Maximum Ratings

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
$V_{CC(TRGT)}$	Target supply voltage	With respect to ground	0	5.5	V
$V_{CC(USB)}$	USB supply voltage	With respect to ground	–	6.0	V
I_I	Target side input current	Inputs	-50.0	50.0	mA
$I_{I(USB)}$	USB supply current	VBUS	–	500.0	mA
I_O	Target side output current	Outputs	-150.0	150.0	mA
$V_{CC(TRGT)}$	Target supply voltage, 5.0 V operation	–	4.75	5.25	V

continued...

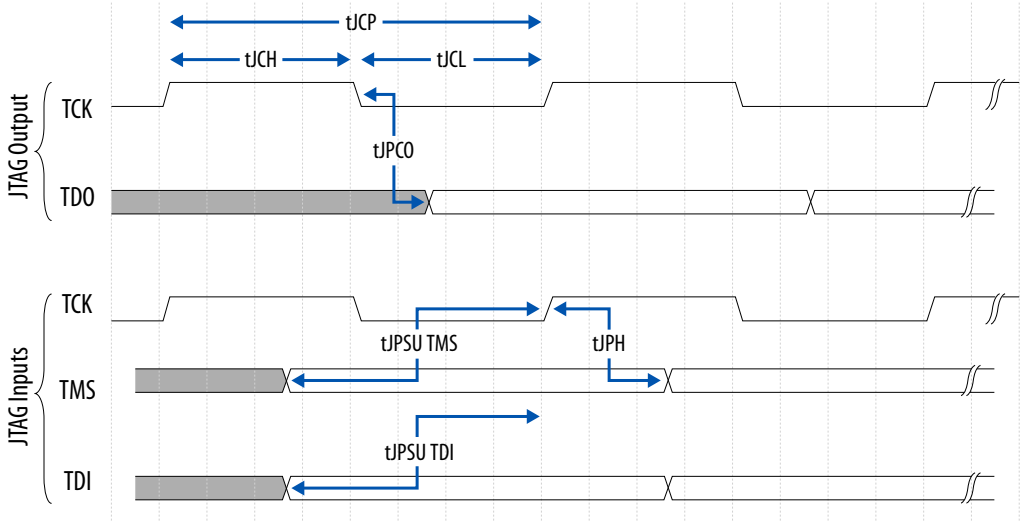
Symbol	Parameter	Conditions	Minimum	Maximum	Unit
	Target supply voltage, 3.3 V operation	–	3.0	3.6	V
	Target supply voltage, 2.5 V operation	–	2.375	2.625	V
	Target supply voltage, 1.8 V operation	–	1.71	1.89	V
	Target supply voltage, 1.5 V operation	–	1.43	1.57	V

Table 9. USB Blaster III FPGA Development Cable DC Operating Conditions

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{IH}	High-level input voltage	4.5 V ≥$V_{CC(TRGT)}$ ≥1.5 V	0.7 × $V_{CC(TRGT)}$	–	–	V
	High-level input voltage	1.5 V ≥$V_{CC(TRGT)}$ ≥1.0 V	0.8 × $V_{CC(TRGT)}$	–	–	V
V_{IL}	Low-level input voltage	4.5 V ≥$V_{CC(TRGT)}$ ≥1.5 V	–	–	0.3 × $V_{CC(TRGT)}$	V
	Low-level input voltage	1.5 V ≥$V_{CC(TRGT)}$ ≥1.0 V	–	–	0.3 × $V_{CC(TRGT)}$	V
V_{OH}	High-level output voltage	The V_{OH} of USB Blaster III is driven by $V_{CC(TRGT)}$ so no specific V_{OH} is defined.				V
V_{OL}	Low-level output voltage	I_{OL} = 16 mA	–	0.07	–	V
$I_{CC(TRGT)}$	No load current of $V_{CC(TRGT)}$	–	15	45	85	µA

5.4. JTAG Timing Constraints and Waveforms

Figure 16. Timing Waveform for JTAG Signals (from Target Device Perspective)



Meet the timing constraints for the target device provided in the table below to use the USB Blaster III FPGA Development Cable at the maximum capability, that is 30 MHz.

Table 10. JTAG Timing Constraints for the Target Device

Timing is based on a slow timing model, which is a worst-case scenario environment. Refer to the respective device data sheet for device-specific JTAG timing specifications.

Symbol	Parameter	Minimum (ns)	Maximum (ns)
tJCP	TCK clock period	33.33	–
tJCH	TCK clock high time	16.67	–
tJCL	TCK clock low time	16.67	–
tJPCO	JTAG port to JTAG header output	–	5.95 (1.8 V)

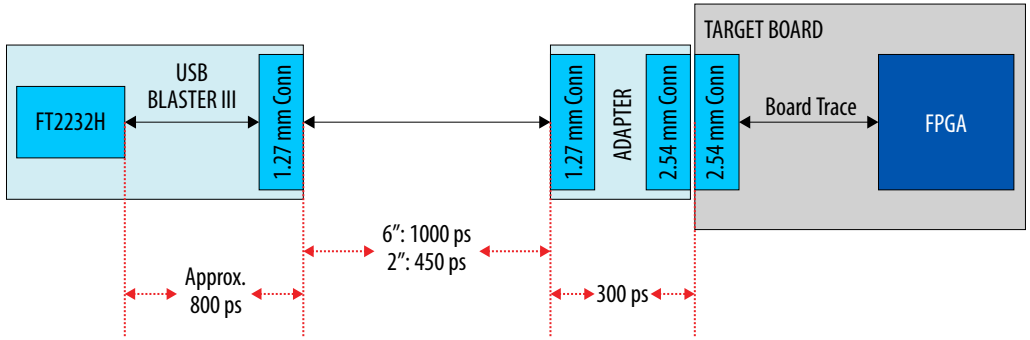
continued...

Symbol	Parameter	Minimum (ns)	Maximum (ns)
tJPSI_TDI	JTAG port setup time (TDI)	16.47	–
tJPSU_TMS	JTAG port setup time (TMS)	16.79	–
tJPH	JTAG port hold time	16.49	–

The timing constraints require that you consider the individual device specifications as well as trace propagation delays. If you do not follow the recommended constraints, you may encounter timing issues at 30 MHz.

If the target design cannot meet these constraints, reduce the possibility of timing issues by slowing the TCK frequency. To learn about running the USB Blaster III FPGA Development Cable at a slower speed or about the auto-adjust frequency feature, refer to the *Changing TCK Frequency with Quartus Prime Programmer* and the *TCK Frequency Auto-Adjust with Quartus Prime Programmer* sections.

Figure 17. USB Blaster III FPGA Development Cable (with Adapter) Timing Constraints



Related Information

- Altera FPGA Documentation
  Your Altera FPGA device's related documentation, including data sheets, can be found in this repository.
- Changing TCK Frequency with Quartus Prime Programmer on page 15
- TCK Frequency Auto-Adjust with Quartus Prime Programmer on page 16

5.4.1. Creating JTAG Timing Constraints in Quartus Prime Software

Creating timing constraints for JTAG in Quartus Prime software involves defining the clock for the JTAG interface and setting input/output delays for the JTAG signals. This ensures that the JTAG communication operates reliably, especially at higher TCK frequencies.

Define the JTAG Clock

The JTAG Test Clock (TCK) is usually considered a virtual clock because it originates from the JTAG download cable, not from an internal FPGA clock source. Define it using the following command:

```
create_clock -name {altera_reserved_tck} -period <TCK_period_in_ns> [get_ports {altera_reserved_tck}]
```

Where: `TCK_period_in_ns` is the period corresponding to your desired TCK frequency.

For example for 15 MHz, the period is 1000/15 = 66.67 ns.

Setting Input and Output Delays for JTAG Signals

You need to specify the maximum and minimum delays for the JTAG signals (TDI, TDO, TMS, TCK) relative to the TCK clock. These values depend on your specific JTAG setup and device characteristics.

- TDI (Test Data In)

```
set_input_delay -max <max_delay_ns> -clock {altera_reserved_tck} [get_ports {altera_reserved_tdi}]
set_input_delay -min <min_delay_ns> -clock {altera_reserved_tck} [get_ports {altera_reserved_tdi}]
```

- TMS (Test Mode Select)

```
set_input_delay -max <max_delay_ns> -clock {altera_reserved_tck} [get_ports {altera_reserved_tms}]
set_input_delay -min <min_delay_ns> -clock {altera_reserved_tck} [get_ports {altera_reserved_tms}]
```

- TDO (Test Data Out)

```
set_input_delay -max <max_delay_ns> -clock {altera_reserved_tck} [get_ports {altera_reserved_tdo}]
set_input_delay -min <min_delay_ns> -clock {altera_reserved_tck} [get_ports {altera_reserved_tdo}]
```

Where: `max_delay_ns` and `min_delay_ns` are external board and cable delays. Review the timing values provided in the *USB Blaster III FPGA Development Cable (with Adapter) Timing Constraints* figure in the *JTAG Timing Constraints and Waveforms* section and refer to the device data sheet for recommended values.

Define False Paths

If certain JTAG paths are not critical for timing analysis, such as paths related to a JTAG debug core that do not impact functional timing, you can declare them as false paths to prevent unnecessary timing analysis on them.

```
set_false_path -from [get_registers {*jtag_debug_core*}] -to [get_registers {*functional_logic*}]
```

Save the command in a Synopsys Design Constraints (SDC) file, for example `jtag_constraints.sdc`. Add this file to your Quartus Prime project so that the Timing Analyzer uses these constraints to analyze and report on the JTAG timing.

Related Information

- Altera FPGA Documentation
  Your Altera FPGA device's related documentation, including data sheets, can be found in this repository.
- JTAG Timing Constraints and Waveforms on page 33

6. Document Revision History for the USB Blaster III FPGA Development Cable User Guide

Document Version	Changes
2026.03.11	Added Stratix III to the *Supported Devices for Configuration Data* table.
2025.12.09	Initial release.

A. Additional Information

A.1. Manually Installing the USB Blaster III FPGA Development Cable Drivers on Windows

You must have system administration (administrator) privileges to install the USB Blaster III FPGA Development Cable drivers.

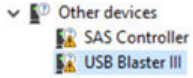
The download cable drivers are included in the Quartus Prime software installer. Before you begin the installation, verify that the USB Blaster III FPGA Development Cable driver is located in your directory: `\<Quartus Prime system directory>\drivers\USB Blaster-iii`.

1. Connect the download cable to your computer's USB port.

   When plugged in for the first time, a message appears stating `Device driver software was not successfully installed`.

2. From the **Windows Device Manager**, locate **Other devices** and right-click one of two **USB Blaster III.**

Figure 18. Other Devices in Windows Device Manager



   You need to install drivers for each interface:

   - JTAG interface
   - System Console interface

3. From the right-click menu, click **Update Driver Software**. The **Update Driver Software - USB Blaster III** dialog box appears.
4. Click **Browse my computer for driver software** to continue.

5. Click **Browse** and navigate to the location of the driver on your system: `\<Quartus Prime system directory>\drivers\USB Blaster-iii`. Click **OK**.
6. Click **Next** to install the driver.
7. Click **Install** when asked whether you want to install.
8. You should now have a JTAG cable showing in your **Windows Device Manager**.

Figure 19. Successful Driver Installation - JTAG Interface

9. Proceed to install the driver for any additional USB Blaster III items listed under **Other Devices** by repeating the steps, beginning with Step 2.

A.2. Binding the FTDI_SIO Driver on Older Linux Kernels

For Linux Kernel Version 6.1, 6.6, 6.12, and above the UARTl automatically binds to the USB Blaster III accordingly. Older versions of Linux Kernel may need to manually bind the FTDI_SIO driver, using either of the following methods.

Method 1 – Manual Binding of the FTDI Driver

1. Execute `lsusb -t` to list the usb tree view.

```
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/4p, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/4p, 480M
    |__ Port 3: Dev 100, If 0, Class=Vendor Specific Class, Driver=, 480M
    |__ Port 3: Dev 100, If 1, Class=Vendor Specific Class, Driver=, 480M
    |__ Port 4: Dev 103, If 0, Class=Vendor Specific Class, Driver=ftdi_sio, 480M
    |__ Port 4: Dev 103, If 1, Class=Vendor Specific Class, Driver=ftdi_sio, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/8p, 480M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=ehci-pci/2p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/6p, 480M
        |__ Port 3: Dev 15, If 0, Class=Vendor Specific Class, Driver=, 480M
        |__ Port 3: Dev 15, If 1, Class=Vendor Specific Class, Driver=, 480M
```

2. Check the `/sys/bus/usb/devices/` folder to find out your device leaf-name.

3. Execute the following command to register our PID/VID to the linux kernel.

```
echo 09FB 6020 | sudo tee /sys/bus/usb-serial/drivers/ftdi_sio/new_id
```

4. Execute the following command to bind the driver to BUS 3 Port 3 Interface 2.

```
echo "3-3:1.1" | sudo tee /sys/bus/usb/drivers/ftdi_sio/bind
```

5. If both interfaces are bound to the FTDI_SIO driver during the previous step , use the following command to unbind the driver from your IF 0, else the first channel bound to the FTDI_SIO driver cannot be used by the USB Blaster III driver.

```
echo "3-3:1.0" | sudo tee /sys/bus/usb/drivers/ftdi_sio/unbind
```

6. Execute `lsusb -t` and validate that the FTDI driver is loaded.

Method 2 – Using udev Rules to Automate FTDI Driver Binding

1. Add `options ftdi_sio vendor=0x09fb product=0x6022` into `/etc/modprobe.d/ftdi.conf`.
2. Create a udev rule to `/etc/udev/rules.d/99-ftdi-uart.rules`.
3. Add this to your udev rule file:

```
ACTION=="add", SUBSYSTEM=="usb", ATTR{idVendor}=="09FB", ATTR{idProduct}=="6020", ATTR{bInterfaceNumber}=="01", RUN
+="/bin/sh -c 'echo $env{DEVPATH} > /sys/bus/usb/drivers/ftdi_sio/bind'"
```

4. Reboot your machine or use this command to reload your udev rules:

```
sudo udevadm control --reload-rules && udevadm trigger
```

5. Execute `lsusb -t` and validate that the FTDI driver is loaded.

A.3. Certification Statements

A.3.1. RoHS Compliance

The table below lists hazardous substances included with the USB Blaster III FPGA Development Cable.

A value of 0 indicates that the concentration of the hazardous substance in all homogeneous materials in the parts is below the relevant threshold as specified by the SJ/T11363-2006 standard.

Table 11. Hazardous Substances and Concentration

Part Name	Lead (Pb)	Cadmium (Cd)	Hexavelent Chromium (Cr6+)	Mercury (Hg)	Polybrominated biphenyls (PBB)	Polybrominated diphenyl Ethers (PBDE)
Electronic Components	0	0	0	0	0	0
Populated Circuit Board	0	0	0	0	0	0
Manufacturing Process	0	0	0	0	0	0
Packing	0	0	0	0	0	0

A.3.2. CE EMI Conformity Caution

This product is delivered conforming to relevant standards mandated by Directive 2004/108/EC. Because of the nature of programmable logic devices, it is possible for the user of this product to modify it in such a way as to generate electromagnetic interference (EMI) that exceeds the limits established for this equipment. Any EMI caused as the result of modifications to the delivered material is the sole responsibility of the user.