CrossLink-NX Soft Error Detection (SED)/Correction (SEC) Usage Guide

Technical Note

FPGA-TN-02076-1.0

February 2020

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CRC	Cyclic Redundancy Check
ECC	Error Correcting Code
DRAM	Dynamic Random Access Memory
IP	Intellectual Property
PLD	Programmable Logic Device
SED	Soft Error Detection
SEC	Soft Error Correction
SEI	Soft Error Injection
SRAM	Static Random Access Memory

1. Introduction

Memory errors can occur when high-energy charged particles alter the stored charge in a memory cell in an electronic circuit. The phenomenon first became an issue in Dynamic Random Access Memory (DRAM), requiring error detection and correction for large memory systems in high-reliability applications. As device geometries have continued to shrink, the probability of memory errors in Static Random Access Memory (SRAM) has become significant for some systems. Designers are using a variety of approaches to minimize the effects of memory errors on system behavior. CrossLink-NX™ devices are unique due to the underlying technology used to build these devices is much more robust and less prone to soft errors.

SRAM-based programmable logic devices (PLDs) store logic configuration data in SRAM cells. As the number and density of SRAM cells in a PLD increase, the probability that a memory error alters the programmed logical behavior of the system increases. A number of traditional approaches have been taken to address this issue, but most involve soft Intellectual Property (IP) cores that you instantiate into the logic of your design, utilizing valuable resources and possibly affecting design performance.

CrossLink-NX devices have an improved hardware implemented Soft Error Detection (SED) circuit which can be used to detect SRAM errors and allow them to be corrected. There are two layers of SED implemented in CrossLink-NX making it more robust and reliable.

This document describes the hard-logic based SED approach taken by Lattice Semiconductor for CrossLink-NX devices. Once soft error is detected, Lattice provides an easy way to optionally perform the Soft Error Correction (SEC) without disturbing the functionality of the device.

2. SED Overview

The SED module in the CrossLink-NX device is an enhanced version as compared to the SED modules implemented in other Lattice devices. Enhancements include:

- Frame by Frame SED check
- Single bit and Multi-bit error detection
- ECC to correct single bit error at the frame level
- Programmable SED clock with a wider clock frequency option

The Crosslink-NX device is based on the Lattice Nexus platform which is developed using FDSOI technology. FDSOI transistors have very less active region which helps to reduce the bit flipping when exposed to alpha and neutron particles.

Some of the key advantages of Crosslink-NX device as compared to other devices are:

- Improved Radiation Tolerance due to reduction in critical area separated by thin-layer of buried oxide (BOX)
- 100x improvement in soft errors

Due to the above technology, the Crosslink-NX device has extremely low bit error rate and fit rate. Details about the fit rate calculation can be found in the Single Event Upset (SEU) Report for CrossLink-NX (FPGA-TN-02174).

This SED module is part of the Configuration block in the CrossLink-NX devices. The configuration data is divided into frames so that the FPGA can be programmed as a whole or in precise parts. The SED hardware reads serial data from the FPGAs configuration memory in the background while the device is in User mode and performs Error Correcting Code (ECC) calculation on every frame of configuration data (see Figure 2.1). Once a single bit of error is detected, Soft Error Upset (SEU), a notification is generated and SED resumes operation. For single bit errors, the corrected value is rewritten to the particular frame using ECC information. If more than one bit error is detected within one frame of configuration data, an error message is generated. In parallel, cyclic redundancy check (CRC) is calculated for the entire bitstream along with ECC.

After the ECC is calculated on all frames of configuration data, cyclic redundancy check (CRC) is calculated for the entire configuration data (bitstream).

Due to the dynamic contents of memories, the CRC and ECC calculations do not include EBR and Large RAM memory. Dynamic RAM should not be used with SED, otherwise SED reports failures.

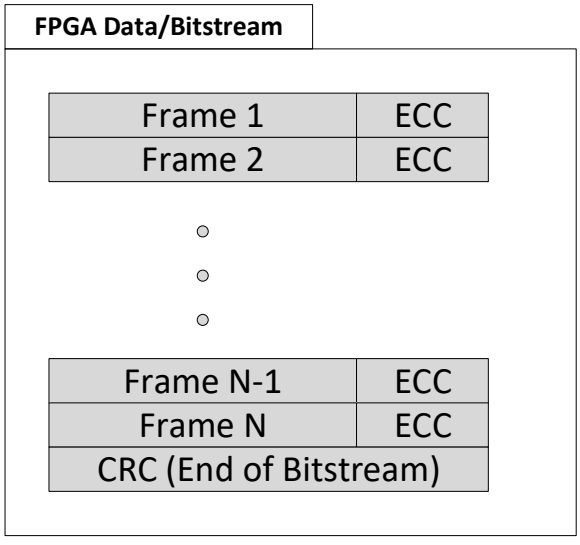

Figure 2.1. CrossLink-NX Bitstream Data Structure

The SED/SEC IP is part of the sysConfig block of the CrossLink-NX device. Figure 2.2 shows the system-level view of the SED/SEC IP.

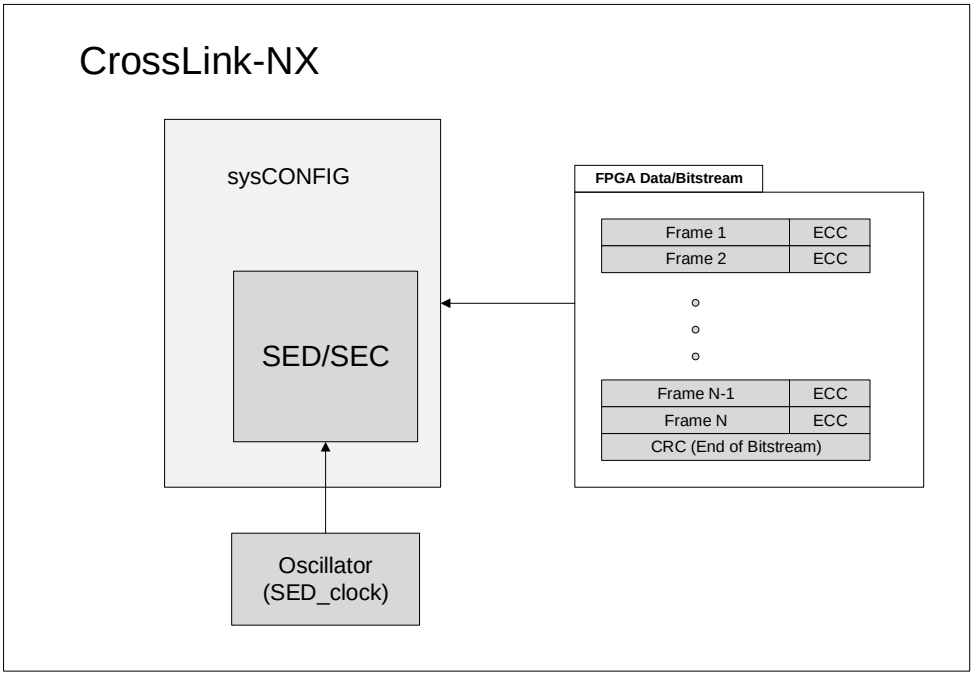

Figure 2.2. CrossLink-NX SED/SEC System Block Diagram

2.1. Block Diagram

The SED block in the CrossLink-NX device contains a number of inputs that control the actual SED block behavior. There are a number of modes that this SED block can operate in. A high-level block diagram showing the user input and output ports is shown in Figure 2.3.

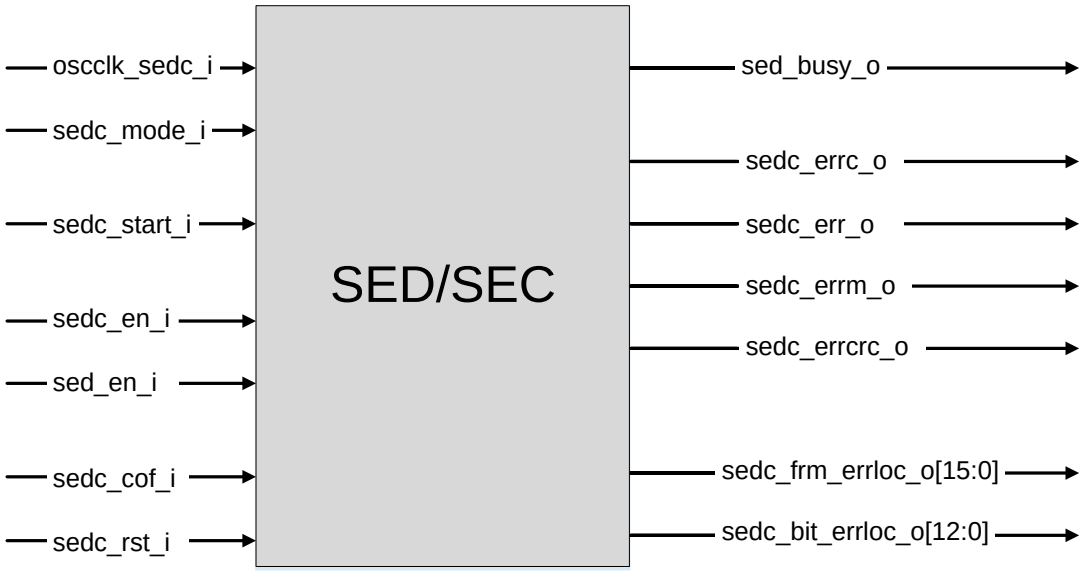

Figure 2.3. SED Block Diagram

3. Port Descriptions

To use the SED IP, instantiate the SED IP as well as the Oscillator primitive using the Lattice Radiant IP Catalog. Refer to Figure 2.1 that shows how to connect the Oscillator to the SEDC IP. Below is a list of port signals used by the SED IP.

Table 3.1. SED Primitive Port Definitions

Port Name	Default Value	Active	Description
osclk_sedc_i	0	Input	User input clock for SED
sedc_mode_i	0	Input	SED mode signal, select between two modes
sedc_start_i	0	Input	SED signal used to start SED
sedc_en_i	0	Input	Signal to enable SED
sed_en_i	0	Input	Signal to enable Soft Error Correction (SEC)
sedc_cof_i	0	Input	SED continue on failure signal
sedc_rst_i	0	Input	Reset the SED IP
sedc_busy_o	0	Output	Signal to indicate SED is in progress
sedc_errc_o	0	Output	Indicate Current SED error
sedc_err_o	0	Output	1-bit (correctable) error detected; sticky bit
sedc_errm_o	0	Output	Multi-bit error detected within one frame; sticky bit
sedc_errcrc_o	0	Output	CRC error for entire bitstream data
sedc_frm_errloc_o[15:0]	0	Output	Frame Error location
sedc_dsr_errloc_o[12:0]	0	Output	Bit error location within a frame

4. Port Descriptions

4.1. Oscclk_sedc_i

The oscclk_sedc_i is a user-selectable clock signal used to run the SED IP. Table 5.1 defines the various clock divider settings that can be used to define the desired clock speed of the SED block.

4.2. sedc_mode_i

sedc_mode_i signal is used to choose the SED mode of operation. There are two SED modes that you can choose from: continuous mode and one-shot mode. For continuous mode, the sedc_mode_i signal is high and once sedc_start_i is HIGH, the SED operation keeps running continuously. For one-shot mode, the sedc_mode_i signal is low and once the SED module detects the low-to-high transition on the sedc_start_i signal, the SED operation runs once. For one-shot mode to operate correctly, you need to make sure sedc_start_i remains asserted until sedc_busy_o is deasserted. In other words, there should not be glitches in the sedc_start_i signal.

4.3. sedc_start_i

This sedc_start_i signal is used to start the SED operation. Once the sedc_start_i signal goes high, the SED cycle starts if sedc_en_i is high. In continuous mode, the sedc_start_i signal must remain high for the duration of SED.. If sedc_start_i goes low during the SED cycle, the process is terminated and sedc_busy_o is deasserted.

4.4. sedc_en_i

The sedc_en_i signal is used to enable SED. SED will not operate, and any in-progress operation aborted, if sedc_en_i is deasserted.

4.5. sed_en_i

If sed_en_i is set, the soft error correction is performed immediately as soon as a single correctable error is detected. If this bit is disabled, the correction is not done.

4.6. sedc_cof_i

The sedc_cof_i stands for SED_continue_on_failure. This signal is used to tell the SED module to run or stop after an error is detected. If sedc_cof_i signal is set to HIGH, the SED operation continues even if an error is encountered. On the other hand, if sedc_cof_i signal is LOW, the SED operation is terminated as soon as an error is detected.

4.7. sedc_rst_i

The sedc_rst_i signal is used to reset the SED IP. This is an asynchronous reset signal.

4.8. sedc_busy_o

The sedc_busy_o signal indicates if SED operation is currently in progress. If the SED is running, the sedc_buys_o is set to HIGH. Once SED operation is complete, this signal goes LOW.

4.9. sedc_errc_o

The SED error current flag indicates if a soft error is detected. As soon as an error is detected, this flag goes high indicating it is a current error. This flag is not sticky.

4.10. sedc_err_o

The sedc_err_o flag is used to indicate if there is a single bit error in a frame. This flag is sticky. To clear this flag, the SED operation has to be disabled.

This single bit error detected is also correctable. The correction is performed only if the sed_en_i signal is set.

4.11. sedc_errm_o

The SED error multiple is used to indicate if 2 or more errors are detected. Once multi-bit errors are detected this flag goes high and those errors are not correctable. This flag is sticky. To clear this flag, the SED module has to be disabled.

4.12. sedc_errcrc_o

The SED error crc indicates if there is a mismatch between calculated CRC of the bitstream as compared to the expected CRC. This error is generated once all the frames of bitstream are read. If there is a single bit error detected, this flag is set. Once the single bit error is corrected, this crc flag is cleared when SED operation runs for the second time.

4.13. sedc_frm_errloc_o[15:0]

The SED frame Error location reports the last location of the frame that errored out. It only provides the frame location for the last 1-bit error. This signal reports the 16-bit error location for the frame that is causing error. This signal is only used for information purposes which can be used for further analysis of the SED errors.

Note: This signal is only valid when SEC is enabled (sed_en_i is true).

4.14. sedc_dsr_errloc_o[12:0]

The SED bit error location reports the bit position in a particular frame that errored out. This information is useful so that the user can perform detailed analysis of the bitstream (on a bit-by-bit basis).

Note: This signal is only valid when SEC is enabled.

5. SED Clock Driver

The SED circuitry is driven by the CrossLink-NX internal oscillator. You have to instantiate the oscillator IP along with the SEDC IP from the Lattice Radiant IP catalog.

The default oscillator frequency is 225 MHz. You can choose to lower the oscillator frequency by configuring the oscillator IP using the Lattice Radiant IP catalog. You can set the SEDCLK_divider setting anywhere between 2 to 256 in integer increments resulting frequency range from 225 MHz to 1.76 MHz (SED Oscillator Frequency = 450 MHz/SEDCLK_divider).

Table 5.1. SED Internal Oscillator Divider Settings

Divider Setting	SEDCLK_Divider	Oscillator Frequency (MHz)
Divide by 2	2	225
Divide by 3	3	150
Divide by 4	4	112.5
—	—	—
—	—	—
—	—	—
Divide by 256	256	1.76

6. SED Flow

This section describes the SED flow. The SED flow will be executed once V_{CC} reaches the data sheet V_{CC} minimum recommended level and sedc_en_i and sedc_start_i are asserted.

The CrossLink-NX device has an advanced SED flow with two levels of SED checks. In the first level of SED check, the bitstream is read one frame at a time and the SED check is performed on a frame-by-frame basis. After all frames of the CrossLink-NX bitstream are read, the SED module checks for CRC of the entire bitstream, second level SED, to check for the bitstream integrity giving the CrossLink-NX device improved SED performance. Figure 6.1 shows the SED flow in CrossLink-NX devices.

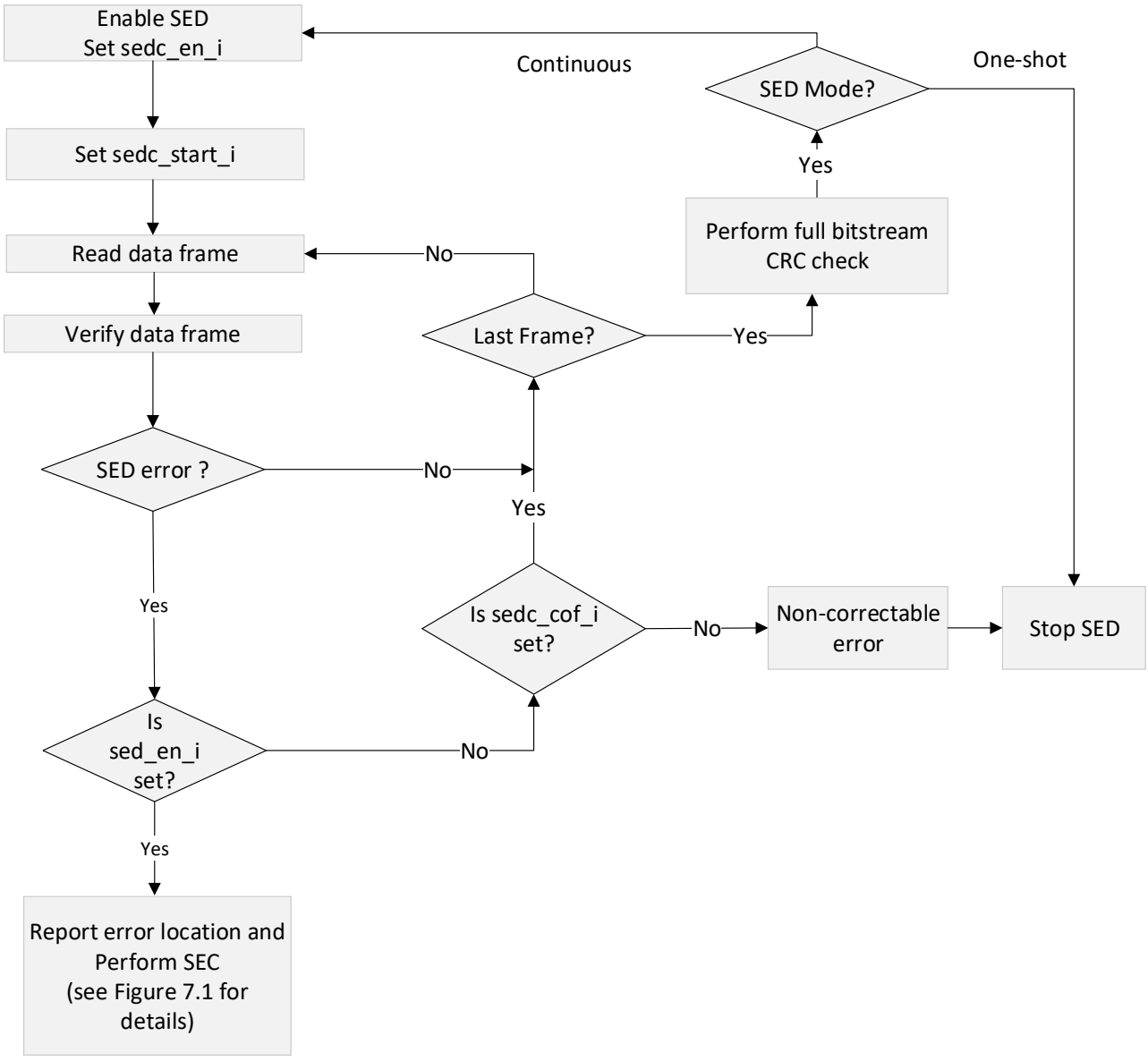

Figure 6.1. SED Flow

6.1. SED Mode

The CrossLink-NX device supports two different SED modes. This provides you the flexibity to run the SED. The first mode is the continuous mode in which the SED runs continuously. The other mode is one-shot mode in which SED runs once for each assertion of sedc_start_i signal.

6.1.1. Continuous Mode

As the name suggests, in continuous mode, the SED runs continuously as long as the sedc_start_i signal is high.

1. Once the SED is enabled, it starts reading bitstream data frame by frame and verifies if the data is read correctly from configuration SRAM. The sedc_busy_o signal is HIGH as long as SED is running.
2. Once SED finishes checking, the sedc_busy_o goes LOW (once the SED cycles through for the first time).
3. The SED cycles through for the second time as long as sedc_start_i is HIGH since the operation is in Continuous Mode.

Once sedc_mode_i is set to 1 and sedc_start_i is always HIGH, the SED operation runs continuously, as shown in Figure 6.2.

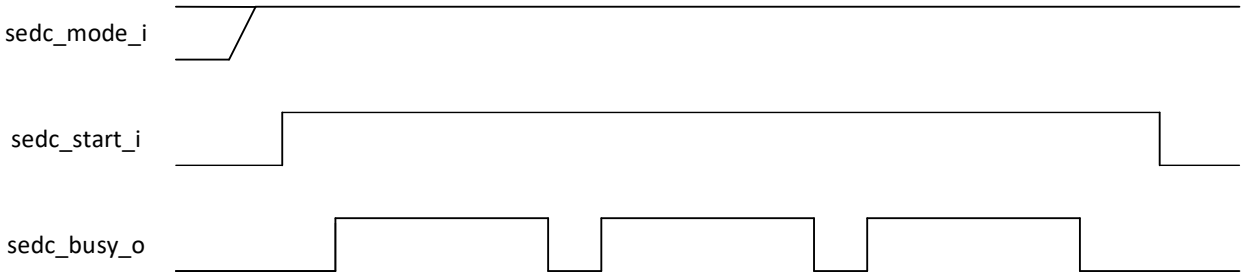

Figure 6.2. SED Continuous Mode

6.1.2. One-shot Mode

In this mode, the SED runs once for each assertion of sedc_start_i signal.

1. For One-shot Mode, the sedc_start_i signal must have a LOW to HIGH transition to start the SED operation.
2. The SED starts reading bitstream data frame by frame and verifies if the data is read correctly from configuration SRAM. The sedc_busy_o signal is HIGH as long as SED is running.
3. The SED finishes checking. The SED error flags are updated and the sedc_busy_o flag goes LOW. Another SED cycle is started by making a LOW to HIGH transition on the sedc_start_i signal.

   Note: If there is any error, disable the sedc_en_i signal to reset all error flags.

In this mode, the sedc_mode_i signal is set to zero. As soon as there is a low to high transition on the sedc_start_i signal, the SED operation starts. The SED operation is run once for each assertion of sedc_start_i signal and when done, this sedc_busy_o goes LOW, as shown in Figure 6.3.

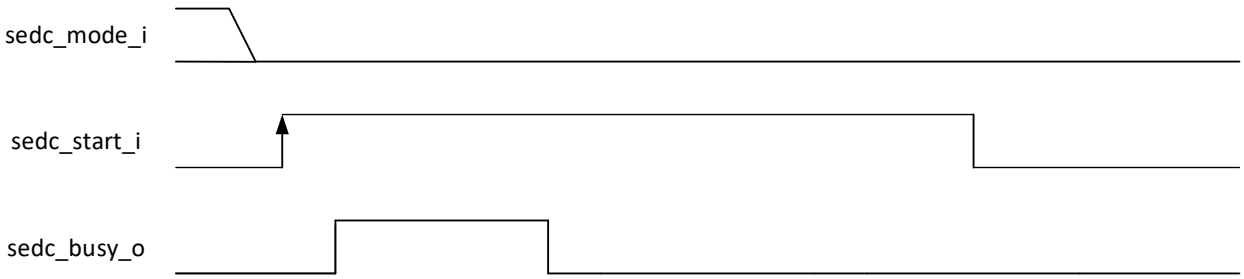

Figure 6.3. SED One-shot Mode

The preferred action to take when an error is detected is to reconfigure the PLD. Reconfiguration can be accomplished by driving the PROGRAMN pin low. This can be done by externally connecting a GPIO pin to PROGRAMN.

6.2. SED Error Handling

The diagram below shows the different types of errors reported by the SED module in Crosslink-NX device. The sedc_errc_o signal flags as soon as there is an error. The sedc_err_o, sedc_errm_o, and sedc_errcrc_o are sticky flags and the SED module had to be restarted to reset these error flags as shown in Figure 6.4.

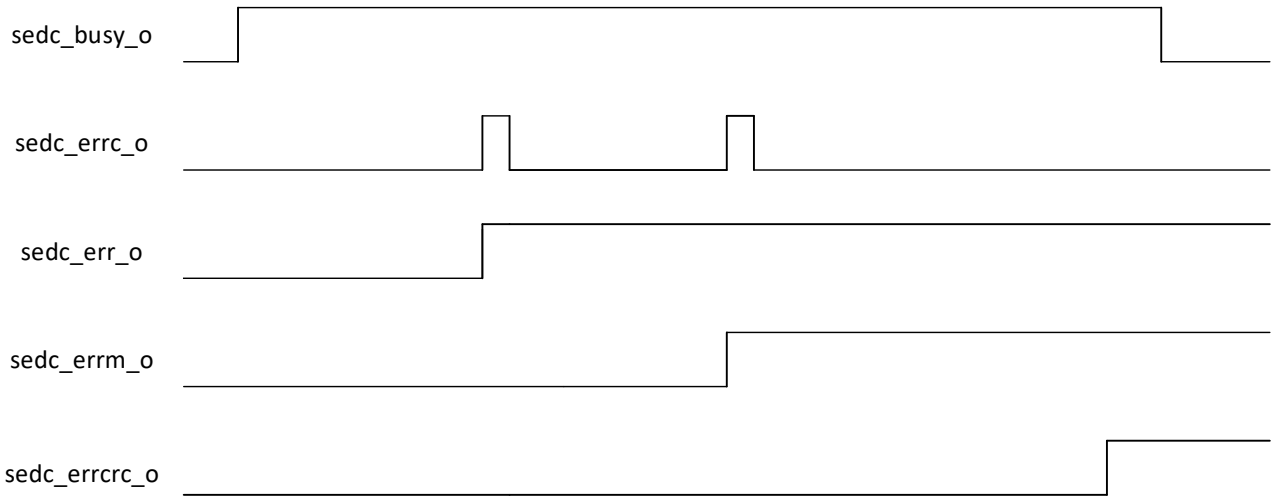

Figure 6.4. SEDC Error Handling Flags

7. SEC Flow

CrossLink-NX device supports real time Soft Error Correction (SEC) feature in which a single bit error can be corrected using ECC at the frame level. Once the SEC is enabled, the SED/SEC module reports the error location, providing details about the error frame and the exact location of the bit error in that frame. Figure 7.1 shows the SEC flow in CrossLink-NX devices.

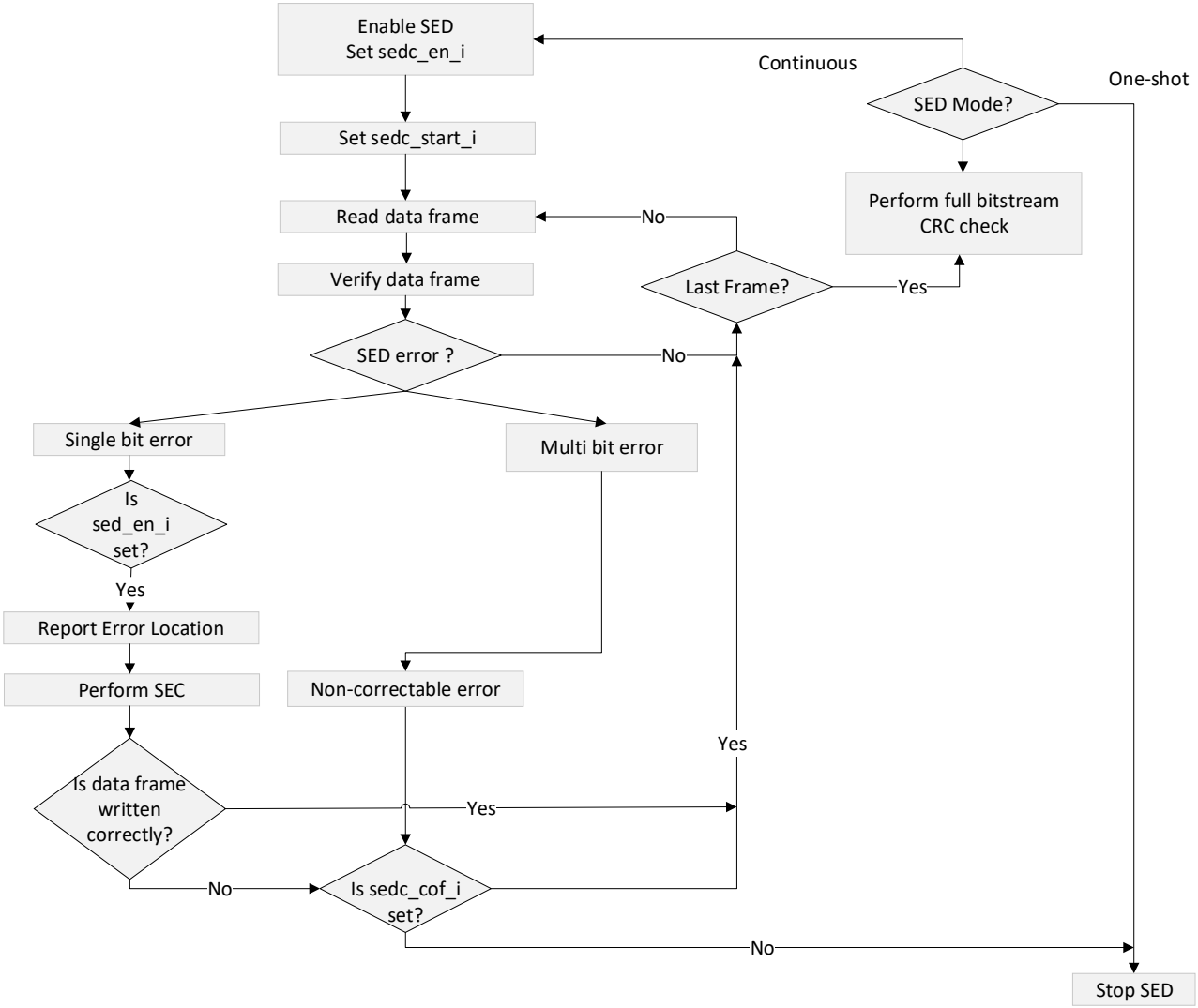

Figure 7.1. SEC Flow

8. SED Run Time

In Crosslink-NX device the amount of time needed to perform a SED check depends on the density of the device, frequency of the SED clock driver signal and the number of shift lanes used to shift data into the device. There will also be some overhead time for calculation, but it is fairly short in comparison. An approximation of the SED run time can be found by using the following formula:

*SED run time (ms) = (Total no of frames * (bytes per frame + overhead)) / SED clock (MHz)*

For example, in LIFCL-40, the run time for SED clock running at 150 MHz is:

Total no. of frame = 9172

Bytes per frame = 85

Overhead bytes = 5

*SED Run time = (9172 * (85 + 5)) / 150 MHz = (9172 * 90) / 150 MHz = 5.5ms*

9. Sample Code

The following simple example code shows how to instantiate the SED primitive. Note that the SED IP can be created using the IP catalog in Lattice Radiant® software version 2.0 or later.

9.1. SED Verilog Example

9.1.1. Verilog Example of SED IP

```
module sed_test (sed_en_i, sedc_cof_i, sedc_en_i, sedc_mode_i, sedc_start_i, osccclk_sedc_i,
sedc_rst_i, sedc_busy_o, sedc_err_o, sedc_errc_o, sedc_errcrc_o, sedc_errm_o,
sedc_frm_errloc_o, sedc_dsr_errloc_o)/* synthesis syn_black_box syn_declare_black_box=1 */;

    input  sed_en_i;
    input  sedc_cof_i;
    input  sedc_en_i;
    input  sedc_mode_i;
    input  sedc_start_i;
    input  osccclk_sedc_i;
    input  sedc_rst_i;
    output  sedc_busy_o;
    output  sedc_err_o;
    output  sedc_errc_o;
    output  sedc_errcrc_o;
    output  sedc_errm_o;
    output  [15:0]  sedc_frm_errloc_o;
    output  [12:0]  sedc_dsr_errloc_o;

endmodule
```

9.1.2. Verilog SED IP Instantiation

```
sed_test sed_module_name    (.sed_en_i(sed_enable),  .sedc_cof_i(sedc_cof),
       .sedc_en_i(sedc_en),        .sedc_mode_i(sedc_mode),    .sedc_start_i(sedc_start),
.osccclk_sedc_i(sedc_clk ),  .sedc_rst_i(sedc_rst), .sedc_busy_o(sedc_busy),
.sedc_err_o(sedc_err1),     .sedc_errc_o(sedc_err_current), .sedc_errcrc_o(sedc_errcrc),
       .sedc_errm_o(sedc_errm), .sedc_frm_errloc_o(sedc_frm_loc),
       .sedc_dsr_errloc_o(sedc_bit_err_loc) );
```

9.2. SED VHDL Example

9.2.1. VHDL Component Instantation

```
component sed_test is
    port(
        sed_en_i: in std_logic;
        sedc_cof_i: in std_logic;
        sedc_en_i: in std_logic;
        sedc_mode_i: in std_logic;
        sedc_start_i: in std_logic;
        oscclk_sedc_i: in std_logic;
        sedc_rst_i: in std_logic;
        sedc_busy_o: out std_logic;
        sedc_err_o: out std_logic;
        sedc_errc_o: out std_logic;
        sedc_errcrc_o: out std_logic;
        sedc_errm_o: out std_logic;
        sedc_frm_errloc_o: out std_logic_vector(15 downto 0);
        sedc_dsr_errloc_o: out std_logic_vector(12 downto 0)
    );
end component;
```

9.2.2. VHDL Instantiation

```
SED_instance: sed_test port map(
    sed_en_i=> sed_enable,
    sedc_cof_i=> sed_cof,
    sedc_en_i=> sedc_en,
    sedc_mode_i=> sedc_mode ,
    sedc_start_i=> sedc_start,
    oscclk_sedc_i=> sedc_clk,
    sedc_rst_i=> sedc_rst,
    sedc_busy_o=> sedc_busy,
    sedc_err_o=> sedc_err1,
    sedc_errc_o=> sedc_err_current,
    sedc_errcrc_o=> sedc_errcrc,
    sedc_errm_o=> sedc_errm,
    sedc_frm_errloc_o=> sedc_frm_loc,
    sedc_dsr_errloc_o=> sedc_bit_err_loc );
```

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Revision 1.0, February 2020

Section	Change Summary
All	Initial release.

www.latticesemi.com