Error Detection and Recovery Using CRC in Altera FPGA Devices

July 2008, Version 1.4 Application Note 357

Introduction

In critical applications, such as avionics, telecommunications, system control, and military applications, it is important to be able to:

- Confirm that the configuration data stored in an FPGA device is correct.
- Alert the system to the occurrence of a configuration error.

Information on SEUs is located in the Products page on the Altera website (**www.altera.com**).

Dedicated circuitry is built into certain devices and consists of a cyclic redundancy check (CRC) error detection feature that can optionally check for SEUs continuously and automatically.

Use of the error detection CRC feature (in the FPGA device) is provided in the Quartus® II software, starting with version 4.1. The Quartus II software CRC feature is supported by the following devices:

- Stratix® IV
- Stratix III
- Stratix II
- Stratix II GX
- Stratix
- Stratix GX
- Cyclone® III
- Cyclone II
- Cyclone

HardCopy®, HardCopy II and Hardcopy III devices do not have configuration circuitry and therefore do not need CRC feature.

This application note describes how to activate and use the error detection CRC feature when a device is in user mode and describes how to recover from configuration errors caused by CRC errors. The content of this application note is only covered the error detection CRC feature for the following devices:

- Stratix II
- Stratix II GX
- Stratix
- Stratix GX

- Cyclone II
- Cyclone

For error detection CRC feature in Stratix IV, Stratix III and Cyclone III devices, please refer to the respective SEU Mitigation chapter in the device handbooks.

This application note includes the following topics:

Error Detection Fundamentals

Error detection determines if the data received through a medium is corrupted during transmission. To accomplish this, the transmitter uses a function to calculate a checksum value for the data and appends the checksum to the original data frame. The receiver uses the same calculation methodology to generate a checksum for the received data frame and compares the received checksum to the transmitted checksum. If the two checksum values are equal, the received data frame is correct and no data corruption occurred during transmission or storage.

The error detection CRC feature uses the same concept. When Stratix and Cyclone series (Stratix II, Stratix II GX, Stratix, Stratix GX, Cyclone II, and Cyclone) devices are in user mode, the error detection CRC feature ensures the integrity of the configuration data.

There are two CRC error checks. One always during configuration and a second optional CRC error check that runs in the background in user mode. This document discusses the user mode CRC error detection feature.

Configuration Error Detection

The error detection CRC feature, available only when the device is in user mode, is an additional feature that functions beyond the frame-based CRC. This feature checks the integrity of the data during the configuration of Stratix and Cyclone series devices. In configuration mode, a frame-based CRC is stored within the configuration data and contains the CRC value for each data frame.

During configuration, the FPGA calculates the CRC value based on the frame of data that is received and compares it against the frame CRC value in the data stream. Configuration continues until either the device detects an error or configuration is complete.

User Mode Error Detection

All Stratix and Cyclone series devices have built-in error detection circuitry to detect data corruption by soft errors in the (configuration random-access memory) CRAM cells.

Soft errors are changes in a CRAM bit's state due to a radiating particle.

When the data bit is rewritten with the correct value by reconfiguring the device, the device functions correctly. This error detection capability continuously computes the CRC of the configured CRAM bits and compares it with the precalculated CRC. If the CRCs match, there is no error in the current configuration CRAM bits. The process of error detection continues until the device is reset (by setting `nCONFIG` low).

The Stratix and Cyclone series device error detection feature does not check memory blocks and I/O buffers. These device memory blocks support parity bits that are used to check the contents of memory blocks for any error. The I/O buffers are not verified during error detection because these bits use flipflops as storage elements that are more resistant to soft errors. Similar flipflops are used to store the precalculated CRC and other error detection circuitry option bits.

The error detection circuitry in Stratix and Cyclone series devices uses a 32-bit CRC IEEE 802 standard, 32-bit polynomial, as the CRC generator. Therefore, a single 32-bit CRC calculation is performed by the device. If a soft error does not occur, the resulting 32-bit signature value will be `0×00000000`, which results in a `0` on the output signal `CRC_ERROR`. If a soft error occurs within the device, the resulting signature value will be non-zero and the `CRC_ERROR` output signal will be `1`.

Stratix II, Stratix II GX, and Cyclone II devices, when in user mode, support the `CHANGE_EDREG` JTAG instruction, which allows you to write to the 32-bit storage register. You can use Jam files (**.jam**) to automate the testing and verification process. This is a powerful design feature that enables you to verify the CRC functionality in-system, on the fly, without having to reconfigure the device. You can then switch to use the CRC circuit to check for real errors induced by an SEU.

You can only execute the CHANGE_EDREG JTAG instruction when the device is in user mode.

Table 1. CHANGE_EDREG JTAG Instruction

JTAG Instruction	Instruction Code	Description
CHANGE_EDREG	00 0001 0101	This instruction connects the 32-bit CRC option register between TDI and TDO. Any precomputed CRC is loaded into the CRC option register to test the operation of the error detection CRC circuitry at the CRC_ERROR pin.

Stratix II and Stratix II GX devices calculate the CRC value during configuration mode. At the end of configuration mode, these devices load the CRC value into the 32-bit storage register.

For Stratix, Stratix GX, Cyclone II, and Cyclone devices, the CRC is computed by the Quartus II software and downloaded into the device as part of the configuration bit stream. These devices store the CRC in the 32-bit storage register at the end of the configuration mode.

When the error detection CRC feature is enabled in user mode, these devices recalculate the CRC based on the contents of the device and compare it against the 32-bit storage register.

CRC_ERROR Pin-Outs

Table 2 describes the CRC_ERROR pin. Tables 3 and 4 show the CRC_ERROR pin locations for the Stratix and Stratix GX device families.

Table 2. CRC_ERROR Pin Description

Pin Name	Pin Type	Description
CRC_ERROR	I/O, output	Active high signal that indicates that the error detection circuit has detected errors in the configuration CRAM bits. This pin is optional and is used when the error detection CRC circuit is enabled. When the error detection CRC circuit is disabled, it is a user I/O pin. The CRC error output, when using the WYSIWYG function, is a dedicated path to the CRC_ERROR pin. The CRC_ERROR pin does not support open-drain or inversion.

WYSIWYG (What You See Is What You Get) is an optimization technique which performs optimization on VQM (Verilog Quartus Mapping) netlist within the Quartus II software.

Table 3. CRC_ERROR Pin Table for Stratix Devices

Device	Device Package						
	484-Pin FineLine BGA *(1)*	672-Pin BGA	672-Pin FineLine BGA	780-Pin FineLine BGA	956-Pin BGA	1,020-Pin FineLine BGA	1,508-Pin FineLine BGA
EP1S10 *(2)*	N14	W10	W10	AA20	—	—	—
EP1S20	N14	W10	W10	AA20	—	—	—
EP1S25	—	W10	W10	AA20	—	AF20	—
EP1S30	—	—	—	AA20	AE21	AF20	—
EP1S40	—	—	—	AA20	AE21	AF20	AN25
EP1S60	—	—	—	—	AE21	AF20	AN25
EP1S80	—	—	—	—	AE21	AF20	AN25

Notes to Table 3:
(1) FineLine BGA is a FineLine Ball Grid Array package type.
(2) EP1S10 engineering sample (ES) devices do not support the error detection feature.

Table 4. CRC_ERROR Pin Table for Stratix GX Devices

Device	Device Package	
	672-Pin FineLine BGA	1,020-Pin FineLine BGA
EP1SGX10C	U15	—
EP1SGX10D	U15	—
EP1SGX25C	U15	—
EP1SGX25D	U15	AE21
EP1SGX25F	—	AE21
EP1SGX40D	—	AE21
EP1SGX40G	—	AE21

The CRC_ERROR pin information for Stratix II, Stratix II GX, Cyclone II, and Cyclone devices is reported in the Device Pin-Outs on the Literature page of the Altera website (**www.altera.com**).

Error Detection Block

You can enable the Stratix or Cyclone series device error detection block in the Quartus II software (see "Software Support" on page 11). This block contains the logic necessary to calculate the 32-bit CRC signature for the configuration CRAM bits in the device.

The CRC circuit continues running even if an error occurs. When a soft error occurs, the device sets the CRC_ERROR pin high. Two types of CRC detection check the configuration bits:

- The first type is the CRAM error checking ability (32-bit CRC) during user mode, for use by the CRC_ERROR pin. There is only one 32-bit CRC value and this 32-bit CRC value covers all of the CRAM data.
- The second type is the 16-bit CRC that is embedded in every configuration data frame. During configuration, after a frame of data is loaded into the FPGA, the precomputed CRC is shifted into the CRC circuitry. At the same time, the CRC value for the data frame shifted-in is calculated. If the precomputed CRC and calculated CRC values do not match, then nSTATUS is set low. Every data frame has a 16-bit CRC, and therefore, there are many 16-bit CRC values for the whole configuration bitstream. Every device has different length of the configuration data frame.

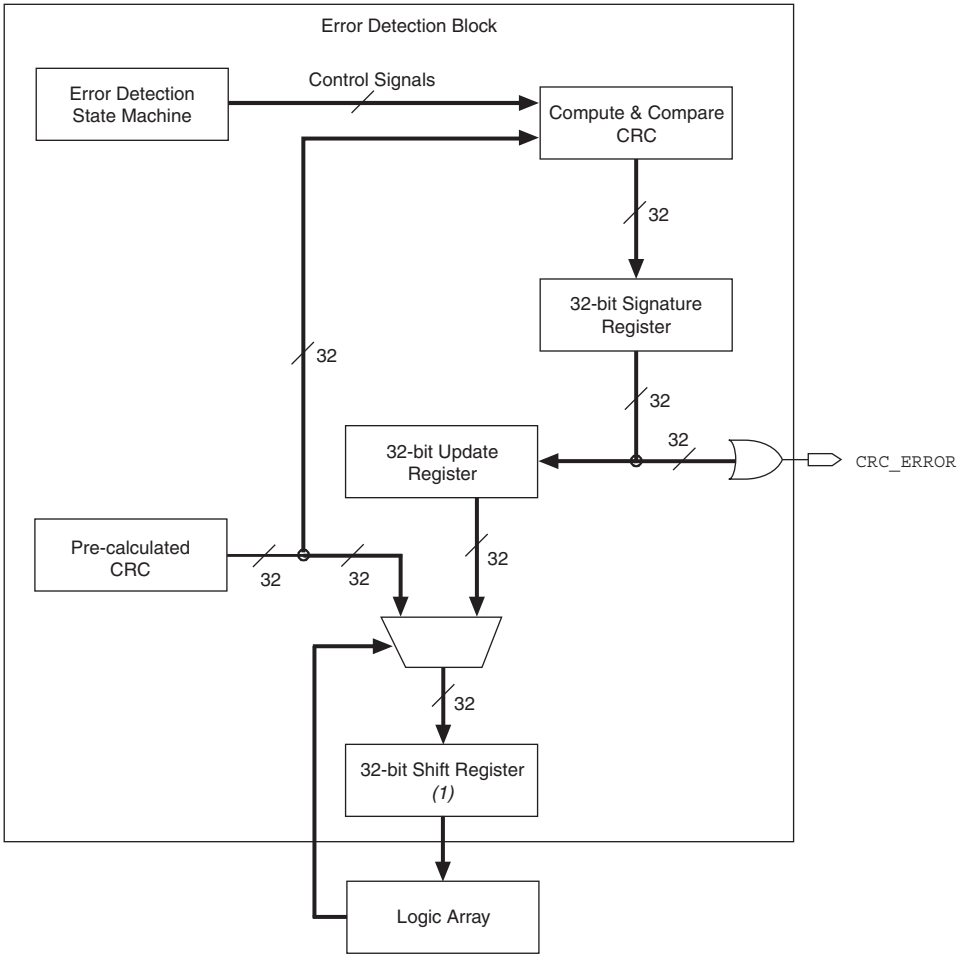
This application note focuses on the first type, the 32-bit CRC only when the device is in user mode.

Error Detection Registers

There are two sets of 32-bit registers in the error detection circuitry that store the computed CRC signature and precalculated CRC value. A nonzero value on the signature register causes the CRC_ERROR pin to set high.

Figure 1 shows the block diagram of the error detection block and the related 32-bit registers:

Figure 1. Error Detection Block Diagram



Note to Figure 1:
(1) You need to clock 31 cycles of clk signal to read out the 32-bit shift register.

Table 5 defines the registers shown in Figure 1.

Table 5. Error Detection Registers

Register	Function
32-bit signature register	This register contains the CRC signature. The signature register contains the result of the user mode calculated CRC value compared against the precalculated CRC value. If no errors are detected, the signature register is all-zeros. A non-zero signature register indicates an error in the configuration CRAM contents. The CRC_ERROR signal is derived from the contents of this register.
32-bit storage register	This register is loaded with the 32-bit precomputed CRC signature at the end of the configuration stage. The signature is then loaded into the 32-bit CRC circuit (called Compute and Compare CRC block as shown in Figure 1) during user mode to calculate the CRC error. This register forms a 32-bit scan chain during execution of the CHANGE_EDREG JTAG instruction. The CHANGE_EDREG JTAG instruction can change the content of the storage register. Hence, the functionality of the error detection CRC circuitry is checked in-system by executing the instruction to inject an error during the operation. The operation of the device is not halted when issuing the CHANGE_EDREG instruction.
32-bit update register	This register is automatically updated with the contents of the signature register, one cycle after the signature register's content are valid. This ensures that the update register is not written by the contents of the signature register at exactly the same time when the shift register is reading its contents.
32-bit shift register	The logic array can access this register. This register also allows user logic to read the contents of the update register or the register containing the pre-calculated CRC value.

Error Detection Timing

When you complete the device configuration and place the device in user mode, you can execute the CHANGE_EDREG JTAG instruction. If there is no soft error, the CRC_ERROR pin is set low before the command executes. You can inject a soft error by changing the 32-bit CRC option register in the error detection CRC circuitry. The CRC_ERROR pin is set high, which flags an error. After verifying the failure induced by the changed CRC value, you can restore the 32-bit CRC value to the correct CRC value using the same instruction and inserting the correct value. Read out the correct value first before updating it with a known bad value. Additionally, you can create Jam files to automate this process.

When the error detection CRC feature is enabled through the Quartus II software, the device automatically activates the CRC process upon entering user mode after configuration and initialization is complete. The CRC_ERROR pin is driven low until an error occurs. The CRC_ERROR pin is driven high when the error detection circuitry has detected corrupted bit(s) in the previous CRC calculation.

After the test completes, Altera recommends that you reconfigure the device (since you changed the CRC option register).

Once the CRC_ERROR pin goes high, it remains high even during the next CRC calculation. This pin does not keep a record of the previous CRC calculation. If the new CRC calculation does not contain any corrupted bits, the CRC_ERROR pin is driven low. The error detection runs until the device is reset.

As soon as the device enters user mode, the error detection CRC feature process is enabled. The error detection circuitry runs off the configuration oscillator. The CRC circuitry runs off an internal configuration oscillator with a divisor that sets the maximum frequency. Table 6 shows the minimum and maximum error detection frequencies.

Table 6. Minimum and Maximum Error Detection Frequencies

Device Type	Error Detection Frequency	Maximum Error Detection Frequency	Minimum Error Detection Frequency	Valid Divisors (2^n)
Stratix and Stratix GX	100 MHz/2^n	100 MHz	390 kHz	0, 1, 2, 3, 4, 5, 6, 7, 8
Stratix II and Stratix II GX	100 MHz/2^n	50 MHz	390 kHz	1, 2, 3, 4, 5, 6, 7, 8
Cyclone and Cyclone II	80 MHz/2^n	80 MHz	312.5 kHz	0, 1, 2, 3, 4, 5, 6, 7, 8

You can set a lower clock frequency by specifying a division factor in the Quartus II software (refer to "Software Support" on page 11). The divisor is a power of two (2^n), where n is between 0 and 8. The divisor ranges from 1 through 256. See the following equation:

$$\text{Error detection frequency} = \frac{100\ \text{MHz}}{2^n}$$

The time it takes for each CRC calculation depends on the device and the error detection clock frequency. Table 7 shows the estimated time for each CRC calculation with minimum and maximum clock frequencies for Stratix and Cyclone series devices.

Table 7. CRC Calculation Time (Part 1 of 2)

Device	Minimum Time (ms) *Note (1)*	Maximum Time (s) *Note (2)*
Stratix II GX Devices		
EP2SGX30	22	2.816
EP2SGX60	39	4.992
EP2SGX90	59	7.552
EP2SGX130	86	11.008
Stratix II Devices		
EP2S15	10	1.28
EP2S30	22	2.816
EP2S60	39	4.992
EP2S90	59	7.552
EP2S130	86	11.008
EP2S180	115	14.72
Stratix Devices		
EP1S10	4.3	1.1
EP1S20	7.2	1.9
EP1S25	9.8	2.5
EP1S30	12.8	3.3
EP1S40	15.3	3.9
EP1S60	21.7	5.6
EP1S80	29.6	7.6
Cyclone Devices		
EP1C3	0.92	0.24
EP1C4	1.3	0.32
EP1C6	1.8	0.45
EP1C12	3.5	0.90
EP1C20	5.4	1.4

Table 7. CRC Calculation Time (Part 2 of 2)

Device	Minimum Time (ms) *Note (1)*	Maximum Time (s) *Note (2)*
Stratix GX Devices		
EP1SGX10C	4.3	1.1
EP1SGX10D	4.3	1.1
EP1SGX25C	9.8	2.5
EP1SGX25D	9.8	2.5
EP1SGX25F	9.8	2.5
EP1SGX40D	15.3	3.9
EP1SGX40G	15.3	3.9
Cyclone II Devices		
EP2C5	2	0.512
EP2C8	3	0.768
EP2C20	6	1.536
EP2C35	11	2.816
EP2C50	16	4.096
EP2C70	23	5.888

Notes to Table 7:

(1) Minimum time corresponds to the maximum error detection clock frequency and may vary with different processes, voltages, and temperatures.

(2) Maximum time corresponds to the minimum error detection clock frequency and may vary with different processes, voltages, and temperatures.

Software Support

The Quartus II software, starting with version 4.1, supports the error detection CRC feature. Enabling this feature generates the `CRC_ERROR` output to the optional dual purpose `CRC_ERROR` pin.

The error detection CRC feature is controlled by the **Device & Pin Options** dialog box in the Quartus II software and uses a 32-bit CRC circuit to ensure data reliability.

The error detection feature using CRC is enabled by performing the following steps:

1. Open the Quartus II software and load a project using a Stratix or Cyclone series device.

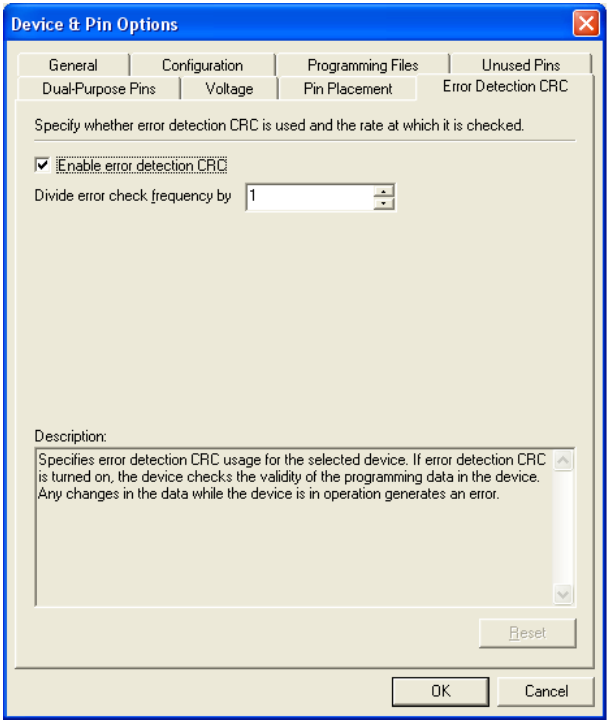
2. On the Assignments menu, click **Settings**. The **Settings** dialog box is shown.

3. In the **Category** list, select **Device**. The **Device** page is shown.
4. Click **Device & Pin Options**. The **Device & Pin Options** dialog box is shown (see Figure 2).
5. In the **Device & Pin Options** dialog box, click the **Error Detection CRC** tab.
6. Turn on **Enable error detection CRC**.
7. In the **Divide error check frequency by** box, enter a valid divisor as documented in Table 6 on page 9.

The divide value divides down the frequency of the configuration oscillator output clock that clocks the CRC circuitry.

8. Click **OK**.

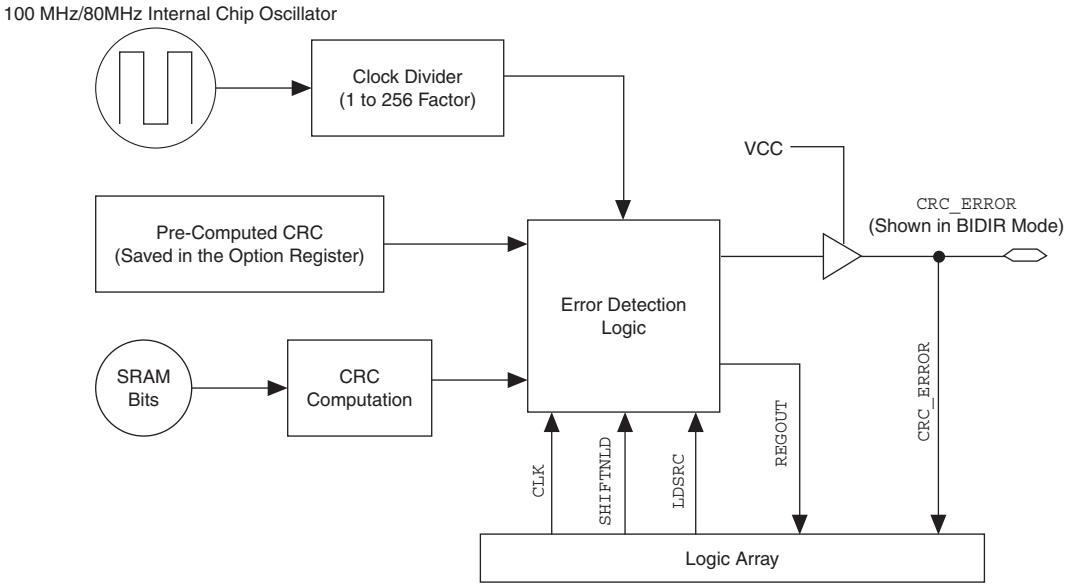
Figure 2. Enabling the Error Detection CRC Feature in the Quartus II Software



Accessing Error Detection Block through User Logic

The error detection circuit contains certain registers to store the computed 32-bit CRC signature, and to allow this signature to be read out by user logic.You have to access to logic array in order to interface user logic with the error detection circuit. `<device>_crcblock` is a WYSIWYG component used to establish the interface from user logic to the error detection circuit. The `<device>_crcblock` primitive atom contains the input and output ports that must be included in the atom. To access the logic array, the `<device>_crcblock` WYSIWYG atom must be inserted into the design. Figure 3 shows the error detection block diagram in FPGA devices and shows the interface which the WYSIWYG atom enables in your design.

Figure 3. Error Detection Block Diagram



The user logic can be affected by the soft error failure, thus reading out the 32-bit CRC signature through the **Shift Register** should not be relied upon to detect a soft error. You should rely on the `CRC_ERROR` output signal itself, since this `CRC_ERROR` output signal can not be affected by soft error.

To include the `<device>_crcblock` WYSIWYG atom in your design, you must also enable the error detection CRC feature in the **Device & Pin Options** dialog box in the Quartus II software.

To enable <device>_crcblock WYSIWYG atom, you have to name the atom for each device accordingly. Table 8 shows the name of the WYSIWYG atom for each device. The input and output ports of the atoms for Stratix II, Stratix II GX, Stratix, Stratix GX, Cyclone II, and Cyclone devices are similar. Table 9 shows the input and output ports that must be included in the atom. Following example shows the input and output ports of a WYSIWYG atom of a Stratix device.

Example 1–1.

```
stratix_crcblock <crcblock_name>

(

    .clk(<clock source>),

    .shiftnld(<shiftnld source>),

    .ldsrc(<ldsrc source>),

    .crcerror(<crcerror out destination>),

    .regout(<output destination>)

);
```

Table 8. WYSIWYG atoms

Device	WYSIWYG atom
Stratix II GX	stratixiigx_crcblock
Stratix II	stratixii_crcblock
Stratix GX	stratixgx_crcblock
Stratix	stratix_crcblock
Cyclone III	cycloneiii_crcblock
Cyclone II	cycloneii_crcblock
Cyclone	cyclone_crcblock

Table 9. CRC Block Input And Output Ports

Port	Input/ Output	Definition
<crcblock_name>	Input	Unique identifier for the CRC block and represents any identifier name that is legal for the given description language such as Verilog HDL, VHDL, AHDL. This field is required.
.clk(<clock source>	Input	This signal designates the clock input of this cell. All operations of this cell are with respect to the rising edge of the clock. Whether it is the loading of the data into the cell or data out of the cell, it always occurs on the rising edge. This port is required.
.shiftnld (<shiftnld source>)	Input	This signal is an input into the error detection block. If shiftnld=1, the data is shifted from the internal shift register to the regout at each rising edge of clk. If shiftnld=0, the shift register parallel loads either the pre-calculated CRC value or the update register contents depending on the ldsrc port input. This port is required.
.ldsrc (<ldsrc source>)	Input	This signal is an input into the error detection block. If ldsrc=0, the pre-computed CRC register is selected for loading into the 32-bit shift register at the rising edge of clk when shiftnld=0. If ldsrc=1, the signature register (result of the CRC calculation) is selected for loading into the shift register at the rising edge of clk when shiftnld=0. This port is ignored when shiftnld=1. This port is required.
.crcerror (<crcerror out destination>)	Output	This signal is the output of the cell that is synchronized to the internal oscillator of the device (100-MHz or 80-MHz internal oscillator) and not to the clk port. It asserts automatically high if the error block detects that a SRAM bit has flipped and the internal CRC computation has shown a difference with respect to the pre-computed value. This signal must be connected either to an output pin or a bidirectional pin. If it is connected to an output pin, you can only monitor the CRC_ERROR pin (the core cannot access this output). If the CRC_ERROR signal is used by core logic to read error detection logic, this signal must be connected to a BIDIR pin. The signal is fed to the core indirectly by feeding a BIDIR pin that has its oe port connected to V_{CC} (see Figure 3).
.regout (<output destination>)	Output	This signal is the output of the error detection shift register synchronized to the clk port, to be read by core logic. It shifts one bit at each cycle. User should clock the clk signal 31 cycles to read out the 32 bits of the shift register. The values at the .regout port are an inversion of the actual values.

Recovering from CRC Errors

The system that the Altera FPGA resides in must control device reconfiguration. After detecting an error on the CRC_ERROR pin, system by strobing nCONFIG low directs the time to perform the reconfiguration at a time when it is safe for the Quartus II software to reconfigure the FPGA.

In the Quartus II software, the error detection CRC feature uses a 32-bit CRC circuit to ensure data reliability and triggers a reconfiguration when affected by an SEU.

While soft errors are uncommon in Altera FPGAs, certain high-reliability applications may require a design to account for these errors.

Conclusion

The purpose of the error detection CRC feature is to detect a flip in any of the configuration CRAM bits in Stratix or Cyclone series devices due to a soft error. By using the error detection circuitry, you can continuously verify the integrity of the configuration CRAM bits.

Document Revision History

Table 10 shows the revision history for this application note.

Table 10. Document Revision History (Part 1 of 2)

Date and Document Version	Changes Made	Summary of Changes
July 2008, v.1.4	■ Updated "Introduction" section. ■ Updated "Table 2", "Table 5", "Table 7", and ■ "Table 9". ■ Updated "Figure 1" and "Figure 3". ■ Updated "Software Support", "Accessing Error Detection Block through User Logic" and "User Mode Error Detection" sections. ■ Added "Table 8".	—

Table 10. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
June 2008, v.1.3	■ Updated "Introduction" and "Error Detection Registers". ■ Updated "Figure 1". ■ Updated "Table 5" and "Table 7". ■ Added new section "Accessing Error Detection Block through User Logic". ■ Added "Figures 3". ■ Added "Table 9".	—
January 2007, v.1.3	■ Minor text edits to Introduction section. ■ Added Document Revision History section.	—

101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support:
www.altera.com/support/
Literature Services:
literature@altera.com

Copyright © 2008 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

I.S. EN ISO 9001