

## Introduction

This section provides information about how Altera® FPGAs ensure configuration reliability and suggestions on laying out the configuration interface on your board to avoid configuration problems. The last section provides suggestions on debugging configuration issues.

## Configuration Reliability

The architecture of Altera FPGAs have been designed to minimize the effects of power supply and data noise in a system, and to ensure that the configuration data is not corrupted during configuration or normal user-mode operation. A number of circuit design features are provided to ensure the highest possible level of reliability from this SRAM technology.

Cyclic redundancy code (CRC) circuitry is used to validate each configuration data frame (sequence of data bits) as it is loaded into the target device. If the CRC calculated by the device does not match the CRC stored in the data frame, the configuration process is halted, and the FPGA drives the `nSTATUS` pin low to indicate an error has occurred. CRC circuitry ensures that noisy systems will not cause errors that yield an incorrect or incomplete configuration.

The device architecture also provides a very high level of reliability in low-voltage brown-out conditions. The device's SRAM cells require a certain voltage level to maintain accurate data. This voltage threshold is significantly lower than the voltage required to activate the device's POR circuitry. Therefore, the target device stops operating if the  $V_{CC}$  starts to fail, and indicates an operation error by driving the `nSTATUS` pin low. The device must then be reconfigured before it can resume proper operation as a logic device. In configuration schemes where `nCONFIG` is tied to  $V_{CC}$ , reconfiguration begins as soon as  $V_{CC}$  returns to an acceptable level. The low pulse on `nSTATUS` resets the configuration device by driving `OE` low. In configuration schemes using an external host, the system must start the reconfiguration process by driving `nCONFIG` high after the power supply returns to the minimum operating voltage.

These device features ensure that Altera FPGAs have the highest possible reliability in a wide variety of environments, and provide the same high level of system reliability that exists in other Altera PLDs.

## Board Layout Tips

Even though the `DCLK` signal (used in synchronous configuration schemes) is typically a low frequency signal, it drives edge-triggered pins on the Altera FPGA. Therefore, any overshoot, undershoot, ringing, or other noise can affect configuration. When designing the board, lay out the `DCLK` trace using the same techniques used to lay out a clock line. The same applies for the `TCK` pin. Since `DATA` set-up and hold times are in relation to the `DCLK` signal, make sure these traces are laid out accordingly.

When configuring multiple devices in a configuration chain, Altera recommends that the DCLK, DATA0, (DATA [7 . . 0]), nCONFIG, nSTATUS, and CONF\_DONE signals are tied together for every device in the configuration chain. This ensures that configuration begins and ends at the same time for each device. Additionally, if one device detects an error and pulls nSTATUS low, all devices in the chain will reset and restart configuration. For debugging purposes in your prototyping environment, it may be useful for each device to have its own pull-up to V<sub>CC</sub> for the nSTATUS and CONF\_DONE signals. This will allow you to determine which device is signaling an error during configuration by monitoring each nSTATUS line individually or if one device is not releasing its CONF\_DONE pin.

In multi-device configuration chains, the configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Specifically, ensure that the DCLK and DATA lines are buffered for every fourth device. For multi-device JTAG chains, ensure that the TCK, TDI, and TMS lines are buffered for every device.

When using a configuration device, it is important to realize that after the configuration device sends all its configuration data, it will wait a limited amount of time for its nCS pin (tie to the FPGA's CONF\_DONE pin) to reach a logic high. Enhanced configuration devices wait for 64 DCLK cycles after the last configuration bit was sent for CONF\_DONE to reach a high state. EPC2 devices wait for 16 DCLK cycles. If the configuration device does not see a logic high on nCS after sending all its configuration data, it will signal an error by driving its OE pin (tied to the FPGA's nSTATUS pin) low. Therefore, if there is a long trace length between the nCS and CONF\_DONE pin this could cause a configuration error since the added capacitance of a longer trace contributes to a slower rise time on the CONF\_DONE signal.

## Debugging Suggestions

If you are experiencing problems configuring your Altera FPGA, there are a number of actions you can take to try to identify your problem. If you have not already done so, you should read the appropriate device family chapters.

The following sections provide some suggestions to try if you are encountering configuration problems.

### All Configuration Schemes

- Ensure the configuration file you are using is for the target device on your board. Double check that the SOF used to create the configuration file is for the device on your board by loading the SOF in the Quartus® II programmer and noting the Device column the programmer reports.
- Ensure your board is receiving adequate power to power the FPGA V<sub>CCINT</sub> and the I/O banks where the configuration and JTAG pins reside.
- Double-check that all configuration pins are properly connected as recommended in the appropriate device family chapter. You should check the connections of these pins by probing at the pins of the device, or via under the BGA package (if possible, not on board traces). Specifically, ensure the MSEL and nCE pins are not left floating and are connected as indicated in the appropriate device family chapter. The nCONFIG, nSTATUS, and CONF\_DONE signals require pull-up resistors to V<sub>CC</sub> (either internal or external pull-up resistors).

- If you are not using the JTAG interface, make sure the JTAG pins on the FPGA are not left floating and are connected to a stable level as indicated in the Configuration Pins tables. Because JTAG configuration takes precedence over all other configuration methods, these pins should not be floating or toggling during configuration.
- Try to configure another device on a different board to determine if it is a universal problem which is seen on all boards or on only one device. If another board is not available, you can swap out the device with another device. If you see the problem with only one device, this points to a problem that is device specific. If you see the problem on multiple devices, this indicates that the problem is with the board or with the configuration set up.
- Probe the DCLK signal to ensure it is a clean signal with no overshoot, undershoot or ringing. A noisy DCLK signal could affect configuration and cause a CRC error. For a chain of FPGAs, you should probe each device in the chain as close to the DCLK pin as possible. Noise on any of these pins could cause configuration to fail for the whole chain.
- To check if the FPGA has started accepting configuration data, you can monitor the INIT\_DONE pin. The INIT\_DONE pin is an optional pin and can be turned on in the Quartus II software through the Enable INIT\_DONE output option. The INIT\_DONE pin is an open-drain output and requires an external pull-up to V<sub>CC</sub>. Therefore, when nCONFIG is low and during the beginning of configuration, the INIT\_DONE will be at a logic high level. After the option bit to enable the INIT\_DONE pin is programmed into the FPGA (during the first frame of configuration data), the INIT\_DONE pin will go low. The transition of INIT\_DONE from high to low signals that the FPGA has indeed begun configuration and started to accept configuration data. If the INIT\_DONE pin remains high, the FPGA has not received the proper configuration file header to indicate the beginning of configuration data.
- If the configuration device or external host has sent all configuration data and CONF\_DONE has not gone high, ensure CONF\_DONE has a pull-up to V<sub>CC</sub> and that it is not grounded or driven low on your board.

## Multi-Device Configuration Chains

- You should combine each device's SOF into one configuration file through the **Convert Programming Files** dialog box in the Quartus II software. For more information, refer to the chapter *Configuration File Format*.
- When generating the configuration file, ensure the configuration files are in the same order as the devices on the board.

## Using an External Host (e.g., Microprocessor or CPLD)

- If using a Raw Binary File (.rbf) to configure your Altera FPGA(s), make sure the least significant bit (LSB) of each byte is sent first. If the file is sent in the wrong order, this will cause a configuration error.
- Scope the DATA and DCLK or nWS signals to check that all timing parameters are met as specified in the tables in the appropriate device family chapter.


- If using passive parallel asynchronous (PPA), make sure  $nRS$  is not left floating. This pin should be driven high if it is not used, otherwise configuration errors can occur.

## Using a Configuration Device

- Make sure the configuration device has been programmed successfully. This can be done through the Quartus II programmer by performing a Verify on the configuration device. If the configuration device has not been programmed successfully, it will not configure the FPGA.
- If you notice no DCLK or DATA output from the configuration device, it is possible the configuration device is in a slave mode or idle state. When using a configuration device, the FPGA's  $V_{CCINT}$  supply must be powered up before the configuration device exits POR. If the configuration device exits POR before the FPGA is powered up, the configuration device will enter slave mode (EPC2/EPC1 device) or will enter an idle state (enhanced configuration device).
- To determine if the FPGA is flagging an error by driving the  $nSTATUS$  signal low or if the configuration device is flagging an error by driving the OE signal low, you can separate the  $nSTATUS$  and OE signals on your board (This can also be done by using a logic analyzer and calculating how many DCLK edges have occurred). This should only be done for debugging purposes. If you disconnect the  $nSTATUS$  and OE line, each signal must have a pull-up to  $V_{CC}$ . During configuration, if the FPGA pulls  $nSTATUS$  low, it has seen a CRC error in the configuration data. If the configuration device pulls OE low, it has sent all its configuration data and has not seen the CONF\_DONE signal go high.
- If using an enhanced configuration device, make sure all pins are connected properly. The PGM pins should not be left floating; they should be driven to choose the specific page the configuration file resides. The WP# should be connected to  $V_{CC}$  to enable programming of the bottom boot block. The BYTE# (available in 100-pin packages) should be connected to  $V_{CC}$ , otherwise programming verification will fail; hence resulting in a configuration failure. In the 100-pin package, make sure the following pins are connected externally: F-A0 to C-A0, F-A1 to G-A1, F-A15 to C-A15, and F-A16 to C-A16.
- If using an enhanced configuration device, the external flash interface pins should be floating or tri-stated during in-system programming and during FPGA configuration. For more information, refer to *Enhanced Configuration Devices (EPC4, EPC8 and EPC16) Data Sheet*.
- If using the enhance configuration devices (Altera configuration devices), follow the POR timing for both the enhance configuration device and the FPGA device. The most reliable power-up sequence is to have  $V_{CCIO}$  supply of the enhance configuration device power up before  $V_{CCINT}$  of the FPGA supply; however, the  $V_{CCINT}$  supply should still be powered before the POR of the enhance configuration device expires. This case is known to power up successfully for all supported enhance configuration device and FPGA device combinations.

## Using JTAG Configuration

- Double-check that all JTAG pins are connected as recommended in the JTAG section. Specifically, make sure TRST (if available) is connected to  $V_{CC}$  and that TCK has a pull-down resistor.
- Double-check that all configuration pins are connected as recommended in the JTAG section. Specifically, make sure the nCE pin is connected to GND or driven low during JTAG programming. The nCONFIG pin must be tied to  $V_{CC}$  or driven high. Also, check that CONF\_DONE is not held low by another device. This is important to remember for multi-device chains.
- Ensure the TDO signal drives out a high enough voltage to meet the next device's TDI minimum high-level input voltage ( $V_{IH}$ ). If the TDO pin resides in an I/O bank whose  $V_{CCIO}$  is set to 3.3 V, the TDO pin will drive out 3.3 V.
- When designing the board, layout the TCK trace using the same techniques used to layout a clock line. Any overshoot, undershoot, ringing, or other noise can affect JTAG configuration.
- Probe the TCK signal to ensure it is a clean signal with no overshoot, undershoot, or ringing. For a chain of devices, you should probe each device in the chain as close to the TCK pin as possible. Noise on any of these pins could cause JTAG programming to fail for the entire chain.

 For further help with debugging your configuration issues, visit the online configuration troubleshooter at [www.altera.com](http://www.altera.com)

## Document Revision History

Table 11-1 shows the revision history for this document.

**Table 11-1.** Document Revision History

Date and Revision	Changes Made	Summary of Changes
October 2008, version 2.3	<ul style="list-style-type: none"> <li>■ Updated “Using a Configuration Device” section.</li> <li>■ Updated new document format.</li> </ul>	—
April 2007, version 2.2	<ul style="list-style-type: none"> <li>■ Added document revision history.</li> </ul>	—
August 2005, version 2.1	<ul style="list-style-type: none"> <li>■ Removed active cross references referring to document outside Chapter 11.</li> </ul>	—
July 2004, version 2.0	<ul style="list-style-type: none"> <li>■ Renamed debugging tool to troubleshooter on page 11-6.</li> </ul>	—
September 2003, version 1.0	<ul style="list-style-type: none"> <li>■ Initial Release.</li> </ul>	—