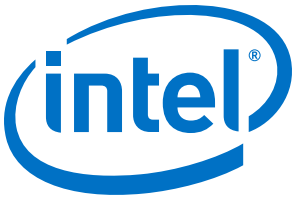


Advanced SEU Detection Intel® FPGA IP User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.1**

ALTADVSEU | 2019.03.26
Latest document on the web: **PDF** | **HTML**

Subscribe
Send Feedback

Contents

1. Advanced SEU Detection Intel® FPGA IP Overview

The Advanced SEU Detection IP core enables you to perform:

- Hierarchy tagging—Allows you to describe the criticality of each portion of your design's hierarchy relative to single event upset (SEU). You perform hierarchy tagging during the design phase.
- Sensitivity processing—Determines the criticality of an SEU detected and located by error detection cyclical redundancy check (EDCRC) hard IP. This feature includes on- and off-chip sensitivity processing. The system performs sensitivity processing at runtime.

Table 1. Features Device Family Support

Feature	Supported Devices
Hierarchy Tagging and Sensitivity Processing	Intel® Arria® 10, Intel Cyclone® 10 GX, Stratix® V, Arria V, and Cyclone V.
Sensitivity Processing	Stratix IV, Arria II GX, and Arria II GZ.

You can select and configure the Advanced SEU Detection IP core through the IP Catalog and parameter editor in the Intel Quartus® Prime software.

The Advanced SEU Detection IP core must be used along with the EMR Unloader Intel FPGA IP core. The EMR Unloader IP core provides Error Message Register (EMR) contents whenever it detects an EDCRC error. Connect the `emr`, `emr_valid` and `emr_error` signals from your EMR Unloader IP variation to the corresponding inputs of your Advanced SEU Detection variation.

Related Information

- Error Message Register Unloader Intel FPGA IP Core User Guide
- Introduction to Intel FPGA IP Cores
  Provides general information about all Intel FPGA IP cores, including parameterizing, generating, upgrading, and simulating IP cores.
- Creating Version-Independent IP and Qsys Simulation Scripts
  Create simulation scripts that do not require manual updates for software or IP version upgrades.
- Project Management Best Practices
  Guidelines for efficient management and portability of your project and IP files.
- Advanced SEU Detection Intel FPGA IP User Guide Archives on page 29
  Provides a list of user guides for previous versions of the Advanced SEU Detection Intel FPGA IP.

2. Advanced SEU Detection Intel FPGA IP Functional Description

The following Intel FPGA devices contain a cyclic redundancy check (CRC) value per CRAM frame. The EDCRC logic can also determine the location and type of upset.

- Intel Arria 10, Intel Cyclone 10 GX, Stratix V, Arria V, and Cyclone V device families contain a 32 bit CRC value
- Stratix IV and Arria II devices contain a 16 bit CRC value

The Intel Quartus Prime software can generate a Sensitivity Map Header File (`.smh`) of the configuration regions of your design that are sensitive to SEU. The software uses the design hierarchy and its assigned advanced SEU detection (ASD) region to to create the `.smh`. During sensitivity processing, the Advanced SEU Detection IP core uses the location information contained in the device EMR to look up the upset location in the `.smh`. It returns whether or not the bit is critical for the design.

You can instantiate the Advanced SEU Detection IP core with the following configurations:

- On-Chip Lookup Sensitivity Processing—The sensitivity processing soft IP provides error location reporting and lookup.
- Off-Chip Lookup Sensitivity Processing—An external unit (such as a microprocessor) performs error location lookup using the EMR information.

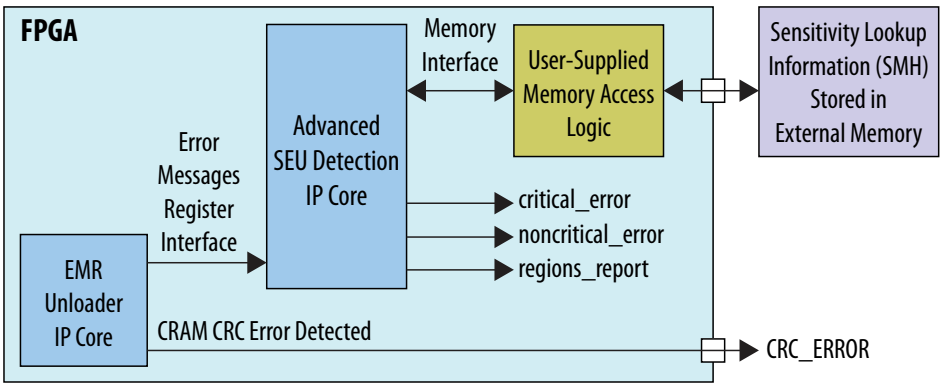
Related Information

SMH File Types on page 13

2.1. On-Chip Lookup Sensitivity Processing

All device families that support SEU detection include a hardened error detection block. This block detects soft errors and provides the location of single-bit errors and double-bit adjacent errors for supported devices. The Advanced SEU Detection IP core reads the error detection register of the error detection block, and then compares single-bit error locations with a sensitivity map. This check determines whether or not the failure affects the device operation.

Figure 1. System Overview for On-Chip Lookup Sensitivity Processing



The Advanced SEU Detection IP core accesses the EMR content (provided by the EMR Unloader IP core or user logic), analyzes the EMR content, and issues a query to an external memory containing the sensitivity map. The system designer must provide the information for the memory access logic and external memory.

To mitigate SEU in the error detection logic, implement an SEU detection circuit that tolerates a soft error in its logic. For example, instantiate two instances of the Advanced SEU Detection IP core in your design and then compare the output of the instances. Each instance of the IP core highlights errors that occur in the other instance as "critical."

Related Information

- Error Message Register Unloader Intel FPGA IP Core User Guide
- Configuration, Design Security, and Remote System Upgrades in Stratix V Devices
  Provides more information about the design security for Stratix V devices.
- Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices
  Provides more information about the design security for Stratix IV devices.
- Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices
  Provides more information about the design security for Arria 10 devices.
- Configuration, Design Security, and Remote System Upgrades in Arria V Devices
  Provides more information about the design security for Arria V devices.
- Configuration, Design Security, and Remote System Upgrades in Cyclone V Devices
  Provides more information about the design security for Cyclone V devices.

2.1.1. On-Chip Sensitivity Processor

When you implement an on-chip sensitivity processor, the Advanced SEU Detection IP core interacts with user-supplied external memory access logic to read the `.smh` stored in external memory. Once it determines the sensitivity of the affected CRAM bit, the IP core can assert a critical error signal so that the system provides an appropriate response. If the SEU is not critical, the critical error signal may be left un-asserted.

On-chip sensitivity processing is autonomous: the FPGA determines whether an SEU affected it without using external logic. On-chip sensitivity processing requires some FPGA logic resources for the external memory interface.

2.1.2. On-Chip Processing Signals

Figure 2. Advanced SEU Detection IP Core Signals for On-Chip Processing

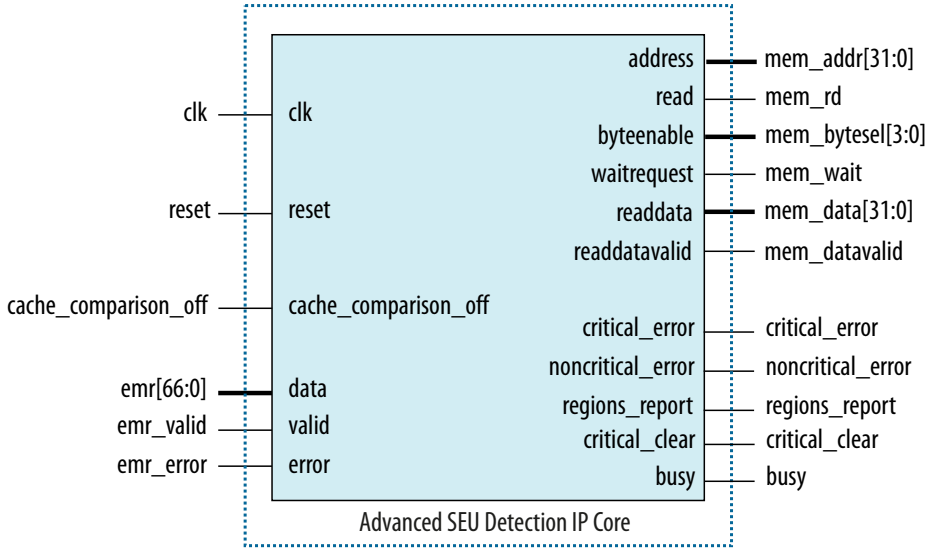


Table 2. Advanced SEU Detection IP Core Signals for On-Chip Processing

Interface	Signals	Type	Width	Description
Clock and Reset	`clk`	Input	1	• Clock input. • Use the same input clock as the EMR Unloader IP core. The input frequency must be sufficient to process the EMR content before the next content may become available. For example, a minimum recommended frequency for Stratix V devices is 30 MHz. If the frequency is too low, the IP core asserts the `critical_error` signal if new EMR content becomes available while the IP core is processing the current content.
	`reset`	Input	1	Active-high reset.
Cache Configuration	`cache_comparison_off`	Input	1	• Static input signal. • Commands the IP core to bypass cache comparison. The EMR value is stored even if it already exists in the cache. • You can use this signal with the internal scrubbing feature for custom design.

continued...

Interface	Signals	Type	Width	Description
Avalon Streaming (Avalon-ST) Sink Interface Signals[1]	emr	Input	• 46 (Stratix IV) • 67 (Cyclone V, Arria V, and Stratix V) • 119 (Intel Arria 10 and Intel Cyclone 10 GX) [2]	Error Message Register (EMR) data input from the EMR Unloader IP core.
	emr_valid	Input	1	Indicates when emr data input is valid.
	emr_error	Input	1	• Indicates when emr data is ignored due to an error. • This error may occur when there is a data overrun from the Intel Unloader IP core.
Errors	noncritical_error	Output	1	Indicates that an SMH lookup determined that the EDCRC error is in a non-critical region.
	critical_error	Output	1	Indicates that an SMH lookup determined that the EDCRC error is in a critical region.
	regions_report	Output	1	• The advanced SEU detection (ASD) region for the error, as reported by the SMH lookup. • The **Largest ASD region ID used** parameter sets this port's width.
	critical_clear	Input	1	• Optional input signal. • Assert this signal to clear error report for the last processed EMR data input. • Clears critical_error and regions_report, or noncritical_error.
	busy	Output	1	• Optional output signal. • Logic high indicates that the ASD IP is busy processing EMR data input. • Signal goes low when processing completes and either the critical_error or noncritical_error signal is asserted.
External Memory Avalon Memory Mapped (Avalon-MM) Master	mem_addr	Output		• Output to the user logic. • Byte address of the 32 bit word to be read.
	mem_rd	Output		• Output to the user logic. • Signals to the user logic to request a read operation.

continued...

(1) Connect the Avalon-ST streaming sink interface to the corresponding Avalon-ST source interface of the EMR Unloader IP core.

(2) The actual EMR data is 78 bits only, [77:0]. Bits [118:78] are reserved.

Interface	Signals	Type	Width	Description
	mem_bytesel	Output		• Output to the user logic. • A 4 bit signal that selects the bytes needed by the IP core. This signal allows 16 bit or 8 bit memories to optimize the number of reads in cases where the IP does not need all 32 bits. If bit 0 of mem_bytesel is 0, the IP core ignores bits 0 to 7 of mem_data. Similarly, if bit 0 of mem_bytesel is 0, the IP core ignores bits 1 to 3.
	mem_wait	Input		• Input from the user logic. • Signals to the memory interface that the read operation is still running. Must be high by the first rising clock after mem_rd is asserted to hold the IP core in a wait state.
	mem_data	Input		• Input from the user logic. • 32 bit data bus. Data must be present if mem_wait goes high and if mem_rd returns low.
	mem_datavalid	Input		• Input from the user logic. • Signals that the mem_data signal contains valid data in response to a previous mem_rd request.

Related Information

Error Message Register Unloader Intel FPGA IP Core User Guide

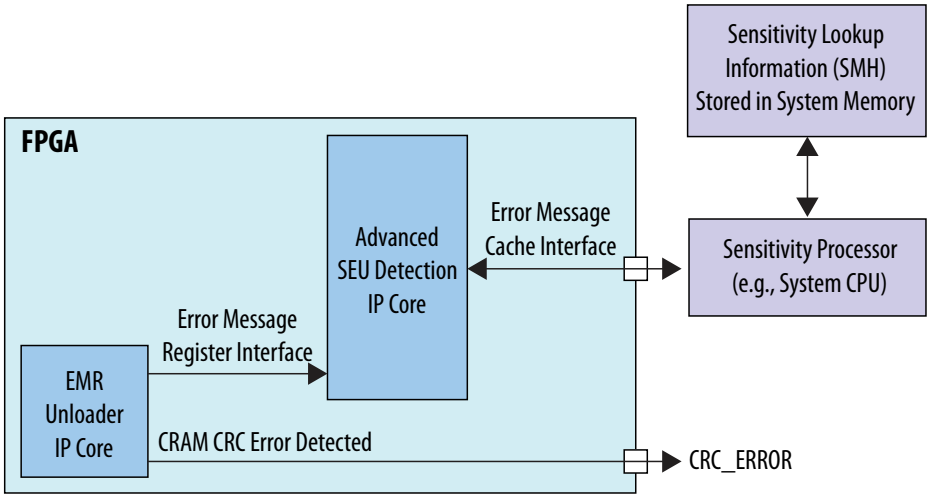
2.2. Off-Chip Lookup Sensitivity Processing

The Advanced SEU Detection IP core analyzes the content of the error detection block's EMR and presents information to a system processor. The processor determines whether the failure affects the device operation. The system processor implements the algorithm to perform a lookup against the .smh.

The off-chip lookup sensitivity processing consists of two components:

- Design logic to interpret content of the EMR of the CRC block and present the information to a processor interface.
- Cache to store off-loaded content of the EMR.

Figure 3. System Overview for Off-Chip Lookup Sensitivity Processing



The EMR processing unit analyzes the content of EMR offloaded from the CRC block by the EMR Unloader IP core upon an SEU. The EMR processing unit writes each unique EMR value into cache, until the cache is full. When the cache is full, it asserts a cache overflow flag to the system interface.

For each new value written into cache, the EMR processing unit asserts an interrupt to the processor. The system processor reads the EMR value and performs a lookup against the `.smh` to determine the criticality of a CRAM location. After the system processor services the interrupt, the EMR processing unit advances the cache line and generates additional interrupt assertions, provided that there is an EMR value in cache that has not been processed.

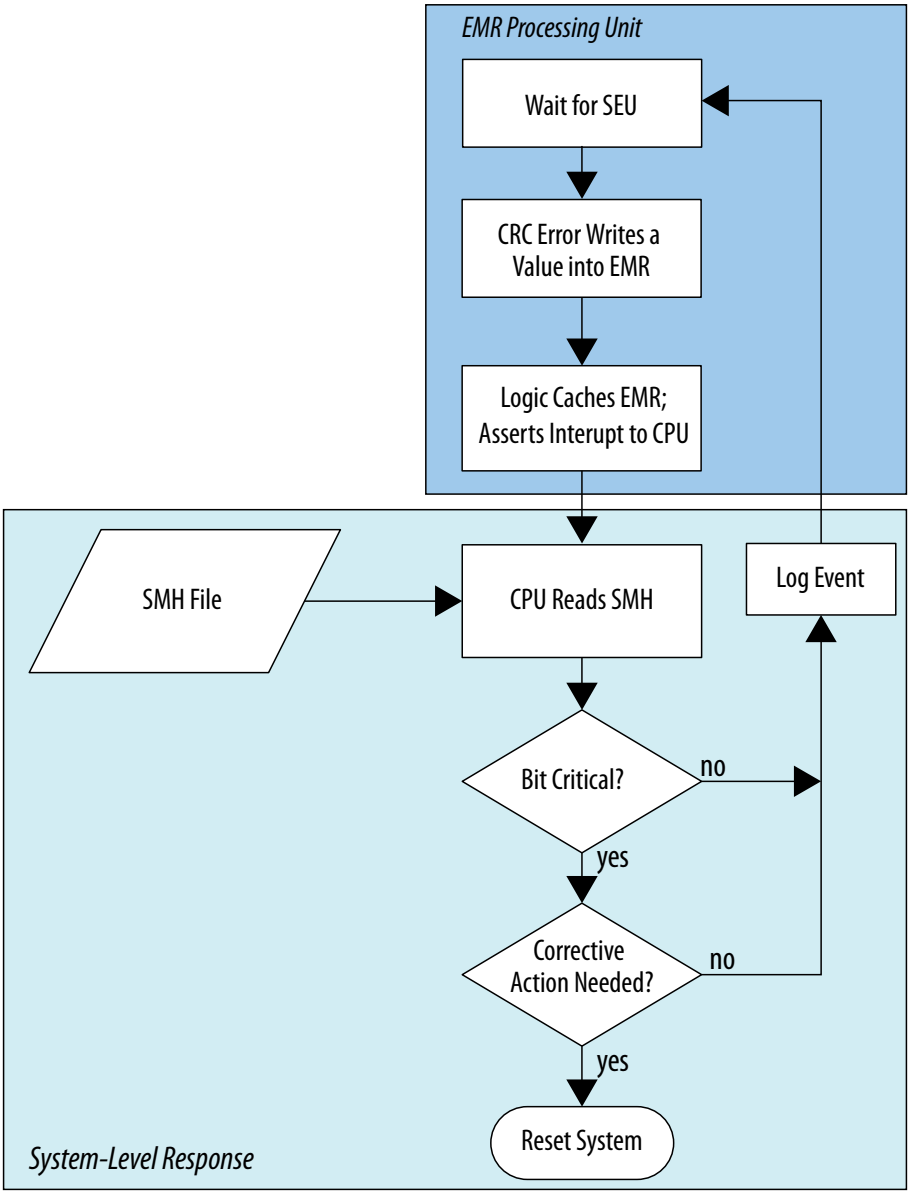
After SMH lookup, the system processor determines the required corrective response.

Related Information

- Error Message Register Unloader Intel FPGA IP Core User Guide
- Configuration, Design Security, and Remote System Upgrades in Stratix V Devices
  Provides more information about the design security for Stratix V devices.
- Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices
  Provides more information about the design security for Stratix IV devices.
- Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices
  Provides more information about the design security for Arria 10 devices.
- Configuration, Design Security, and Remote System Upgrades in Arria V Devices
  Provides more information about the design security for Arria V devices.
- Configuration, Design Security, and Remote System Upgrades in Cyclone V Devices
  Provides more information about the design security for Cyclone V devices.

2.2.1. Off-Chip Lookup Sensitivity Processing Operation Flow

Figure 4. Off-Chip Lookup Sensitivity Processing Operation Flow



Related Information

SMH Lookup on page 13

2.2.2. External Sensitivity Processor

When you implement an external sensitivity processor, a CPU (such as the ARM processor in Intel SoC devices) receives an interrupt request when the FPGA detects an SEU. The CPU then reads the FPGA's error message register and looks up the bit sensitivity in the `.smh` stored in the CPU's memory space.

With external sensitivity processing, the FPGA does not need to implement an external memory interface or store the `.smh`. If the system already has a CPU, external sensitivity processing may be more hardware efficient than on-chip processing.

2.2.3. Off-Chip Processing Signals

Off-chip and on-chip sensitivity processing use similar signals, except the off-chip sensitivity processing uses an EMR cache interface instead of an external memory interface.

Figure 5. Advanced SEU Detection IP Core Signals for Off-Chip Processing

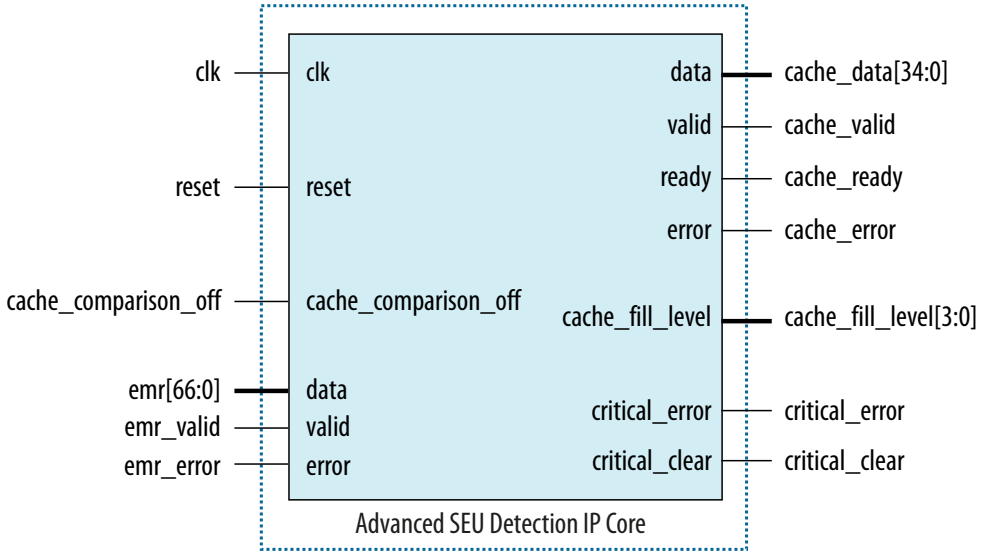


Table 3. Advanced SEU Detection IP Core Signals for Off-Chip Processing

Interface	Signals	Type	Width	Description
Clock and Reset	`clk`	Input	1	• Clock input. • Use the same input clock as the EMR Unloader IP core. The input frequency must be sufficient to process the EMR content before the next content may become available. For example, a minimum recommended frequency for Stratix V devices is 30 MHz. If the frequency is too low, the IP core asserts the `critical_error` signal if new EMR content becomes available while the IP core is processing the current content.
	`reset`	Input	1	Active-high reset.
Cache Configuration	`cache_comparison_off`	Input	1	• Static input signal. • Commands the IP core to bypass cache comparison. • You can use this signal with the internal scrubbing feature for custom design.

continued...

Interface	Signals	Type	Width	Description
Avalon-ST Sink Interface Signals[3]	emr	Input	• 46 (Stratix IV) • 67 (Cyclone V, Arria V, and Stratix V) • 119 (Intel Arria 10 and Intel Cyclone 10 GX)[4]	EMR data input from the EMR Unloader IP core.
	emr_valid	Input	1	Indicates when emr data input is valid.
	emr_error	Input	1	• Indicates when emr data is ignored due to an error. • This error may occur when there is a data overrun from the EMR Unloader IP core.
Errors	critical_error	Output	1	Indicates that a critical EDCRC error is detected. The IP core asserts this signal when any of the following conditions is met: • emr_data indicates a critical EDCRC error. • emr_error is asserted, indicating lost EMR content. • New emr_data becomes available before the previous data is processed, i.e., an emr_data overrun.
	critical_clear	Input	1	• Optional input signal. • Assert this signal to clear the critical_error signal.
Avalon-ST Source Interface Signals	cache_data	Output	• 30 (Stratix IV) • 35 (Cyclone V, Arria V, and Stratix V) • 78 (Intel Arria 10 and Intel Cyclone 10 GX)	• Error cache data. • Provides the location information for an EMR cache entry.
	cache_valid	Output	1	This signal is asserted when the cache contains correctable error data.
	cache_ready	Input	1	Indicates that the Avalon stream interface is ready.
	cache_error	Output	1	This Avalon stream control signal indicates a cache overflow condition. The IP core asserts this signal when new EMR data becomes available for a full cache (cache_fill_level = cache_depth).
Cache Status	cache_fill_level	Output	4	Indicates how many entries are in the cache.

Related Information

Error Message Register Unloader Intel FPGA IP Core User Guide

(3) Connect the Avalon-ST streaming sink interface to the corresponding Avalon-ST source interface of the EMR Unloader IP core.

(4) The actual EMR data is 78 bits only, [77:0]. Bits [118:78] are reserved.

2.2.4. SMH Lookup

The **.smh** file represents a hash of the CRAM bit settings on a design. Related groups of CRAM are mapped to a signal bit in the sensitivity array. During an SEU event, the application can perform a lookup against the `.smh` to determine if a bit is used. By using the information about the location of a bit, you can reduce the effective soft error rate in a running system.

The following criteria determine the criticality of a CRAM location in your design:

- Routing—All bits that control a utilized routing line.
- Adaptive logic modules (ALMs)—If you configure an ALM, the IP core considers all CRAM bits related to that ALM sensitive.
- Logic array block (LAB) control lines—If you use an ALM in a LAB, the IP core considers all bits related to the control signals feeding that LAB sensitive.
- M20K memory blocks and digital signal processing (DSP) blocks—If you use a block, the IP core considers all CRAM bits related to that block sensitive.

Related Information

Off-Chip Lookup Sensitivity Processing Operation Flow on page 10

2.2.4.1. SMH File Types

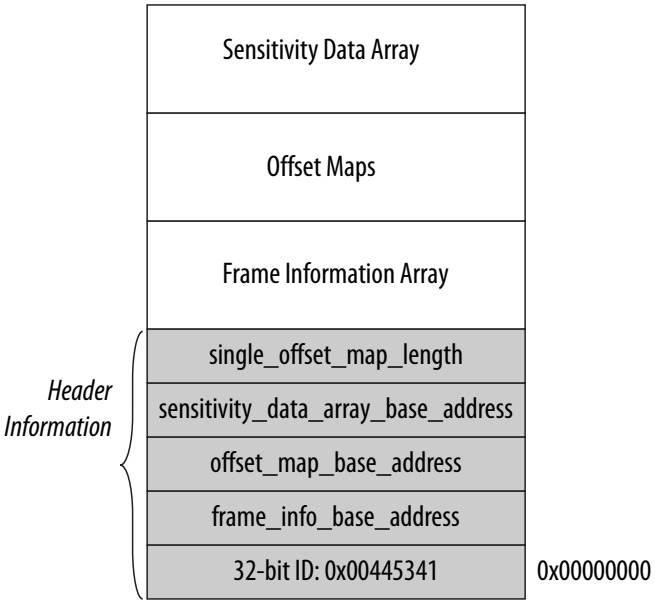
The **.smh** is an Intel-format hexadecimal file. You can generate the following **.smh** file revisions:

- Revision 1—Generated for Stratix IV and Arria II devices. This revision does not support hierarchy tagging, and does not contain tag size or region map information.
- Revision 2—Generated for Stratix V, Arria V, and Cyclone V devices. The generated **.smh** contains tag size and region map information.
- Revision 3—Generated for Intel Arria 10 and Intel Cyclone 10 GX devices. The generated **.smh** contains tag size and region map information, and can accommodate longer sensitivity data addresses.

2.2.4.1.1. Revision 1 SMH

In revision 1 files, the sensitivity map header provides basic information about the `.smh` format. This information includes the base addresses for the frame information, offset maps and length of the single offset map, and the sensitivity data array.

Figure 6. Revision 1 SMH



Revision 1 files contain the following arrays:

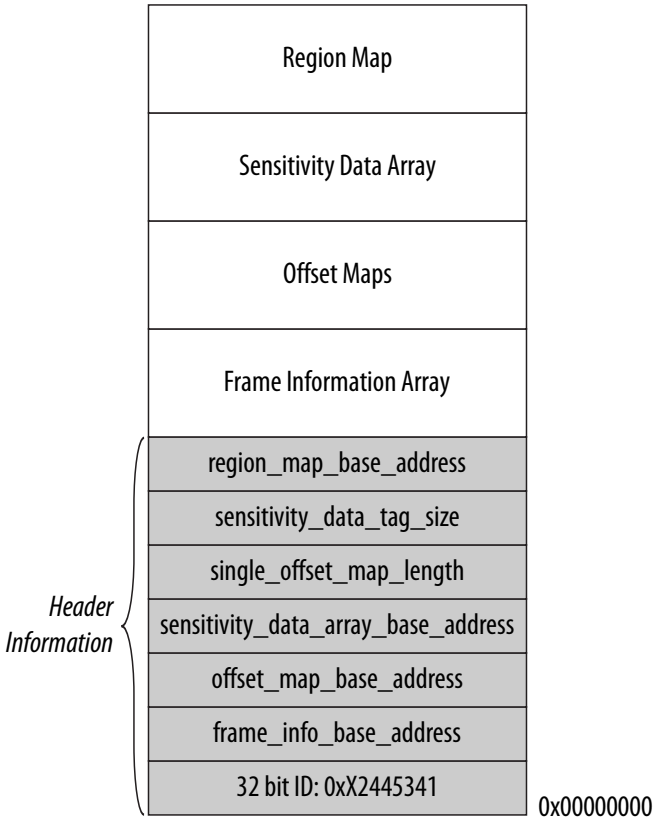
- Frame information array—Contains a 32 bit string for each frame in the device. The frame number serves as the index for the frame information string. Each frame information string provides the following information:
  - `offset_map_array_index` (Bits 7:0)—Index for the offset map array that this frame uses.
  - `frame_info_data_offset` (Bits 31:8)—24 bit address offset into the sensitivity array for this frame.

  Note: For Stratix IV and Arria II devices, the frame information array lists CRAM and embedded RAM frame strings. However, the `.smh` sets the embedded RAM frame string to 0xFFFFFFF in the frame information array entry because the EDCRC circuitry and sensitivity processing only correct CRAM frames. For all other device families, the frame information array lists only the CRAM frame strings.
- Offset map array—The offset map information array is a set of arrays containing 16 bit offset maps. Each offset map value represents an additional offset into the sensitivity array for a frame group. Each offset map value is 16 bits. The `offset_map_length` string in the header information defines the size of each offset map array.
- Sensitivity data array—The sensitivity data array is a flat-bit vector where 1 specifies a sensitive bit and 0 specifies an insensitive bit.

2.2.4.1.2. Revision 2 SMH

In revision 2 files, the sensitivity map header is an extension of revision 1 header format. The header information provides basic information about the `.smh` revision 2, and includes all the revision 1 header information fields. The additional fields include size of the sensitivity data tag size in bits, base addresses for the region map, and 32 bit CRC signature of the corresponding `.sof` file.

Figure 7. Revision 2 SMH



The 32 bit ID of the sensitivity map header revision 2 is defined as follows:

- Bits 23:0—Intel FPGA sensitivity map header ID 0x445341
- Bits 24:27—Bit mask for the header information
  - Bit 24—Reserved
  - Bit 25—Indicates the presence of sensitivity tag information in the **.smh** file
  - Bit 26:27—Reserved
- Bit 28—Indicates the presence of a 32 bit CRC signature of a corresponding **.sof**
- Bits 29:31—Reserved

Revision 2 files contain the following arrays:

- Frame information array—Contains a 32 bit string for each frame in the device. The frame number serves as the index for the frame information string. Each frame information string provides the following information:
  - `offset_map_array_index` (bits 7:0)—Index for the offset map array that this frame uses.
  - `frame_info_data_offset` (bits 31:8)—24-bit address offset into the sensitivity array for this frame.
- Offset map array—The offset map information array is a set of arrays containing 16 bit offset maps. Each offset map value represents an additional offset into the sensitivity array for a frame group. Each offset map value is 16 bits. The size of each offset map array is defined by the `offset_map_length` string contained in the header information.
- Sensitivity data array—The size of the single sensitivity data entry or tag (`sensitivity_data_tag_size`) is in bits and aligned to power of 2. The sensitivity data array is a flat sensitivity tag vector where a sensitive tag of 0 specifies a bit insensitive for all regions, and non-zero tag specifies an offset into region map.
- Region map information array—The region map information array contains a 16-bit string for each non-zero sensitivity tag. The sensitivity data tag serves as the index-1 for the region map array. The string is a bitmask of the regions, the bit is sensitive for. Each region can be identified in the bitmask by mask 1 << (Region ID - 1).

Table 4. Revision 2 SMH File Size and ASD Regions Based on Sensitivity Tag

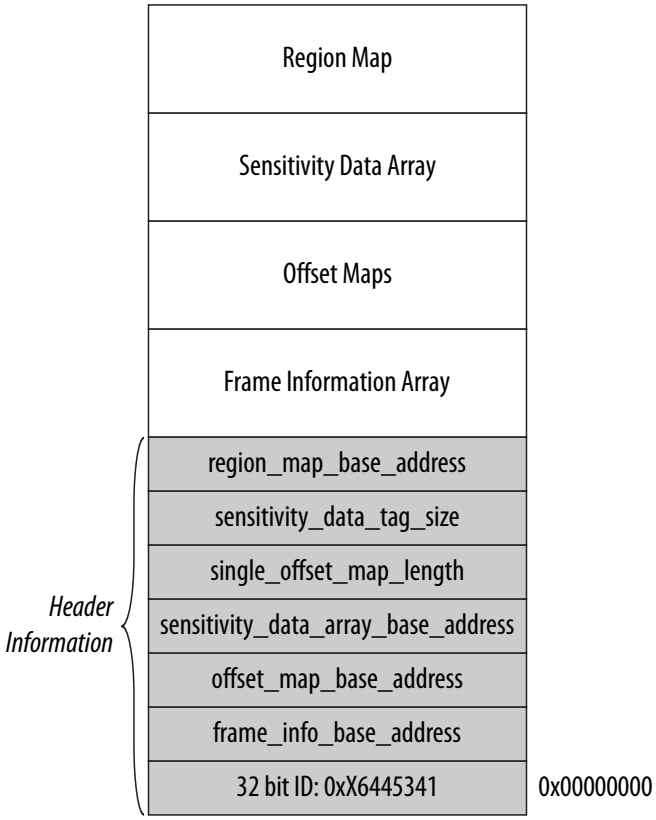
These SMH sizes are for Stratix V 5SGXEA7 device with a SOF size of 31,731,193 bytes.

Number of ASD Regions	Sensitivity Tag Size (bits)	SMH Size (bytes)
1	1	2,296,736
2-3	2	3,984,920
3-15	4	7,361,308
10-127	8	14,114,024

2.2.4.1.3. Revision 3 SMH

The revision 3 SMH file format is an extension of the revision 2 header format that accomodates longer sensitivity data addresses.

Figure 8. Revision 3 SMH



The file header information is the same as in revision 2, except it has a different 32 bit ID: 0xX6445341. The sensitivity map header definition 32 bit ID is the same as revision 2 except bit 26 indicates the usage of longer sensitivity data addresses.

The frame information array contains a 48 bit entry for each frame in the device. Like revision 2, the frame number serves as the index for the frame information entry. Each frame information entry contains:

- `offset_map_array_index`—Bits [47:32] are the 16 bit index for the offset map array.
- `frame_info_data_offset`—Bits [31:0] are the 32 bit address offset into the sensitivity array for `sensitivity_data_tag_size` = 1.

The offset map array, sensitivity data array, and region map information array have the same definition as revision 2.

3. Using the Advanced SEU Detection Intel FPGA IP

Use the Intel Quartus Prime parameter editor to generate an instance of the Advanced SEU Detection IP core. You **must** have a license to use the IP core. You cannot evaluate it with the OpenCore Plus feature.

3.1. Customizing and Generating IP Cores

You can customize IP cores to support a wide variety of applications. The Intel Quartus Prime IP Catalog and parameter editor allow you to quickly select and configure IP core ports, features, and output files.

3.1.1. IP Catalog and Parameter Editor

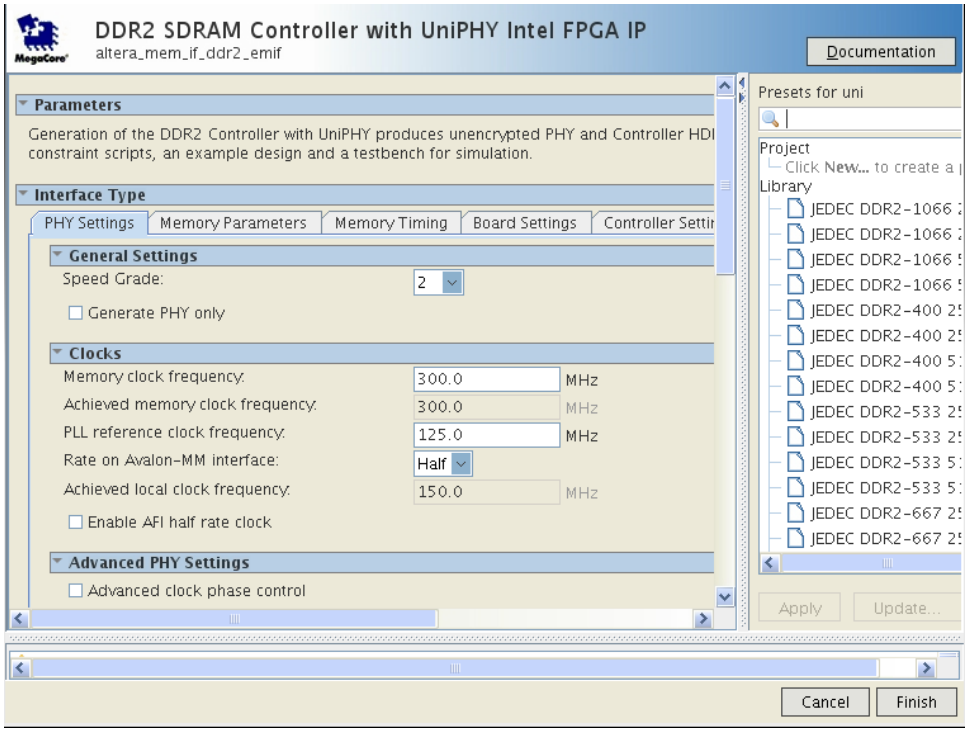
The IP Catalog displays the IP cores available for your project, including Intel FPGA IP and other IP that you add to the IP Catalog search path.. Use the following features of the IP Catalog to locate and customize an IP core:

- Filter IP Catalog to **Show IP for active device family** or **Show IP for all device families**. If you have no project open, select the **Device Family** in IP Catalog.
- Type in the Search field to locate any full or partial IP core name in IP Catalog.
- Right-click an IP core name in IP Catalog to display details about supported devices, to open the IP core's installation folder, and for links to IP documentation.
- Click **Search for Partner IP** to access partner IP information on the web.

The parameter editor prompts you to specify an IP variation name, optional ports, and output file generation options. The parameter editor generates a top-level Intel Quartus Prime IP file (`.ip`) for an IP variation in Intel Quartus Prime Pro Edition projects.

The parameter editor generates a top-level Quartus IP file (`.qip`) for an IP variation in Intel Quartus Prime Standard Edition projects. These files represent the IP variation in the project, and store parameterization information.

Figure 9. IP Parameter Editor (Intel Quartus Prime Standard Edition)



3.1.2. The Parameter Editor

The parameter editor helps you to configure IP core ports, parameters, and output file generation options. The basic parameter editor controls include the following:

- Use the **Presets** window to apply preset parameter values for specific applications (for select cores).
- Use the **Details** window to view port and parameter descriptions, and click links to documentation.
- Click **Generate ➤ Generate Testbench System** to generate a testbench system (for select cores).
- Click **Generate ➤ Generate Example Design** to generate an example design (for select cores).
- Click **Validate System Integrity** to validate a system's generic components against companion files. (Platform Designer systems only)
- Click **Sync All System Info** to validate a system's generic components against companion files. (Platform Designer systems only)

The IP Catalog is also available in Platform Designer (**View ➤ IP Catalog**). The Platform Designer IP Catalog includes exclusive system interconnect, video and image processing, and other system-level IP that are not available in the Intel Quartus Prime IP Catalog. Refer to *Creating a System with Platform Designer* or *Creating a System with Platform Designer (Standard)* for information on use of IP in Platform Designer (Standard) and Platform Designer, respectively.

Related Information

- Creating a System with Platform Designer
- Creating a System with Platform Designer (Standard)

3.1.3. Specifying IP Core Parameters and Options

Follow these steps to specify IP core parameters and options.

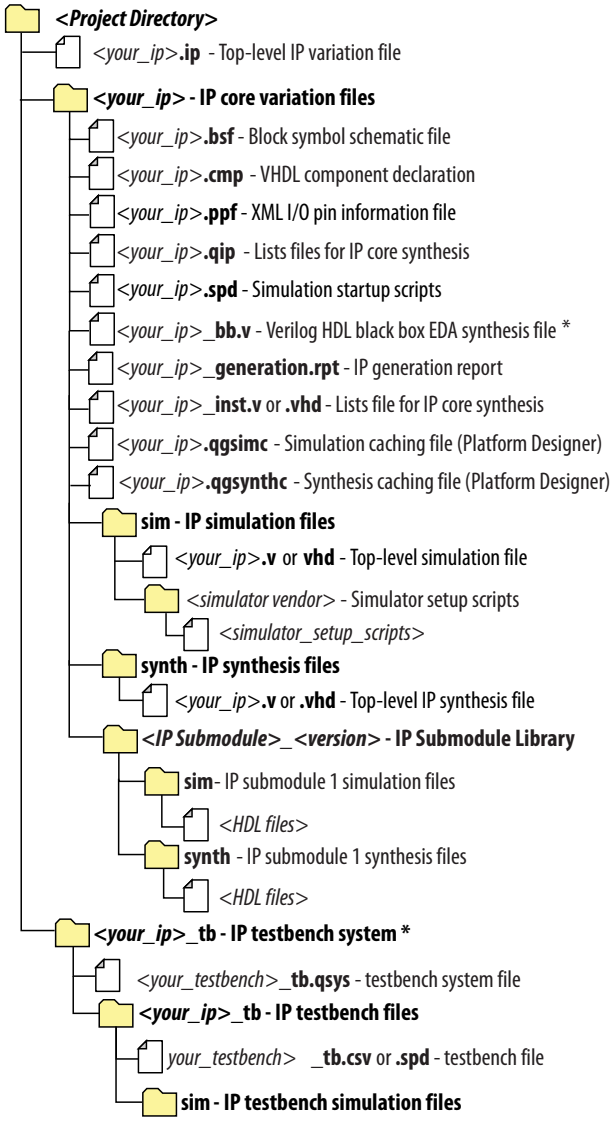
1. In the Platform Designer IP Catalog (**Tools ➤ IP Catalog**), locate and double-click the name of the IP core to customize. The parameter editor appears.
2. Specify a top-level name for your custom IP variation. This name identifies the IP core variation files in your project. If prompted, also specify the target FPGA device family and output file HDL preference. Click **OK**.
3. Specify parameters and options for your IP variation:
   - Optionally select preset parameter values. Presets specify all initial parameter values for specific applications (where provided).
   - Specify parameters defining the IP core functionality, port configurations, and device-specific features.
   - Specify options for generation of a timing netlist, simulation model, testbench, or example design (where applicable).
   - Specify options for processing the IP core files in other EDA tools.
4. Click **Finish** to generate synthesis and other optional files matching your IP variation specifications. The parameter editor generates the top-level `.qsys` IP variation file and HDL files for synthesis and simulation. Some IP cores also simultaneously generate a testbench or example design for hardware testing.
5. To generate a simulation testbench, click **Generate ➤ Generate Testbench System**. **Generate Testbench System** is not available for some IP cores that do not provide a simulation testbench.
6. To generate a top-level HDL example for hardware verification, click **Generate ➤ HDL Example**. **Generate ➤ HDL Example** is not available for some IP cores.

The top-level IP variation is added to the current Intel Quartus Prime project. Click **Project ➤ Add/Remove Files in Project** to manually add a `.qsys` (Intel Quartus Prime Standard Edition) or `.ip` (Intel Quartus Prime Pro Edition) file to a project. Make appropriate pin assignments to connect ports.

3.1.3.1. IP Core Generation Output (Intel Quartus Prime Pro Edition)

The Intel Quartus Prime software generates the following output file structure for individual IP cores that are not part of a Platform Designer system.

Figure 10. Individual IP Core Generation Output (Intel Quartus Prime Pro Edition)



* If supported and enabled for your IP core variation.

Table 5. Output Files of Intel FPGA IP Generation

File Name	Description
<your_ip>.ip	Top-level IP variation file that contains the parameterization of an IP core in your project. If the IP variation is part of a Platform Designer system, the parameter editor also generates a .qsys file.
<your_ip>.cmp	The VHDL Component Declaration (.cmp) file is a text file that contains local generic and port definitions that you use in VHDL design files.
<your_ip>_generation.rpt	IP or Platform Designer generation log file. Displays a summary of the messages during IP generation.

continued...

File Name	Description
<*your_ip*>.qgsimc (Platform Designer systems only)	Simulation caching file that compares the .qsys and .ip files with the current parameterization of the Platform Designer system and IP core. This comparison determines if Platform Designer can skip regeneration of the HDL.
<*your_ip*>.qgsynth (Platform Designer systems only)	Synthesis caching file that compares the .qsys and .ip files with the current parameterization of the Platform Designer system and IP core. This comparison determines if Platform Designer can skip regeneration of the HDL.
<*your_ip*>.qip	Contains all information to integrate and compile the IP component.
<*your_ip*>.csv	Contains information about the upgrade status of the IP component.
<*your_ip*>.bsf	A symbol representation of the IP variation for use in Block Diagram Files (.bdf).
<*your_ip*>.spd	Input file that ip-make-simscript requires to generate simulation scripts. The .spd file contains a list of files you generate for simulation, along with information about memories that you initialize.
<*your_ip*>.ppf	The Pin Planner File (.ppf) stores the port and node assignments for IP components you create for use with the Pin Planner.
<*your_ip*>_bb.v	Use the Verilog blackbox (_bb.v) file as an empty module declaration for use as a blackbox.
<*your_ip*>_inst.v or _inst.vhd	HDL example instantiation template. Copy and paste the contents of this file into your HDL file to instantiate the IP variation.
<*your_ip*>.regmap	If the IP contains register information, the Intel Quartus Prime software generates the .regmap file. The .regmap file describes the register map information of master and slave interfaces. This file complements the .sopcinfo file by providing more detailed register information about the system. This file enables register display views and user customizable statistics in System Console.
<*your_ip*>.svd	Allows HPS System Debug tools to view the register maps of peripherals that connect to HPS within a Platform Designer system. During synthesis, the Intel Quartus Prime software stores the .svd files for slave interface visible to the System Console masters in the .sof file in the debug session. System Console reads this section, which Platform Designer queries for register map information. For system slaves, Platform Designer accesses the registers by name.
<*your_ip*>.v <*your_ip*>.vhd	HDL files that instantiate each submodule or child IP core for synthesis or simulation.
mentor/	Contains a msim_setup.tcl script to set up and run a simulation.
aldec/	Contains a script rivierapro_setup.tcl to setup and run a simulation.
/synopsys/vcs /synopsys/vcsmx	Contains a shell script vcs_setup.sh to set up and run a simulation. Contains a shell script vcsmx_setup.sh and synopsys_sim.setup file to set up and run a simulation.
/cadence	Contains a shell script ncsim_setup.sh and other setup files to set up and run an simulation.
/xcelium	Contains an Parallel simulator shell script xcelium_setup.sh and other setup files to set up and run a simulation.
/submodules	Contains HDL files for the IP core submodule.
<*IP submodule*>/	Platform Designer generates /synth and /sim sub-directories for each IP submodule directory that Platform Designer generates.

3.2. Advanced SEU Detection IP Core Parameters

Parameter Group	Parameter		Description
	Name	Legal Value	
General	CRC error cache depth	2, 4, 8,16, 32, 64	• Specifies how many non-critical cyclic redundancy check (CRC) error to ignore. • Default value is 8.
	Largest ASD region ID	1 to 16	• Indicates the largest ASD SEU detection region ID in your design. • Configures the width of the `regions_report` port. • Default value is 1.
Sensitivity Data Access	Use on-chip sensitivity processing	ON, OFF	• Configures the IP core to use on-chip sensitivity processing or off-chip sensitivity processing. • When enabled, implements an external memory interface in the IP.
	Memory interface address width	—	• Specifies width of the address bus connected to the external memory interface. • Default value is 32. For on-chip sensitivity processing only.
	Sensitivity data start address	—	• Specifies the offset added to all addresses the external memory interface generates. • Default value is 0x0. For on-chip sensitivity processing only.

4. SEU Mitigation on CRAM Array

Critical applications require an SEU recovery strategy. The Intel Quartus Prime software provides SEU detection, and allows you to design a recovery response to reduce SEU disruption.

4.1. Enabling the Advanced SEU Detection Feature in the Intel Quartus Prime Software

To enable the Advanced SEU Detection feature in the Intel Quartus Prime software and generate an `.smh`, turn on **Generate SEU sensitivity map file (.smh)** in the **Device and Pin Options** dialog box (**Assignments > Device > Device and Pin Options**).

Note: You must have a licensed version of Intel Quartus Prime software to generate SMH files.

4.2. Hierarchy Tagging

The Intel Quartus Prime hierarchy tagging feature enables customized soft error classification by indicating design logic susceptible to soft errors. Hierarchy tagging improves design-effective FIT rate by tagging only the critical logic for device operation. You also define the system recovery procedure based on knowledge of logic impaired by SEU. This technique reduces downtime for the FPGA and the system in which the FPGA resides. The Intel Arria 10, Intel Cyclone 10 GX, Stratix V, Arria V, and Cyclone V devices support hierarchy tagging.

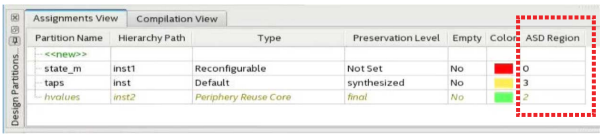
The `.smh` contains a mask for design sensitive bits in a compressed format. The Intel Quartus Prime software generates the sensitivity mask for the entire design. Hierarchy tagging provides the following benefits:

- Increases system stability by avoiding disruptive recovery procedures for inconsequential errors.
- Allows diverse corrective action for different design logic.

4.2.1. Using Partitions to Specify Logic Sensitivity ID

1. In the Intel Quartus Prime software, designate a design block as a design partition.
2. Specify the sensitivity ID assigned to the partition in the **ASD Region** column in the **Design Partitions** window.

Figure 11. ASD Region Column in the Design Partitions Window



Assign the partition a numeric sensitivity value from 0 to 16. The value represents the sensitivity tag associated with the partition.

- A sensitivity tag of 1 is the same as no assignment, and indicates a basic sensitivity level, which is "region used in design". If a soft error occurs in this partition, the Advanced SEU Detection IP core reports the error as a critical error in the sensitivity region 1.
- A sensitivity tag of 0 is reserved, and indicates unused CRAM bits. You can explicitly set a partition to 0 to indicate that the partition is not critical. This setting excludes the partition from sensitivity mapping.

Note: You can use the same sensitivity tag for multiple design partitions.

Alternatively, use the following assignment:

```
set_global_assignment -name PARTITION_ASD_REGION_ID <asd_id> -section_id
<partition_name>
```

4.3. Sensitivity Map Header File Lookup

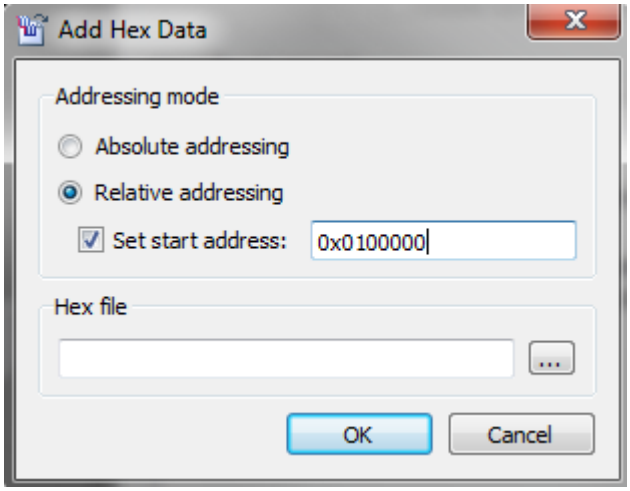
The `.smh` contains critical bit information about the design. The Intel Quartus Prime software generates sensitivity data as a standard Intel hex (big-endian) `.smh` file during `.sof` generation.

4.3.1. Programming Sensitivity Map Header File into Memory

You can program an `.smh` into any type of memory. For example, to use CFI flash memory, follow these steps:

1. Rename the `.smh` to <*file_name*>**.hex**, or convert it to little-endian <*file_name*>**.hex** if required.
2. In the Intel Quartus Prime software, click **File > Convert Programming Files**.
3. In the **Convert Programming Files** window under **Output programming file**, select the desired options.
4. To add hex data, follow these steps:
   a. Click **Add Hex Data**.
   b. In the **Add Hex Data** dialog box, turn on **Set start address** and enter a start address.
   c. In the **Hex file box**, click browse to select the `.hex` file, and click **OK**.

Figure 12. Add Hex Data Dialog Box



5. Click **Generate**.

4.3.2. Performing a Lookup for SMH Revision 1 (Stratix IV and Arria II)

To perform a lookup into the sensitivity map header data using a bit, byte, and frame number from an EMR for Stratix IV and Arria II devices:

1. Read the 32 bit frame information string for the frame number:
   - Address = <*frame_info_base_address*> + (frame*4)
   - Return value = (`frame_info_data_offset`, `offset_map_array_index`)
2. Read the offset map information for a frame. The return value for the offset map information is 16 bits:
   - Address = `offset_map_base_address` + offset array for current frame + offset data value for current byte and bit

   Where:
   - Offset array for current frame = `offset_map_array_index` * `offset_map_length`
   - Offset data value for current byte and bit = [(byte * 8) + bit] * 2
   - Return value = `offset_map_value`
3. Read the 8 bit sensitivity value:
   - Address = `sensitivity_data_array_base_address` + `frame_info_data_offset` + (`offset_map_value`/8)
   - Return value = `sensitive_bit_word[7:0]`
4. Read the sensitive bit. The offset map value provides the sensitive bit index. A value of 1 indicates a critical bit, and a value of 0 indicates a non-critical bit.
   - Sensitive bit = `sensitive_bit_word[bit_index]`

   Where:
   - bit_index = `offset_map_value[2:0]`

4.3.3. Performing a Lookup for SMH Revision 2 (Stratix V, Arria V, and Cyclone V Devices)

To perform a lookup into the sensitivity map header data using a bit, byte, and frame number from an EMR for Stratix V, Arria V, and Cyclone V devices:

1. Read the 32 bit frame information string for the frame number:
   - Address = <*frame_info_base_address*> + (frame*4)
   - Return value = (`frame_info_data_offset`, `offset_map_array_index`)
2. Read the frame's offset map information. The return value is 16 bits.
   - Address = `offset_map_base_address` + offset array for current frame + offset data value for current byte and bit

   Where:
   - Offset array for the current frame = `offset_map_array_index` * `offset_map_length`
   - Offset data value for the current byte and bit = [(byte * 8) + bit] * 2
   - Return value = `offset_map_value`
3. Read the 8 bit sensitivity value:
   - Address = `sensitivity_data_array_base_address` + `frame_info_data_offset` + (`offset_map_value` * `sensitivity_data_tag_size` / 8)
   - Return value = `sensitive_bit_word[7:0]`
4. Read the sensitivity data tag. The offset map value provides the sensitive bit index. The return value for the sensitivity tag is `sensitivity_data_tag_size` bit length. A zero tag indicates that the bit is not critical for any region; a non-zero tag indicates an offset in the region map.

   `sensitive_tag` = (`sensitive_data word >> tag_shift`) & `tag_mask`

   Where:
   - `tag_shift` = (`offset_map_value` * `sensitivity_data_tag_size`)[2:0]
   - `tag_mask` = (0x1 << `sensitivity_data_tag_size`) - 1
5. Read the region mask for a non-zero sensitivity tag only. The return value for the region mask is 16 bits.

   `region_mask` = `region_map_base_address` + (`sensitivity_data_tag` - 1) * 2

4.3.4. Performing a Lookup for SMH Revision 3 (Intel Arria 10 and Intel Cyclone 10 GX Devices)

To perform a lookup into the sensitivity map header data using a bit, byte, and frame number from an EMR for Intel Arria 10, Intel Cyclone 10 GX devices:

1. Read the frame information entry's higher two bytes to obtain the frame index in the offset map array:

- Address = `frame_info_base_address` + (frame address * 6)
- Return value = `offset_map_array_index`

2. Read the frame information entry's lower four bytes to obtain the frame sensitivity data offset:
   - Address = `frame_info_base_address` + (frame address * 6) + 2
   - Return value = `frame_info_data_offset`
3. Read the frame's offset map information. The return value is 16 bits.
   - Address = `offset_map_base_address` + offset array for current frame + offset data value for current frame-based double word and frame-based bit

   Where:
   - Offset array for the current frame = `offset_map_array_index` * `offset_map_length`
   - Offset data value for the current frame-based double word and frame-based bit = [(double word * 32) + bit] * 2
   - Return value = `offset_map_value`
4. Read the 8-bit sensitivity value:
   - Address = `sensitivity_data_array_base_address` + `frame_info_data_offset` * `sensitivity_data_tag_size` + (`offset_map_value` * `sensitivity_data_tag_size` / 8)
   - Return value = `sensitive_data_word[7:0]`
5. Read the sensitivity data tag. The offset map value provides the sensitive bit index. The return value for the sensitivity tag is `sensitivity_data_tag_size` bit length. A zero tag indicates that the bit is not critical for any region; a non-zero tag indicates an offset in the region map.

   `sensitive_tag` = (`sensitive_data_word` >> `tag_shift`) & `tag_mask`

   Where:
   - `tag_shift` = (`offset_map_value` * `sensitivity_data_tag_size`)[2:0]
   - `tag_mask` = (0x1 << `sensitivity_data_tag_size`) - 1
6. Read the region mask for a non-zero sensitivity tag only. The return value for the region mask is 16 bits.

   `region_mask` = `region_map_base_address` + (`sensitivity_data_tag` - 1) * 2

5. Advanced SEU Detection Intel FPGA IP User Guide Archives

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
18.1	Intel FPGA Advanced SEU Detection IP Core User Guide
17.1	Intel FPGA Advanced SEU Detection IP Core User Guide
16.1	Altera Advanced SEU Detection IP Core User Guide
16.0	Altera Advanced SEU Detection IP Core User Guide

6. Document Revision History for the Advanced SEU Detection Intel FPGA IP User Guide

Document Version	Intel Quartus Prime Version	Changes
2019.03.26	18.1	Corrected the link to the user guide archive for version 16.0 of the Advanced SEU Detection Intel FPGA IP.
2019.01.14	18.1	Updated the description of cache_valid in the table listing the IP core the signals for off-chip processing.
2018.09.12	18.0	• Updated the topic about performing a lookup for SMH Revision 3 (Intel Arria 10 and Intel Cyclone 10 GX Devices). • Updated document structure. • Corrected a broken link to the archived user guide for the 17.1 version of the Advanced SEU Detection IP core.
2018.05.16	18.0	• Added note to *Advanced SEU Detection Core Signals for On-Chip Processing* and *Advanced SEU Detection Core Signals for Off-Chip Processing* to indicate reserved bits in `emr`. • Renamed "Intel FPGA Advanced SEU Detection" to "Advanced SEU Detection Intel FPGA IP" as per Intel rebranding. • Added the following topics from Intel Quartus Prime Pro Edition Handbook: — *On-Chip Sensitivity Processor* section. — *Off-Chip Sensitivity Processor* and subtopics.

Date	Version	Changes
November 2017	2017.11.06	• Rebranded as Intel. • Added Intel Cyclone 10 GX device support.
October 2016	2016.10.31	• Added sensitivity processing as supported feature for Stratix V, Arria 10, Arria V, Arria V GZ, and Cyclone V. • Updated sensitivity map header information names.
May 2016	2016.05.02	Clarified the frame information array information for revision 1 `.smh` files.
November 2015	2015.11.02	• Added Arria 10 support information for SMH revision 3 lookups. • Updated information on SMH revision 2 lookups. • Updated on-chip and off-chip processing signals. • Changed Quartus II references to Quartus Prime.
May 2015	2015.05.04	• Added note to possible values for `sensitivity_data_tag_size` field. • Updated largest ASD region ID in the Altera Advanced SEU Detection IP Core Parameters. • Updated supported device for performing lookup for Revision 1 and 2.

continued...

Date	Version	Changes
		• Updated features and device family support by combining in a table. • Removed duplicated signals in Off-Chip processing signals table. • Updated SMH frame information array description to reduce redundancies.
June 30 2014	2014.06.30	• Updated supported devices. • Replaced information about the MegaWizard Plug-in Manager with the IP Catalog.
December 2012	1.0	Initial release.