Altera Fault Injection IP Core User Guide

2015.12.15

UG-01173

Subscribe Send Feedback

The Altera® Fault Injection IP core injects errors into the configuration RAM (CRAM) of an FPGA device.

This procedure simulates soft errors that can occur during normal operation due to single event upsets (SEUs). SEUs are rare events, and are therefore difficult to test. After you instantiate the Fault Injection IP core into your design and configure your device, you can use the Quartus® Prime Fault Injection Debugger tool to induce intentional errors in the FPGA to test the system's response to these errors.

Related Information

- **Debugging Single Event Upset Using the Fault Injection Debugger (Quartus Prime Standard Edition Handbook Volume 3: Verification)**
- **Debugging Single Event Upset Using the Fault Injection Debugger (Quartus Prime Pro Edition Handbook Volume 3: Verification)**

Features

- Allows you to evaluate system response for mitigating single event functional interrupts (SEFI).
- Allows you to perform SEFI characterization in-house, eliminating the need for entire system beam testing. Instead, you can limit the beam testing to failures in time (FIT)/Mb measurement at the device level.
- Scale FIT rates according to the SEFI characterization that is relevant to your design architecture. You can randomly distribute fault injections throughout the entire device, or constrain them to specific functional areas to speed up testing.
- Optimize your design to reduce disruption caused by a single event upsets (SEU).

Device Support

The Fault Injection IP core supports Stratix® V family devices. For assistance with support for Arria® V or Cyclone® V family devices, file a service request using mySupport on the Altera website.

Resource Utilization and Performance

The Quartus Prime software generates the following resource estimate for the Stratix A7 FPGA. Results for other devices are similar.

101 Innovation Drive, San Jose, CA 95134

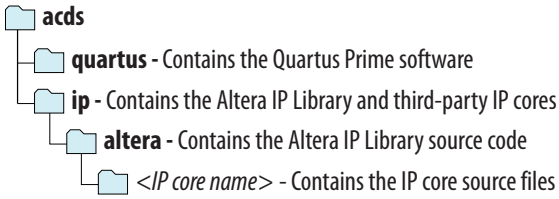
Table 1: Fault Injection IP Core FPGA Performance and Resource Utilization

Device	ALMs	Logic Registers		M20K
		Primary	Secondary	
Stratix V A7	3,821	5,179	0	0

Licensing IP Cores

The Altera IP Library provides many useful IP core functions for your production use without purchasing an additional license. Some Altera MegaCore® IP functions require that you purchase a separate license for production use. However, the OpenCore® feature allows evaluation of any Altera IP core in simulation and compilation in the Quartus Prime software. After you are satisfied with functionality and performance, visit the Self Service Licensing Center to obtain a license number for any Altera product.

Figure 1: IP Core Installation Path



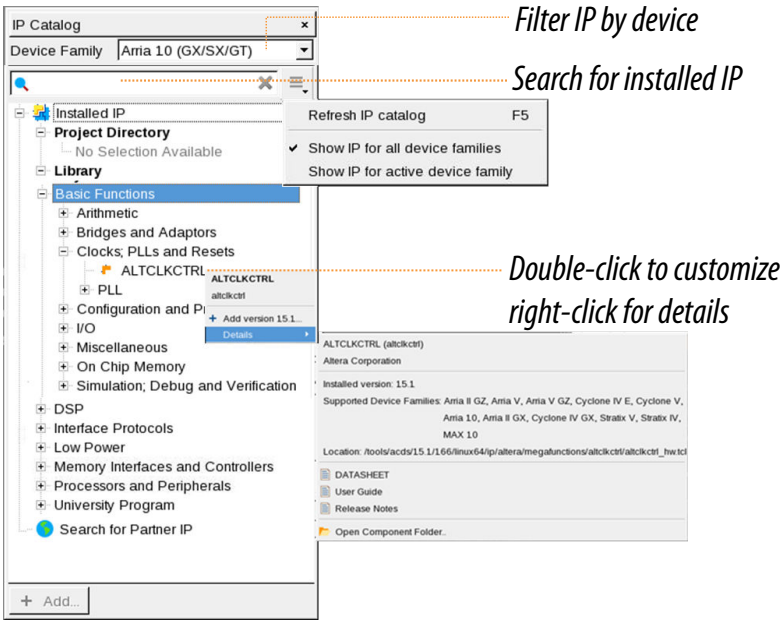
Note: The default IP installation directory on Windows is **<*drive*>:\altera**<*version number*>; on Linux it is <*home directory*>**/altera/** <*version number*>.

Customizing and Generating IP Cores

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

IP Catalog and Parameter Editor

The IP Catalog (**Tools** > **IP Catalog**) and parameter editor help you easily customize and integrate IP cores into your project. Use the IP Catalog and parameter editor to select, customize, and generate files representing the custom IP variation in your project.



The IP Catalog displays the installed IP cores available for your design. Double-click any IP core to launch the parameter editor and generate files representing your IP variation. Use the following features to help you quickly locate and select 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, open the IP core's installation folder, and click links to IP documentation.
- Click **Search for Partner IP**, to access partner IP information on the Altera website.

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 Qsys system file (**.qsys**) or Quartus Prime IP file (**.qip**) representing the IP core in your project. You can also parameterize an IP variation without an open project.

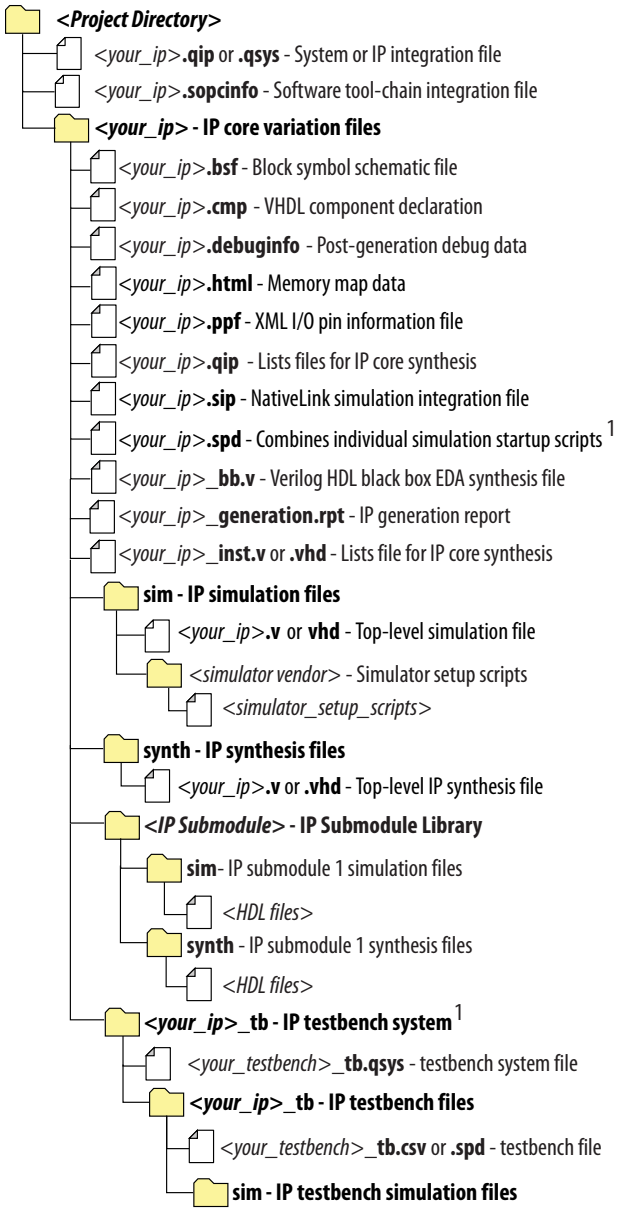
The IP Catalog is also available in Qsys (**View** > **IP Catalog**). The Qsys IP Catalog includes exclusive system interconnect, video and image processing, and other system-level IP that are not available in the Quartus Prime IP Catalog. For more information about using the Qsys IP Catalog, refer to *Creating a System with Qsys* in the *Quartus Prime Handbook*.

Note: The IP Catalog (**Tools** > **IP Catalog**) and parameter editor replace the MegaWizard™ Plug-In Manager for IP selection and parameterization, beginning in Quartus II software version 14.0. Use the IP Catalog and parameter editor to locate and paramaterize Altera IP cores.

Files Generated for Altera IP Cores and Qsys Systems

The Quartus Prime software generates the following output file structure for IP cores and Qsys systems. For Arria 10 devices and newer, the generated **.qsys** file must be added to your project to represent IP and Qsys systems. For devices released prior to Arria 10 devices, the generated **.qip** and **.sip** files must be added to your project to represent IP and Qsys systems.

Figure 2: Files generated for IP cores and Qsys Systems



1. If supported and enabled for your IP core variation.

Table 2: IP Core and Qsys Simulation Generated Files

File Name	Description
<*my_ip*>.qsys	The Qsys system or top-level IP variation file. <*my_ip*> is the name that you give your IP variation. You must add the **.qsys** file to your Quartus project to enable NativeLink for Arria 10 and Stratix 10 device families.
<*system*>.sopcinfo	Describes the connections and IP component parameterizations in your Qsys system. You can parse its contents to get requirements when you develop software drivers for IP components. Downstream tools such as the Nios II tool chain use this file. The **.sopcinfo** file and the **system.h** file generated for the Nios II tool chain include address map information for each slave relative to each master that accesses the slave. Different masters may have a different address map to access a particular slave component.
<*my_ip*>.cmp	The VHDL Component Declaration (**.cmp**) file is a text file that contains local generic and port definitions that you can use in VHDL design files.
<*my_ip*>.html	A report that contains connection information, a memory map showing the address of each slave with respect to each master to which it is connected, and parameter assignments.
<*my_ip*>_generation.rpt	IP or Qsys generation log file. A summary of the messages during IP generation.
<*my_ip*>.debuginfo	Contains post-generation information. Used to pass System Console and Bus Analyzer Toolkit information about the Qsys interconnect. The Bus Analysis Toolkit uses this file to identify debug components in the Qsys interconnect.
<*my_ip*>.qip	Contains all the required information about the IP component to integrate and compile the IP component in the Quartus Prime software.
<*my_ip*>.csv	Contains information about the upgrade status of the IP component.
<*my_ip*>.bsf	A Block Symbol File (**.bsf**) representation of the IP variation for use in Quartus Prime Block Diagram Files (**.bdf**).
<*my_ip*>.spd	Required input file for `ip-make-simscript` to generate simulation scripts for supported simulators. The **.spd** file contains a list of files generated for simulation, along with information about memories that you can initialize.
<*my_ip*>.ppf	The Pin Planner File (**.ppf**) stores the port and node assignments for IP components created for use with the Pin Planner.
<*my_ip*>_bb.v	You can use the Verilog black-box (**_bb.v**) file as an empty module declaration for use as a black box.

File Name	Description
<*my_ip*>.sip	Contains information required for NativeLink simulation of IP components. You must add the **.sip** file to your Quartus project to enable NativeLink for Arria II, Arria V, Cyclone IV, Cyclone V, MAX 10, MAX II, MAX V, Stratix IV, and Stratix V devices.
<*my_ip*>_inst.v or **_inst.vhd**	HDL example instantiation template. You can copy and paste the contents of this file into your HDL file to instantiate the IP variation.
<*my_ip*>.regmap	If the IP contains register information, the .**regmap** file generates. 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 enables register display views and user customizable statistics in System Console.
<*my_ip*>.svd	Allows HPS System Debug tools to view the register maps of peripherals connected to HPS within a Qsys system. During synthesis, the .**svd** files for slave interfaces visible to System Console masters are stored in the **.sof** file in the debug section. System Console reads this section, which Qsys can query for register map information. For system slaves, Qsys can access the registers by name.
<*my_ip*>.v or **<*my_ip*>.vhd**	HDL files that instantiate each submodule or child IP core for synthesis or simulation.
mentor/	Contains a ModelSim® script **msim_setup.tcl** to set up and run a simulation.
aldec/	Contains a Riviera-PRO 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 VCS® simulation. Contains a shell script **vcsmx_setup.sh** and **synopsys_ sim.setup** file to set up and run a VCS MX® simulation.
/cadence	Contains a shell script **ncsim_setup.sh** and other setup files to set up and run an NCSIM simulation.
/submodules	Contains HDL files for the IP core submodule.
<*IP submodule*>/	For each generated IP submodule directory, Qsys generates **/synth** and **/sim** sub-directories.

Instantiating the Fault Injection IP Core

The Fault Injection IP core does not require you to set any parameters. To use the IP core, create a new IP instance, include it in your Qsys system, and connect the signals as appropriate.

Note: You must use the Fault Injection IP core with the Error Message Register (EMR) Unloader IP core.

The Fault Injection and the EMR Unloader IP cores are available in Qsys and the IP Catalog. Optionally, you can instantiate them directly into your RTL design, using Verilog HDL, SystemVerilog, or VHDL.

Using the EMR Unloader IP Core

The EMR Unloader IP core provides an interface to the EMR, which is updated continuously by the device's EDCRC that checks the device's CRAM bits CRC for soft errors.

Figure 3: Example Qsys System Including the Fault Injection IP Core and EMR Unloader IP Core

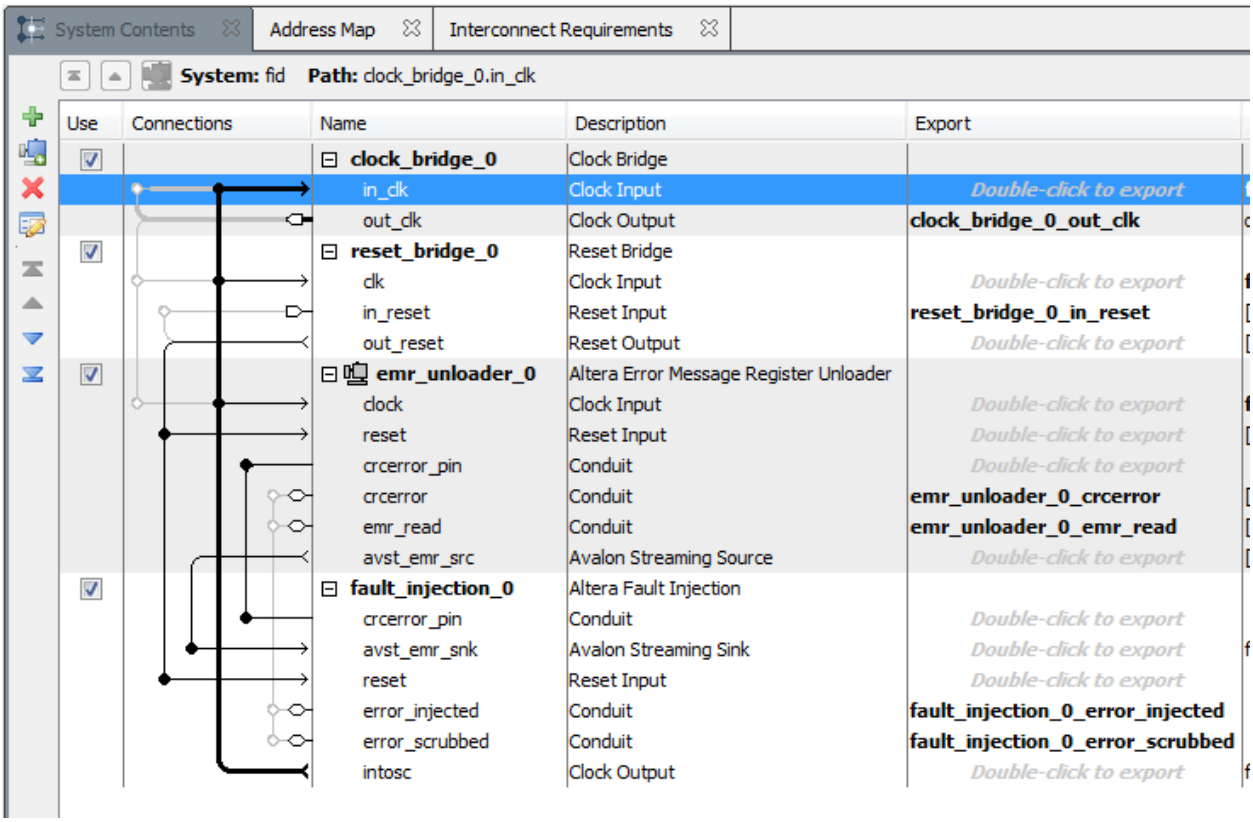
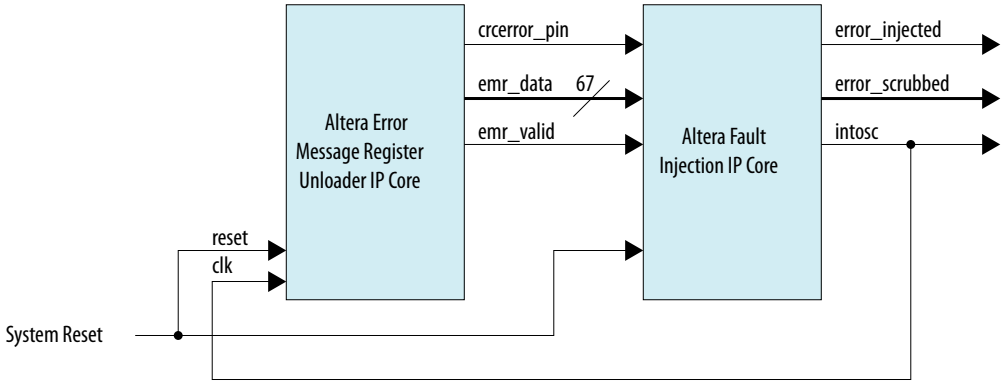


Figure 4: Example Altera Fault Injection IP Core and EMR Unloader IP Core Block Diagram



Related Information

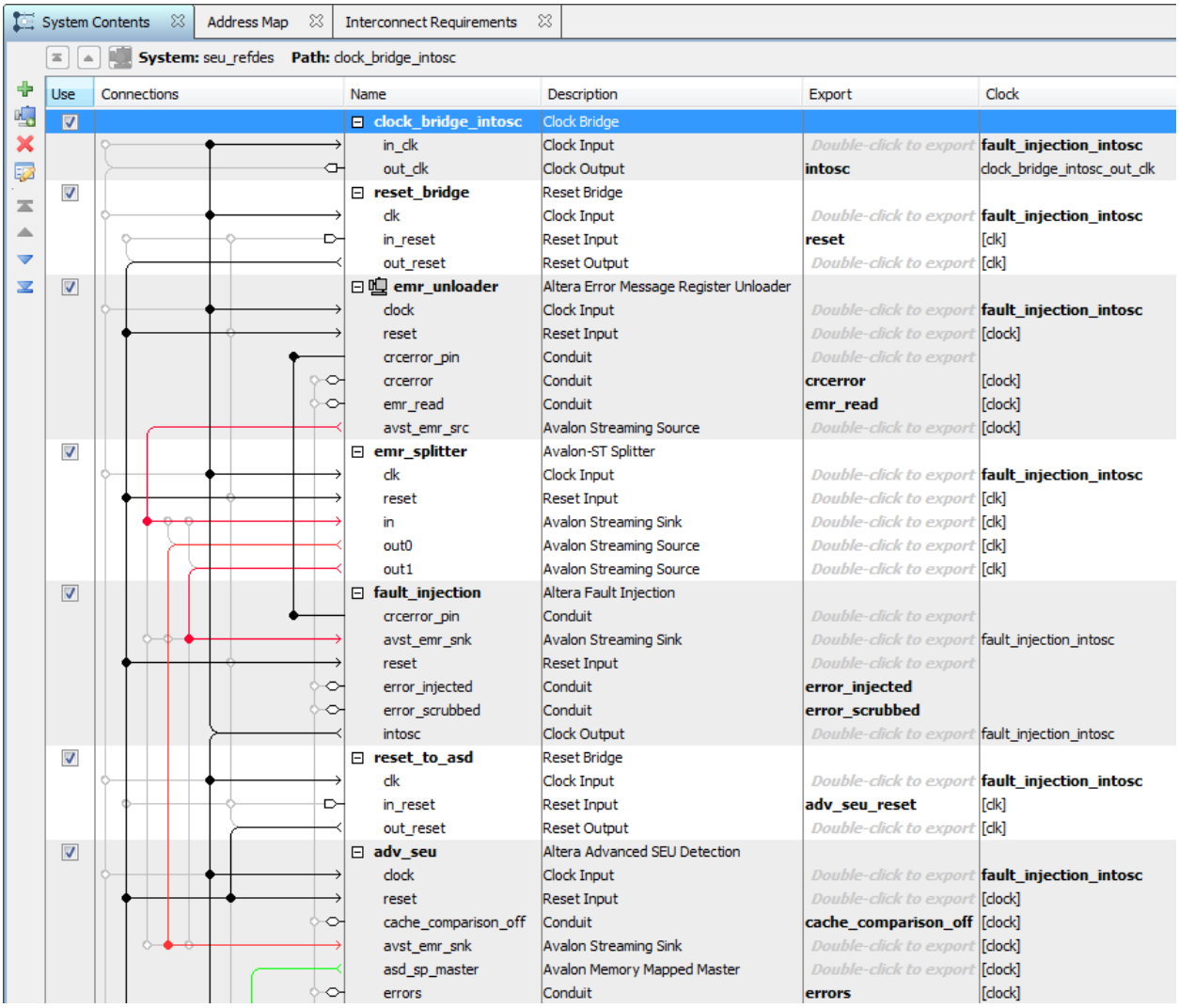
- **Altera Error Message Unloader IP Core User Guide**

Using the Advanced SEU Detection IP Core

Use the Advanced SEU Detection (ASD) IP core when SEU tolerance is a design concern.

You must use the EMR Unloader IP core with the ASD IP core. Therefore, if you use the ASD IP and the Fault Injection IP in the same design, they must share the EMR Unloader output via an Avalon-ST splitter component. The following figure shows a Qsys system in which an Avalon-ST splitter distributes the EMR contents to the ASD and Fault Injection IP cores.

Figure 5: Using the ASD and Fault Injection IP in the Same Qsys System



Related Information

- **Altera Advanced SEU Detection (ALTERA_ADV_SEU_DETECTION) IP Core User Guide**

Functional Description

With the Altera Fault Injection IP core, designers can perform SEFI characterization in-house, scale FIT rates according to SEFI characterization, and optimize designs to reduce effect of SEUs.

Single Event Upset Mitigation

Integrated circuits and programmable logic devices such as FPGAs are susceptible to SEUs. SEUs are random, nondestructive events, caused by two major sources: alpha particles and neutrons from cosmic

rays. Radiation can cause either the logic register, embedded memory bit, or a configuration RAM (CRAM) bit to flip its state, thus leading to unexpected device operation.

Arria V, Cyclone V, Stratix V and newer devices have the following CRAM capabilities:

- Error Detection Cyclical Redundance Checking (EDCRC)
- Automatic correction of an upset CRAM (scrubbing)
- Ability to create an upset CRAM condition (fault injection)

For more information about SEU mitigation in Altera devices, refer to the *SEU Mitigation* chapter in the respective device handbook.

Related Information

Altera Website: Single Event Upsets

Using the Fault Injection Debugger and Fault Injection IP Core

The Fault Injection Debugger works together with the Fault Injection IP core. First, you instantiate the IP core in your design, compile, and download the resulting configuration file into your device. Then, you run the Fault Injection Debugger from within the Quartus Prime software or from the command line to simulate soft errors.

The Fault Injection Debugger communicates with the Fault Injection IP core via the JTAG interface. You perform debugging using the Fault Injection Debugger in the Quartus Prime software or using the command-line interface.

- The Fault Injection Debugger allows you to operate fault injection experiments interactively or by batch commands, and allows you to specify the logical areas in your design for fault injections.
- The command-line interface is useful for running the debugger via a script.

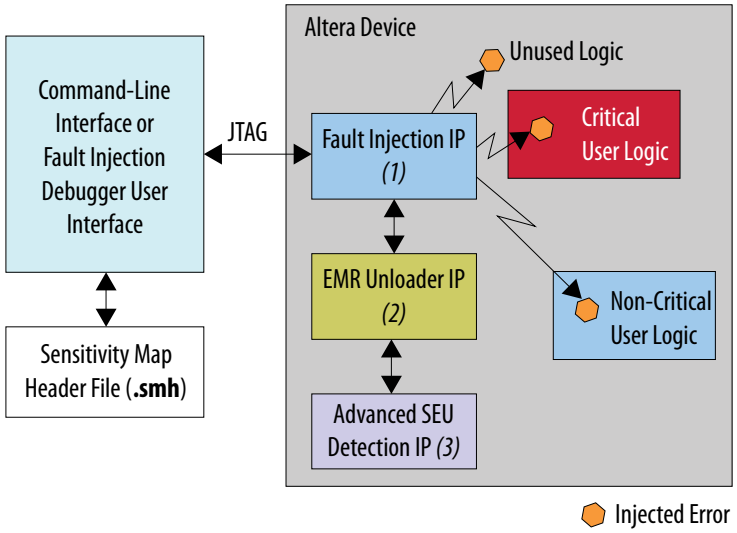
The Fault Injection IP accepts commands from the JTAG interface and reports status back through the JTAG interface.

Note: The Fault Injection IP core is implemented in soft logic in your device; therefore, you must account for this logic usage in your design. One methodology is to characterize your design's response to SEU in the lab and then omit the IP core from your final deployed design.

You use the Fault Injection IP core with the following IP cores:

- The Error Message Register (EMR) Unloader IP core, which reads and stores data from the hardened error detection circuitry in Altera devices.
- (Optional) The Advanced SEU Detection (ASD) IP core, which compares single-bit error locations to a sensitivity map during device operation to determine whether a soft error affects it.

Figure 6: Fault Injection Debugger Overview Block Diagram



Notes:
1. The fault Injection IP flips the bits of the targeted logic.
2. The Fault Injection Debugger and Advanced SEU Detection IP use the same EMR Unloader instance.
3. The Advanced SEU Detection IP core is optional.

Related Information

- Download Center
- AN 539: Test Methodology or Error Detection and Recovery using CRC in Altera FPGA Devices
- Understanding Single Event Functional Interrupts in FPGA Designs White Paper
- Altera Fault Injection IP Core User Guide
- Altera Error Message Unloader IP Core User Guide
- Altera Advanced SEU Detection (ALTERA_ADV_SEU_DETECTION) IP Core User Guide

Altera Fault Injection IP Pin Description

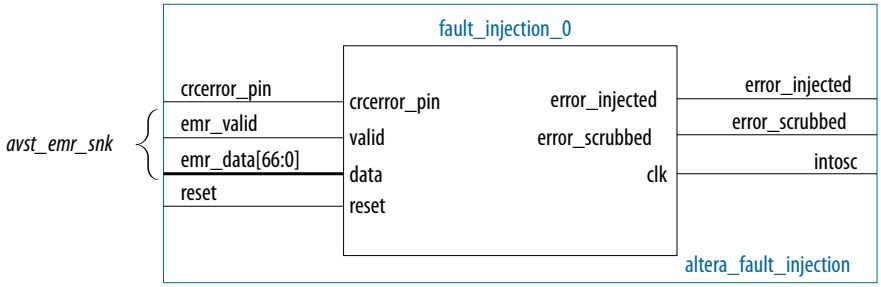
The Altera Fault Injection IP core includes the following I/O pins.

Table 3: Altera Fault Injection IP Core I/O Pins

Pin Name	Pin Direction	Pin Description
`crcerror_pin`	input	Input from Altera Error Message Register Unloader IP. This signal is asserted when a CRC error has been detected by the device's EDCRC.

Pin Name	Pin Direction	Pin Description
emr_data	input	Error Message Register (EMR) contents. Refer to the appropriate device handbook for the EMR fields. This input complies with the Avalon Streaming data interface signal.
emr_valid	input	Indicates the emr_data inputs contain valid data. This is an Avalon Streaming valid interface signal.
Reset	input	Module reset input.
error_injected	output	Indicates an error was injected into CRAM as commanded via the JTAG interface.
error_scrubbed	output	Indicates the device scrubbing is complete as commanded via the JTAG interface.
intosc	output	Optional output. The Altera Fault Injection IP uses this clock, for example, to clock the EMR_ unloader block.

Figure 7: Altera Fault Injection IP Pin Diagram



Document Revision History

Table 4: Document Revision History

Date	Version	Changes
2015.12.15	15.1	• Changed Quartus II to Quartus Prime software. • Fixed self-referencing related link.

Date	Version	Changes
2015.05.04	15.0	Initial release.