Lattice Avant SED/SEC User Guide

Preliminary Technical Note

FPGA-TN-02290-0.80

December 2023

Disclaimers

Inclusive Language

This document was created consistent with Lattice Semiconductor's inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice's inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CRC	Cyclic Redundancy Check
CRC32	Cyclic Redundancy Check – 32 bits
CRAM	Configuration Random Access Memory
EBR	Embedded Block RAM
ECC	Error Correction Code
IP	Intellectual Property
LMMI	Lattice Memory Mapped Interface
PLD	Programmable Logic Device
SEC	Soft Error Correction
SED	Soft Error Detection
SEDC	Soft Error Detection/Correction
SEI	Soft Error Injection
SEU	Single Event Upset
SoC	System-on-a-Chip
SRAM	Static Random Access Memory

1. Introduction

This document describes the hard-logic soft error detection (SED) and soft error correction (SEC) implemented in the Lattice Avant™-AT-G and Avant-AT-X device families. When a soft error is detected, Avant-AT-G/X devices provide an easy way to optionally perform SEC without affecting the functionality of the device.

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 Configuration Random Access Memory (CRAM), requiring error detection and correction for large memory systems in high-reliability applications. As device geometries continue to shrink, the probability of memory errors in Static Random Access Memory (SRAM) becomes significant for some systems.

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. Most traditional approaches that are taken to address this issue involve soft intellectual property (IP) cores that you instantiate in your design. Such approaches utilize valuable resources, possibly affecting design performance.

Avant-AT-G/X devices have an improved hardware-implemented SED circuit that can be used to detect and correct SRAM errors. There are two layers of SED/SEC, error correction code (ECC) logic to detect and correct single bit error per data frame and detect two-bit errors and cyclic redundancy check (CRC) logic to detect multi-bit errors in the device.

2. Overview of the SEDC IP

The Soft Error Detection/Correction (SEDC) IP offers the following enhanced features:

- Frame-by-frame SED check
- Multiple regions (four regions as hardware default) run in parallel for fast SED and SEC performance
- 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
- Force error capability for system-level simulation

The SED module is part of the Configuration block in Avant-AT-G/X devices. The configuration data is divided into frames in multiple regions so that the FPGA can be programmed as a whole or in precise parts. The SED hardware reads serial data from the FPGA's configuration memory frame-by-frame in the background while the device is in user mode and performs ECC calculation on every frame of configuration data (see Figure 2.1). Once a single bit error is detected, a single event upset (SEU) notification is generated and SED resumes operation. When SEC is enabled, single-bit errors are corrected; the corrected value is rewritten to the 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, CRC is calculated for the entire bitstream along with ECC. After the ECC is calculated on all frames of configuration data, CRC is calculated for the entire configuration data (bitstream). Due to the dynamic contents of memories, the CRC and ECC calculations do not include Embedded Block RAM (EBR). Dynamic RAM must not be used with SED. Otherwise, the SED reports failures when normal RAM content changes occur.

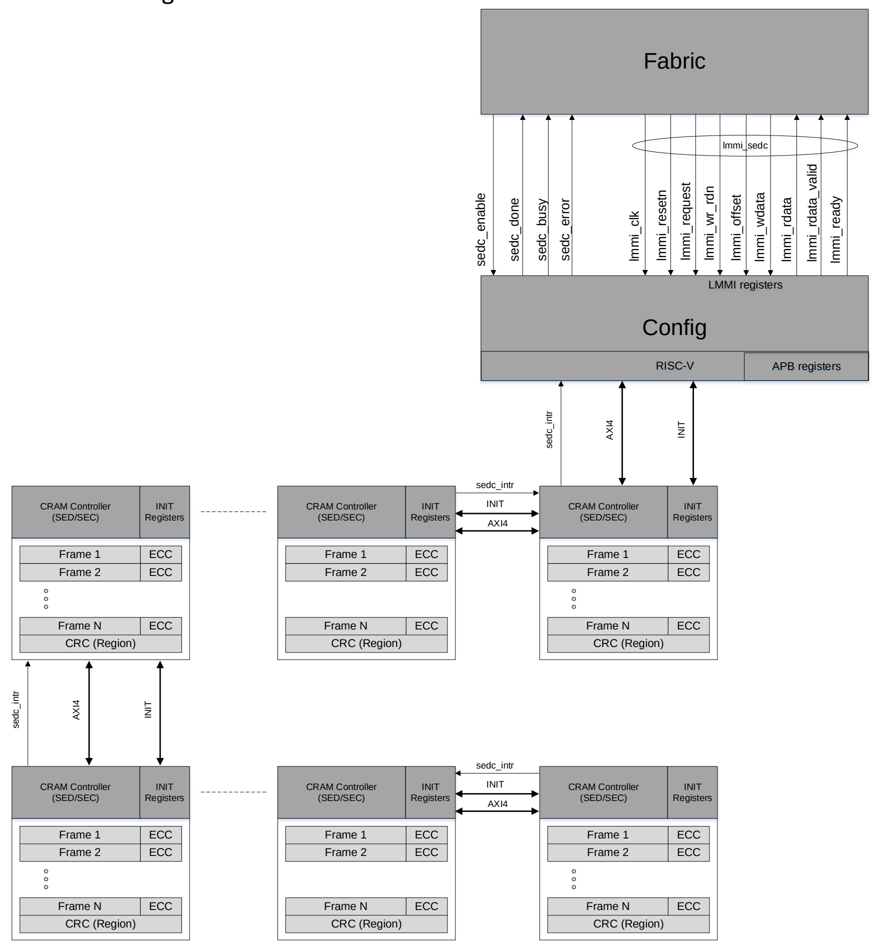

Figure 2.1. Bitstream Data Structure

The SEDC IP is part of the sysCONFIG block of devices built on the Avant-AT-G/X device family. Figure 2.2 shows the system-level view of the SEDC IP.

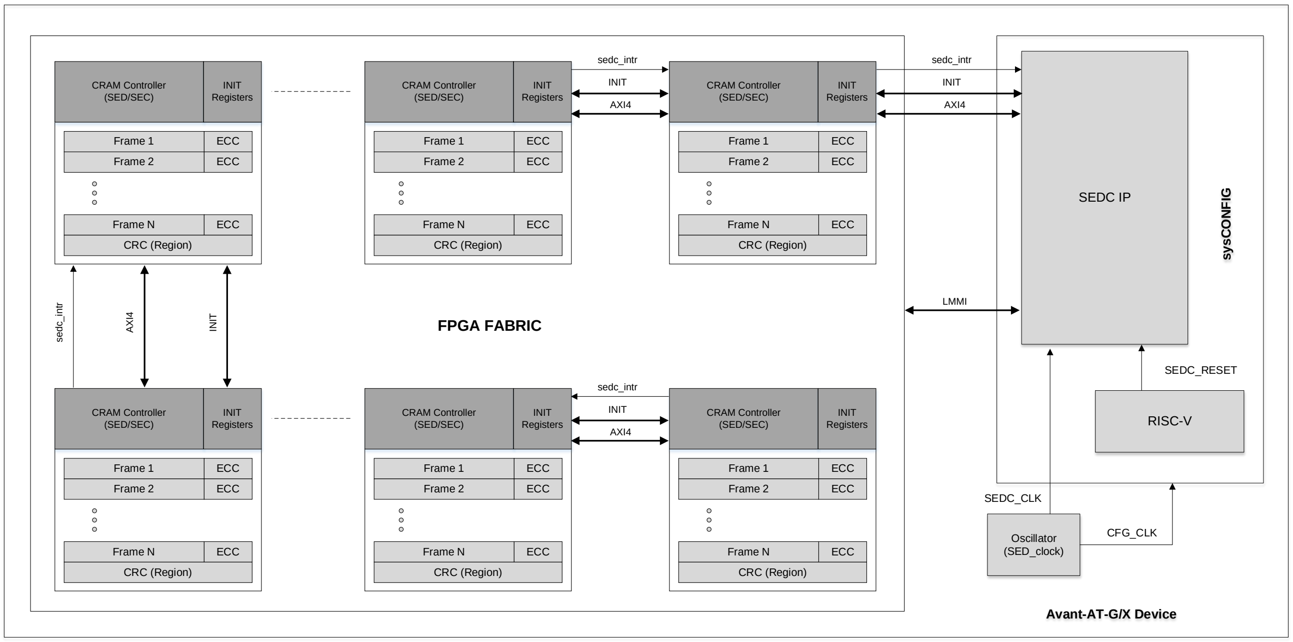

Figure 2.2. SEDC System Block Diagram

2.1. SEDC IP Signal Description

For more information on Avant SEDC IP signals, refer to the *Signal Description* section of the *Avant-G/X SEDC Module IP User Guide (FPGA-IPUG-02232)*.

2.2. SEDC IP Register Description

For more information on Avant SEDC IP register map description, refer to the *Register Description* section of the *Avant-G/X SEDC Module IP User Guide (FPGA-IPUG-02232)*.

3. SEDC IP Setup

In a typical user application, the SEDC IP must be instantiated along with the LMMI host logic, which could access the SEDC IP control registers and status.

3.1. SEDC IP Application

Figure 3.1 shows a typical SEDC IP application. Through the Lattice Memory Mapped Interface (LMMI), you can select the SEDC mode, obtain the SEDC running status and error location in case of a correctable SED error.

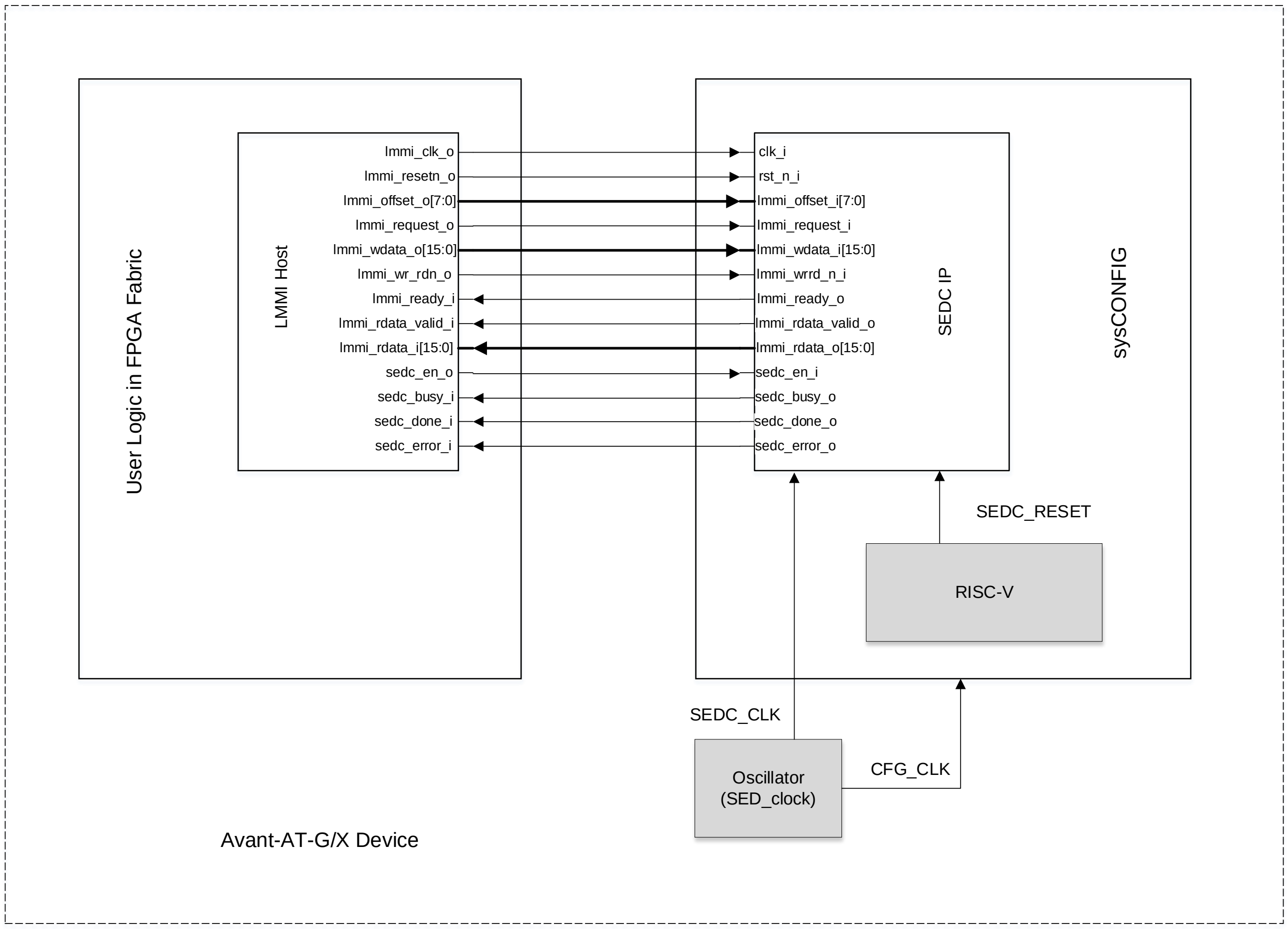

Figure 3.1. SEDC IP Application Block Diagram

3.2. SEDC IP Clock and Reset

The SED circuitry is driven by the FPGA's internal oscillator. You must instantiate the oscillator IP (SEDCLK option enabled) along with the SEDC IP from the Lattice Radiant™ software IP catalog, and route the clock and reset signals between them, as shown in Figure 3.2.

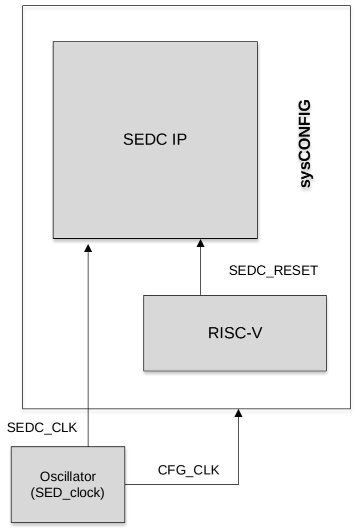

Figure 3.2. SEDC IP Clock and Reset

The default oscillator frequency is 400 MHz. You can choose to lower the oscillator frequency by configuring the oscillator IP using the Lattice Radiant software IP catalog. You can set the SEDCLK_divider setting anywhere between 1 to 256 in integer increments resulting in frequency range from 400 MHz to 1.56 MHz (SED oscillator frequency = 400 MHz / SEDCLK_divider).

Table 3.1. SEDC Internal Oscillator Divider Settings

Divider Setting	SEDCLK_Divider	SEDC Clock Frequency (MHz)
Divide by 1	1	400
Divide by 2	2	200
Divide by 3	3	133.33
—	—	—
—	—	—
—	—	—
Divide by 256	256	1.56

The reset of the SEDC hardware primitive is directly controlled by the RISC-V microcontroller inside the sysCONFIG engine. There is no user control to the SEDC engine reset.

4. SEDC Flow

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

Avant-AT-G/X devices have an advanced SEDC flow with two levels of SEDC 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 the frames of the device bitstream are read, the SEDC module performs a CRC of the entire bitstream (second-level SED) to check for bitstream integrity giving the device improved SED performance. Figure 4.1 shows the SEDC flow in Avant-AT-G/X devices.

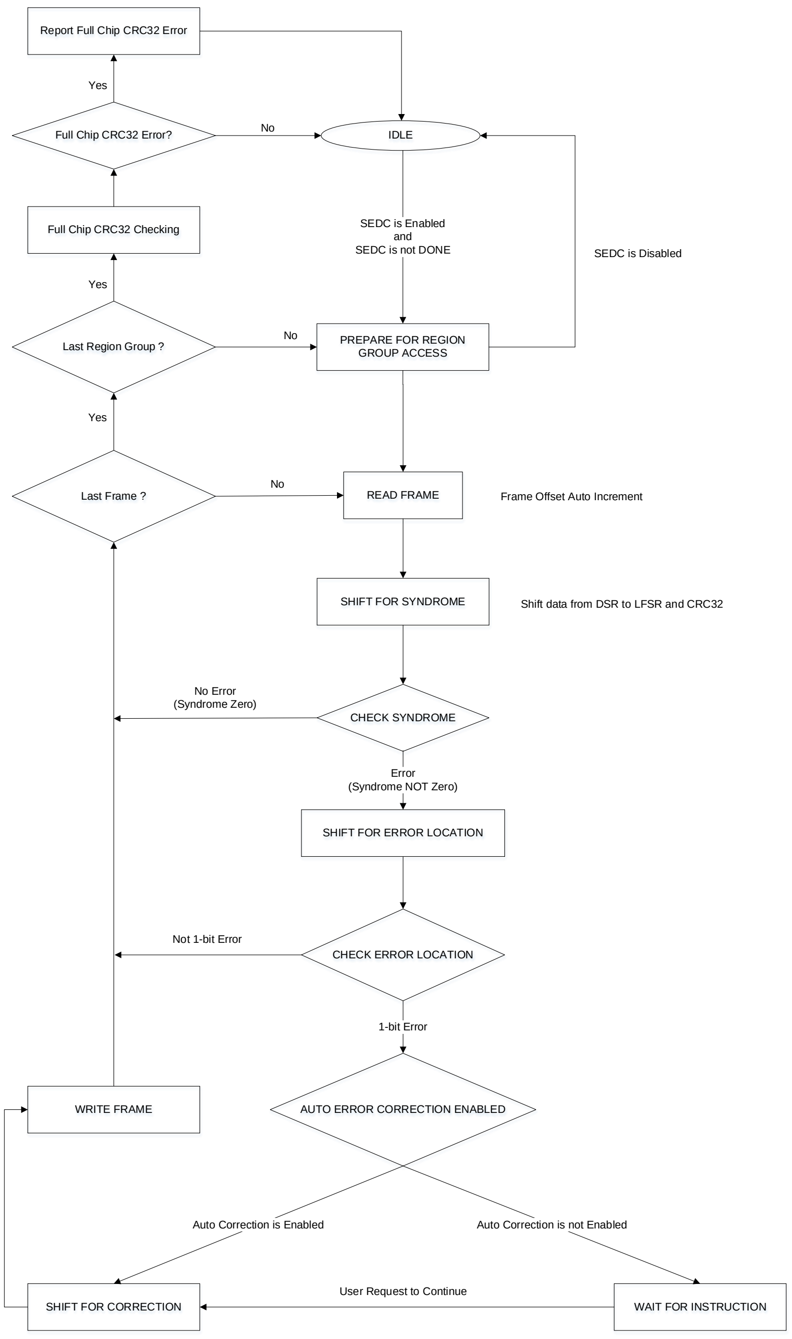

Figure 4.1. SEDC Flow

4.1. SEDC Modes

Avant-AT-G/X devices support four different SEDC modes. These modes give you the flexibility to run the SED. The SEDC mode can be set through the LMMI bus. Table 4.1 lists the available SEDC modes.

Table 4.1. SEDC Modes

reg_sedc_single_scan_en	reg_sedc_auto_err_corret	Mode	Mode Description	Note
0	0	Continuous Mode	Continuous Mode with Auto 1-bit Error Correction	Hardware will keep scanning the frames. When there is an SEDC error detected, the hardware will halt and trigger the RISC-V microcontroller to notify users about the errors.
	1		Continuous Mode without Auto 1-bit Error Correction	Hardware will keep scanning the frames even when there is an error detected. If the error is a 1-bit error, the hardware will correct it automatically and resume scanning the frames without re-checking immediately after the correction.
1	0	One-Shot Mode	One-shot Mode with Auto 1-bit Error Correction	Hardware will perform a one-time scan of all frames. When there is an SEDC error detected, the hardware will halt and trigger the RISC-V microcontroller to notify users about the errors.
	1		One-shot Mode without Auto 1-bit Error Correction	Hardware will perform a one-time scan of all frames. The hardware will keep scanning the frames even when there is an error detected. If the error is a 1-bit error, the hardware will correct it automatically and resume scanning the frames without re-checking immediately after the correction.

4.1.1. Continuous Mode

In Continuous Mode with reg_sedc_single_scan_en set to 0, the SED runs continuously as long as the *sedc_enable* signal is high. The following lists the process flow for Continuous Mode:

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

When *sedc_mode_i* is set to 1 and *sedc_start_i* is always high, the SED operation runs continuously, as shown in Figure 4.2.

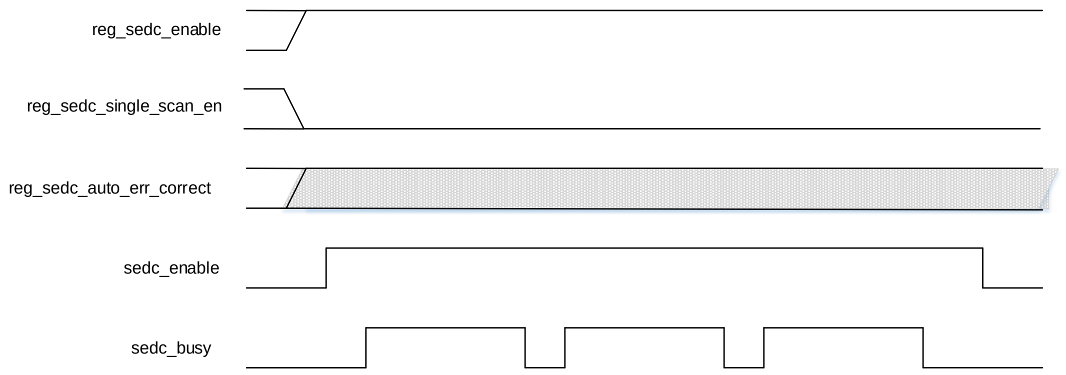

Figure 4.2. SED Continuous Mode

4.1.2. One-Shot Mode

In this mode with reg_sedc_single_scan_en set to 1, the SED runs once for each assertion of the sedc_enable signal. The following lists the process flow for the One-Shot-Mode:

1. For One-Shot Mode, the *sedc_enable* 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 the configuration SRAM. The *sedc_busy* signal is high as long as the SED is running.
3. The SED finishes checking. The SED error flags are updated and the *sedc_busy* flag goes low. Another SED cycle is started by making a low to high transition on the *sedc_enable* signal.

   Note: If there is an error, disable the *sedc_enable* signal to reset all error flags.

As soon as there is a low to high transition on the *sedc_enable* signal, the SED operation starts. The SED operation is run once for each assertion of the *sedc_enable* signal and when done, the *sedc_busy* signal goes low, as shown in Figure 4.3.

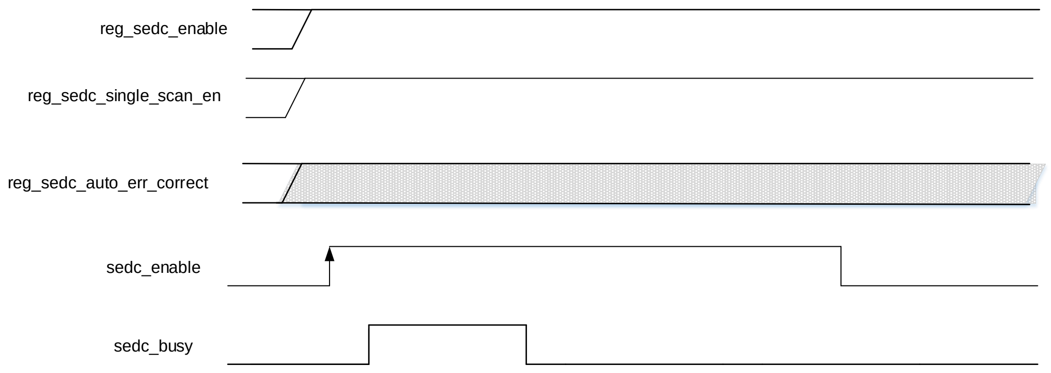

Figure 4.3. SED One-Shot Mode

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.

5. SEC Flow

Avant-AT-G/X devices are built to support real time SEC feature in which a single-bit error can be corrected using ECC at the frame level. When the SEC is enabled, the SEDC IP reports the error location through the LMMI bus with details about the error frame and the exact location of a single-bit error in that frame. Figure 5.1 shows the SEC flow in Avant-AT-G/X devices.

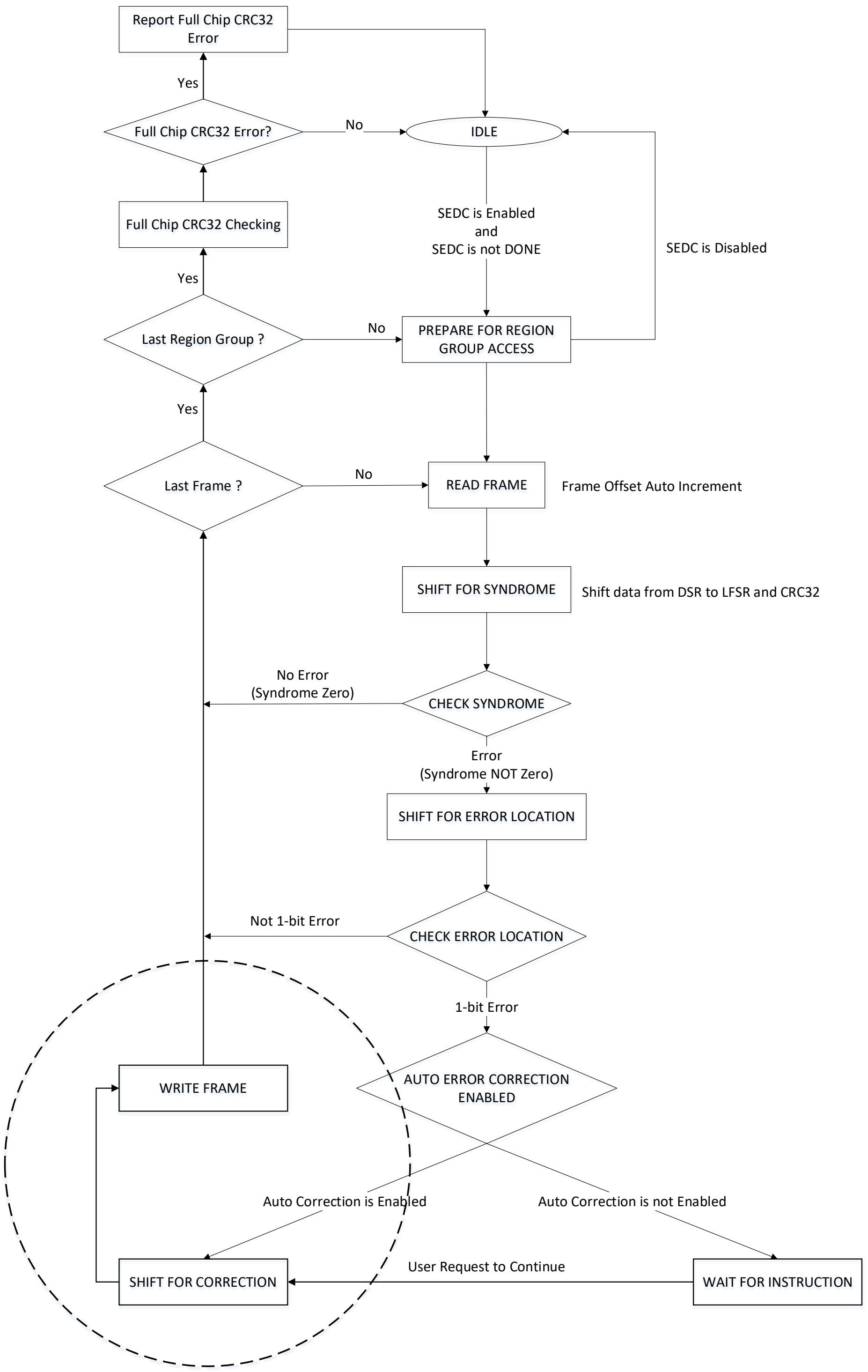

Figure 5.1. SEC Flow

6. Code Examples

The following sections show the Verilog and VHDL code examples on how to instantiate the SEDC IP.

6.1. SEDC IP Verilog Example

6.1.1. Verilog Module Example

```
module sedc_test (clk_i, rst_n_i, sedc_enable, sedc_done, sedc_busy, sedc_sedc_error,
lmmi_request_i, lmmi_wrrd_n_i, lmmi_offset_i, lmmi_wdata_i, lmmi_ready_o, lmmi_rdata_o,
lmmi_rdata_valid_o, lmmi_rdata_o)/* synthesis syn_black_box syn_declare_black_box=1 */;

    input  clk_i;
    input  rst_n_i;
    input  sedc_enable;
    input  sedc_done;
    input  sedc_busy;
    input  sedc_error;
    input  sedc_rst_i;
    output lmmi_request_i;
    output mmi_wrrd_n_i;
    output [7:0] lmmi_offset_i;
    output [15:0] lmmi_wdata_i;
    output lmmi_ready_o;
    output lmmi_rdata_valid_o;
    output [15:0] lmmi_rdata_o;

endmodule
```

6.1.2. Verilog Instantiation

```
sedc_test sedc_module_name (.clk_i(clk_i),
                     .rst_n_i(rst_n_i),
                     .sedc_enable(sedc_enable),
                     .sedc_done(sedc_done),
      .sedc_busy(sedc_busy),
      .sedc_error(sedc_error),
      .lmmi_request_i(lmmi_request_i),
      .lmmi_wrrd_i(lmmi_wrrd_i ),
      .lmmi_offset_i(lmmi_offset_i),
      .lmmi_wdata_i(lmmi_wdata_i),
      .lmmi_ready_o(lmmi_ready_i),
      .lmmi_rdata_valid_o(lmmi_rdata_valid_o),
      .lmmi_rdata_o(lmmi_rdata_o) );
```

6.2. SEDC IP VHDL Example

6.2.1. VHDL Component Declaration

```
component sedc_test is
    port(
        clk_i: in std_logic;
        rst_n_i: in std_logic;
        sedc_enable: in std_logic;
        sedc_done: out std_logic;
        sedc_busy: out std_logic;
        sedc_error: out std_logic;
        lmmi_request_i: in std_logic;
        lmmi_wrrd_n_i: in std_logic;
        lmmi_offset_i: in std_logic_vector(7 downto 0);
        lmmi_wdata_i: in std_logic_vector (15 downto 0);
        lmmi_ready_o: out std_logic;
        lmmi_rdata_valid_o: out std_logic;
        lmmi_rdata_o: out std_logic_vector(15 downto 0)
    );
end component;
```

6.2.2. VHDL Instantiation

```
SEDC_instance: sed_test port map(
    clk_i=> clk_i,
    rst_n_i=> rst_n_i,
    sedc_enable=> sedc_enable,
    sedc_done=> sedc_done,
    sedc_busy=> sedc_busy,
    sedc_error=> sedc_error,
    lmmi_request_i=> lmmi_request_i,
    lmmi_wrrd_n_i=> lmmi_wrrd_n_i,
    lmmi_offset_i=> lmmi_offset_i,
    lmmi_wdata_i=> lmmi_wdata_i,
    lmmi_ready_o=> lmmi_ready_o,
    lmmi_rdata_valid_o=> lmmi_rdata_valid_o,
    lmmi_rdata_o=> lmmi_rdata_o);
```

References

For more information, refer to the following resources:

- Avant-G/X SEDC Module IP User Guide (FPGA-IPUG-02232)
- Lattice Radiant Software User Guide
- Lattice Radiant Software web page
- Avant-G web page
- Avant-X web page
- Lattice Insights web page for training courses and learning plans

Technical Support Assistance

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

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 0.8, December 2023

Section	Change Summary
All	Preliminary release.