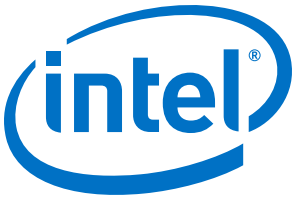


Intel® FPGA Fault Injection IP Core User Guide

Updated for Intel® Quartus® Prime Design Suite: **17.1**


Subscribe


Send Feedback

UG-01173 | 2017.11.06

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

Contents

1 Intel® FPGA Fault Injection IP Core User Guide

The Intel® FPGA 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 Intel Quartus® Prime Fault Injection Debugger tool to induce intentional errors in the FPGA to test the system's response to these errors.

Related Links

- 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)

1.1 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).

1.2 Device Support

The Fault Injection IP core supports Intel Arria® 10, Intel Cyclone® 10 GX and Stratix V family devices. The Cyclone V family supports Fault Injection on devices with the -SC suffix in the ordering code. Contact your local sales representative for ordering information on -SC suffix Cyclone V devices.

1.3 Resource Utilization and Performance

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

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

1.4 Installing and Licensing Intel FPGA IP Cores

The Intel Quartus Prime software installation includes the Intel FPGA IP library. This library provides many useful IP cores for your production use without the need for an additional license. Some Intel FPGA IP cores require purchase of a separate license for production use. The Intel FPGA IP Evaluation Mode allows you to evaluate these licensed Intel FPGA IP cores in simulation and hardware, before deciding to purchase a full production IP core license. You only need to purchase a full production license for licensed Intel IP cores after you complete hardware testing and are ready to use the IP in production.

The Intel Quartus Prime software installs IP cores in the following locations by default:

Figure 1. IP Core Installation Path

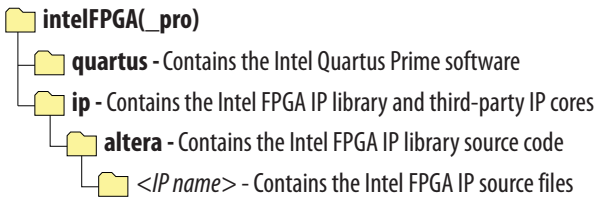


Table 2. IP Core Installation Locations

Location	Software	Platform
`<drive>:\intelFPGA_pro\quartus\ip\altera`	Intel Quartus Prime Pro Edition	Windows*
`<drive>:\intelFPGA\quartus\ip\altera`	Intel Quartus Prime Standard Edition	Windows
`<home directory>:/intelFPGA_pro/quartus/ip/altera`	Intel Quartus Prime Pro Edition	Linux*
`<home directory>:/intelFPGA/quartus/ip/altera`	Intel Quartus Prime Standard Edition	Linux

1.5 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.

1.5.1 IP Catalog and Parameter Editor

The IP Catalog displays the IP cores available for your project. 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 2. IP Parameter Editor (Intel Quartus Prime Pro Edition)

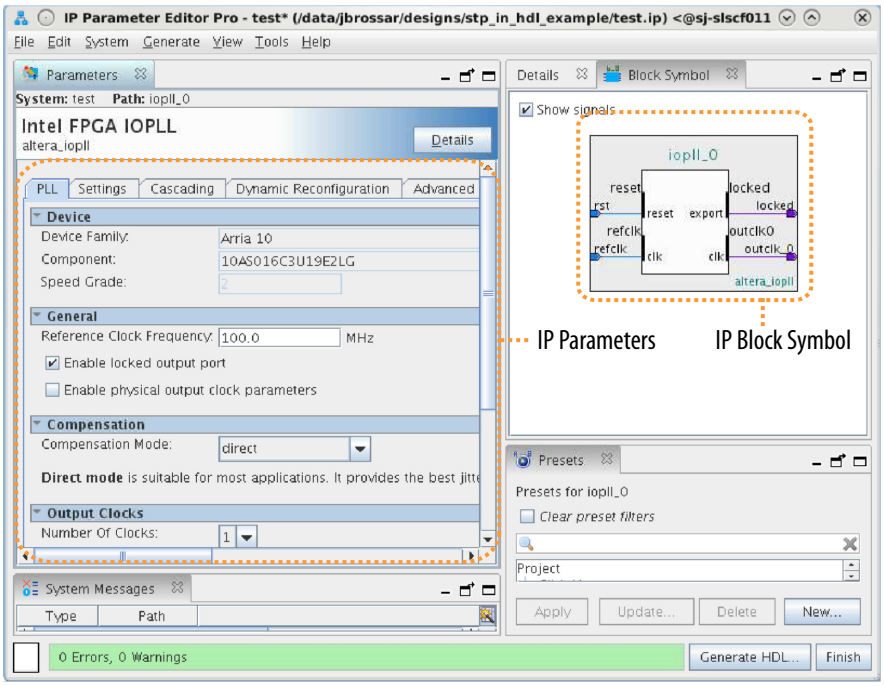
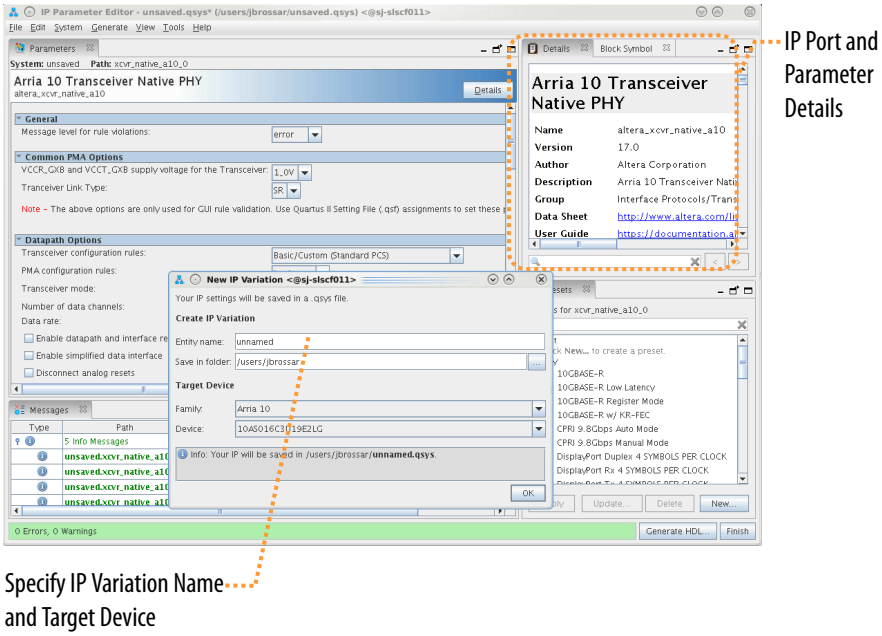


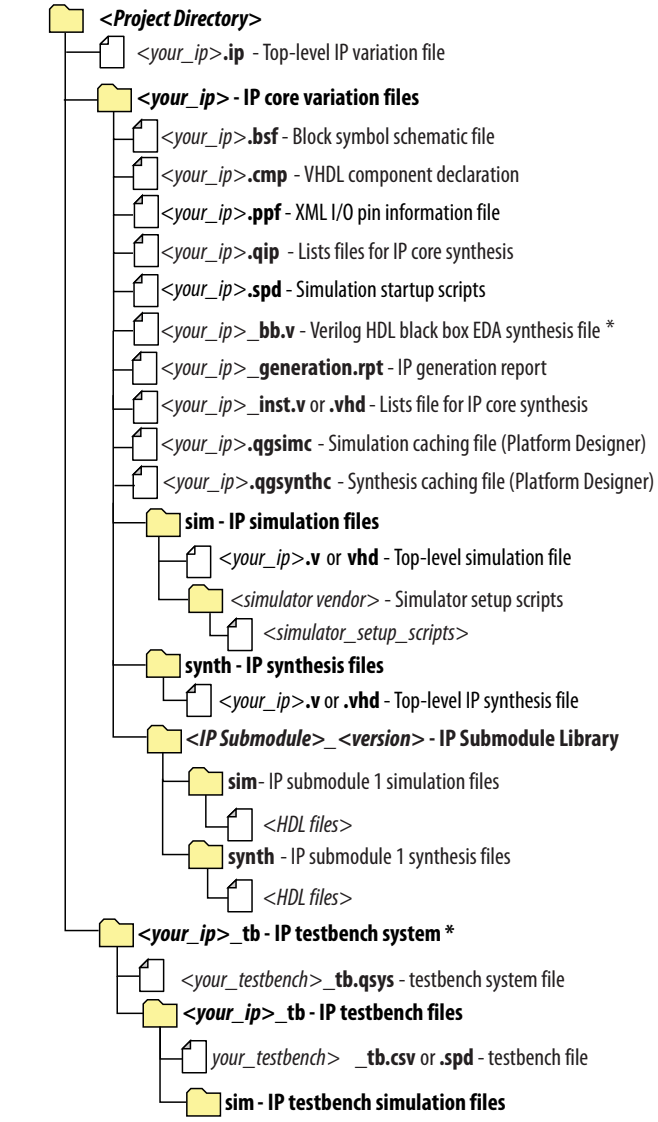
Figure 3. IP Parameter Editor (Intel Quartus Prime Standard Edition)



1.5.2 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 4. Individual IP Core Generation Output (Intel Quartus Prime Pro Edition)



* If supported and enabled for your IP core variation.

Table 3. 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 ModelSim 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*>/	Platform Designer generates /synth and /sim sub-directories for each IP submodule directory that Platform Designer generates.

1.6 Instantiating the Intel FPGA 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 Platform Designer (Standard) 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 Platform Designer (Standard) and the IP Catalog. Optionally, you can instantiate them directly into your RTL design, using Verilog HDL, SystemVerilog, or VHDL.

1.6.1 About 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 5. Example Platform Designer (Standard) System Including the Fault Injection IP Core and EMR Unloader IP Core

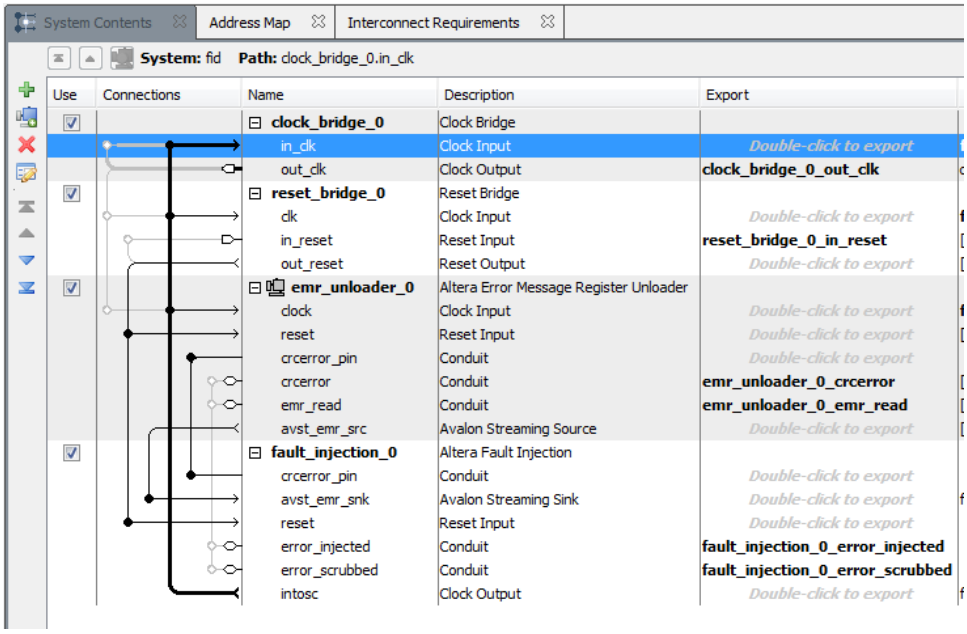
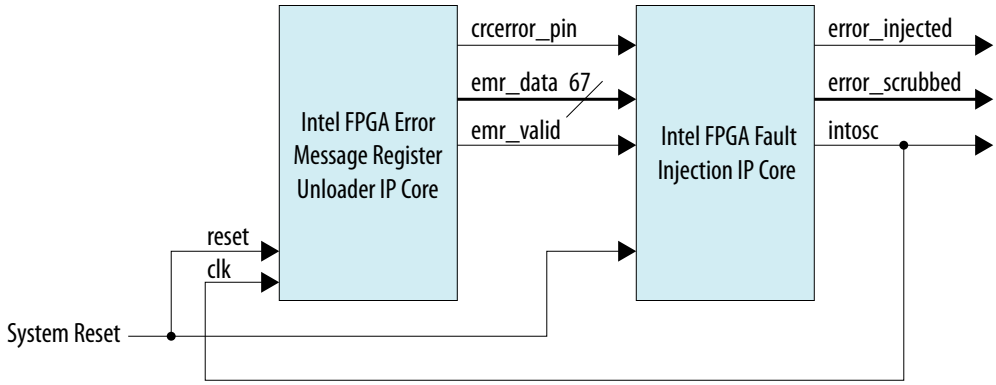


Figure 6. Example Intel FPGA Fault Injection IP Core and EMR Unloader IP Core Block Diagram

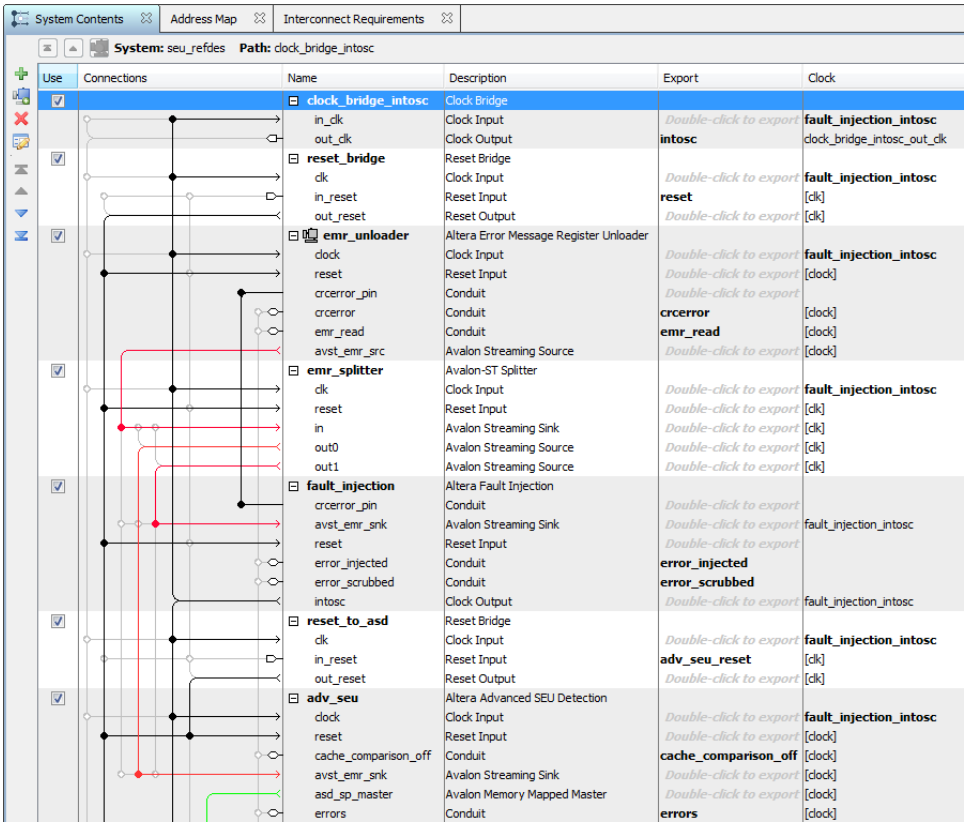


1.6.2 About the Advanced SEU Detection IP Core

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 Platform Designer (Standard) system in which an Avalon-ST splitter distributes the EMR contents to the ASD and Fault Injection IP cores.

Figure 7. Using the ASD and Fault Injection IP in the Same Platform Designer (Standard) System



1.7 Functional Description

With the Intel FPGA 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.

1.7.1 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.

Intel Arria 10, Intel Cyclone 10 GX, 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 Intel FPGA devices, refer to the *SEU Mitigation* chapter in the respective device handbook.

1.7.2 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 Intel Quartus Prime software or from the command line to simulate soft errors.

- 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.

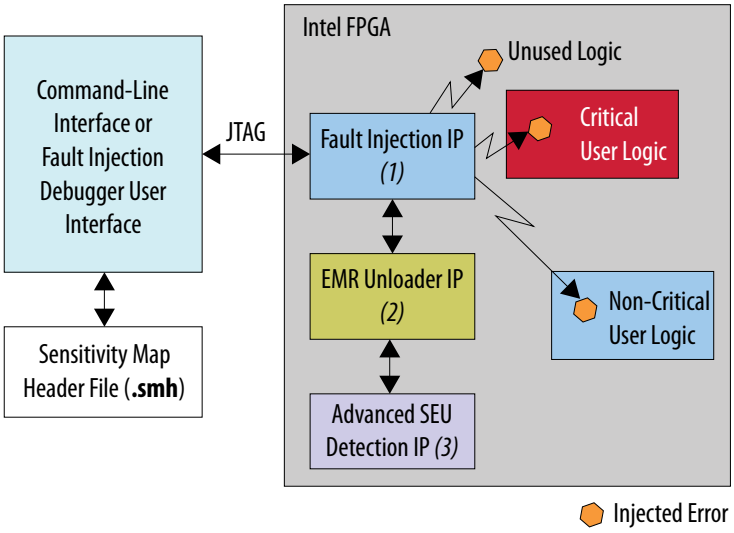
The Fault Injection Debugger communicates with the Fault Injection IP core via the JTAG interface. The Fault Injection IP accepts commands from the JTAG interface and reports status back through the JTAG interface.

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 Intel FPGA 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 8. 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.

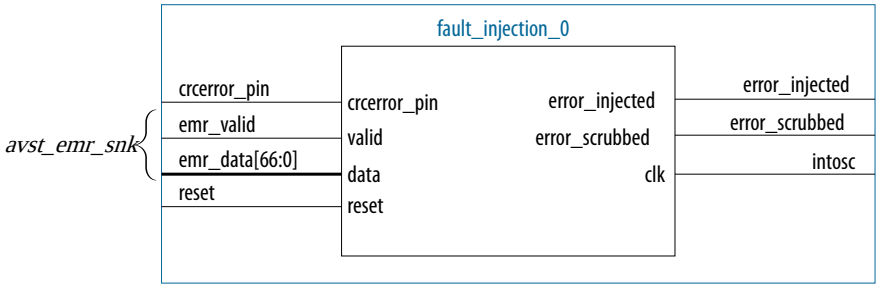
1.7.3 Intel FPGA Fault Injection IP Pin Description

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

Table 4. Intel FPGA Fault Injection IP Core I/O Pins

Pin Name	Pin Direction	Pin Description
`crcerror_pin`	input	Input from Intel FPGA Error Message Register Unloader IP. This signal is asserted when a CRC error has been detected by the device's EDCRC.
`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 Intel FPGA Fault Injection IP uses this clock, for example, to clock the EMR_unloader block.

Figure 9. Intel FPGA Fault Injection IP Pin Diagram



1.8 Intel FPGA Fault Injection IP Core User Guide Archives

If a version is not listed, the manual for the previous version applies.

IP Core Version	User Guide
16.1	Altera Fault Injection IP Core User Guide
15.1	Altera Fault Injection IP Core User Guide
15.0	Altera Fault Injection IP Core User Guide

1.9 Document Revision History for Intel FPGA Fault Injection IP Core User Guide

Date	Version	Changes
2017.11.06	17.1	• Rebranded as Intel. • Added Intel Cyclone 10 GX device support.
2016.10.31	16.1	Updated device support.
2015.12.15	15.1	• Changed Quartus II to Quartus Prime software. • Fixed self-referencing related link.
2015.05.04	15.0	Initial release.