

FT-Unshades2: the User Friendly Framework as interface for designer support

Hipólito Guzmán-Miranda

Javier Barrientos

Luis Sanz

Miguel A. Aguirre

Patricio Molina

SEFUW, 3rd edition
15-17th March 2016

The story so far...

- Fault Injection as a concept is good overall, but can be improved
- The real value is in the INFORMATION we can get from the F.I. experiment
- Designers are typically busy with deadlines, campaigns and other issues
- Fault Injection tools should be usable, effective and with a reasonable difficulty curve
- In our experience, many gains on ease of use can be achieved through UI design

User Interface can help with:

- Design preparation
- Basic tool usage
- Basic post processing
- Easing difficulty curve

But it's more difficult to help with

- Many configuration options at the same time
- Advanced tool usage (custom campaigns)
- Custom post processing

The experience with FT-Unshades2

- Design Preparation Flow
- FT-UNSHADES2 and Remote Access
- ASIC mode. Campaign settings.
- FPGA mode. Campaign setting
- Debugging. Detailed analysis
- Postprocessing
- Advanced issues

Design Preparation Flow

Implement a design using the standard design flow considering:

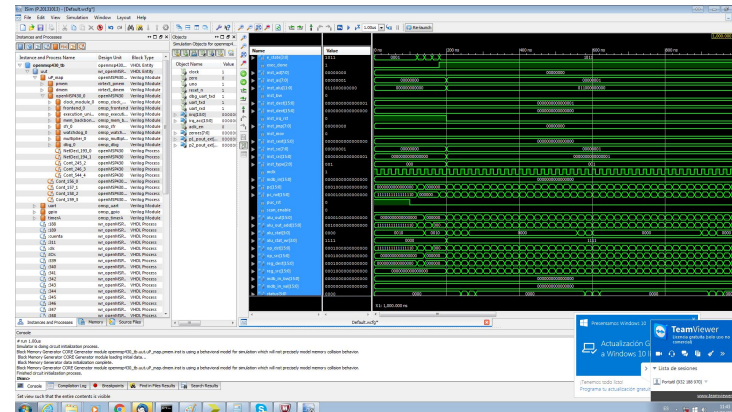
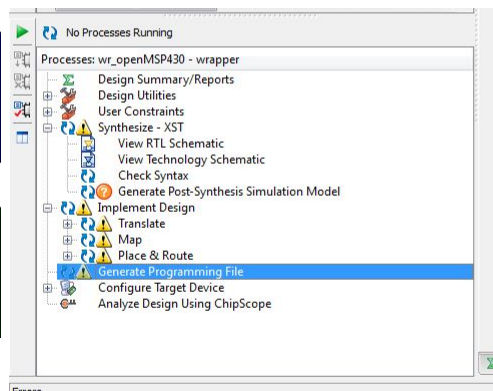
- Pinout fixed by the PCB
- SelectMap port has to remain open
- Bit allocation file (.II) is required
- Avoid DLLs and packing REGISTERS in PADS
- Generate a bitstream (.bit)

Stimuli set is obtained using a standard simulator

- A simulation of the circuit behavior is required
- Standard VCD format is used to obtain this file

Generate
design.pin

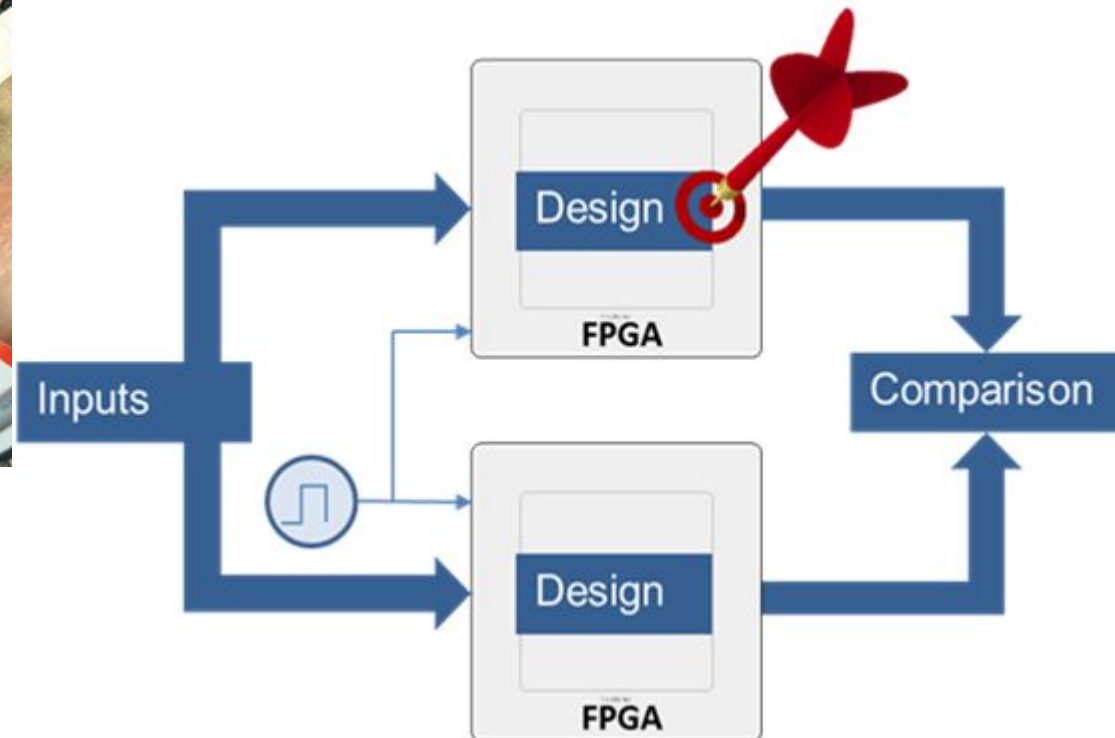
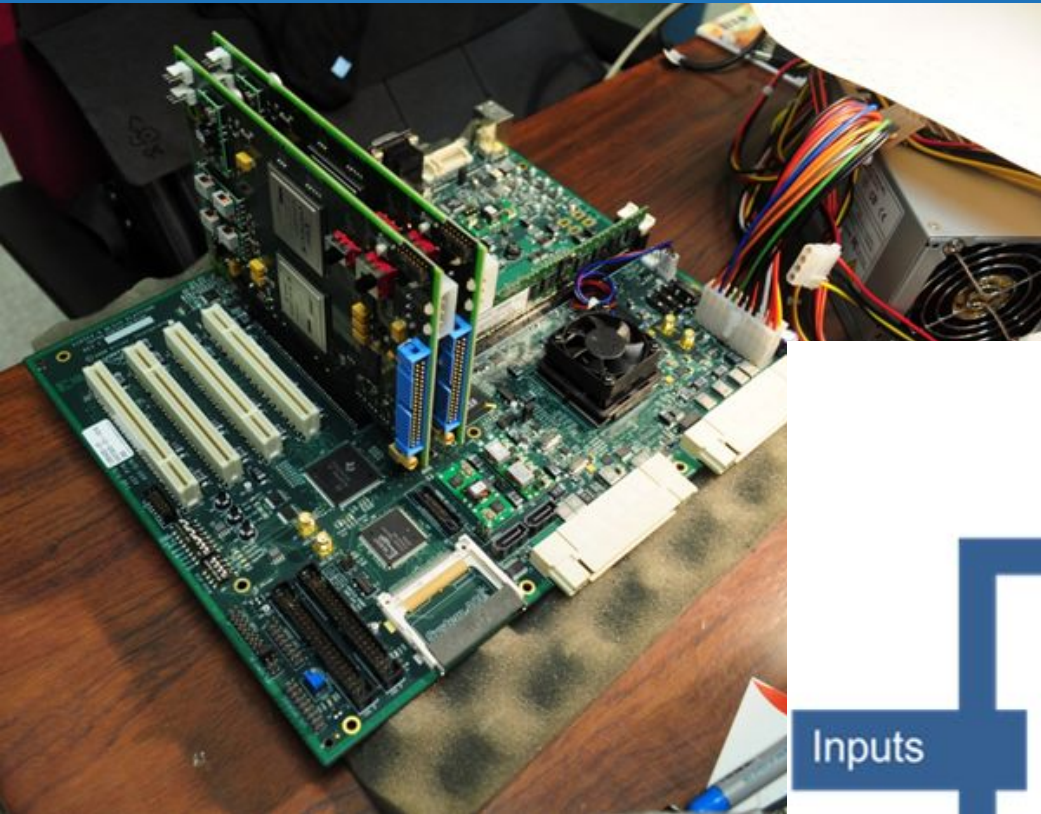
Make
design.ucf



Generate
design.vcd

Make
design.dat

FT-UNSHADES2 system



Remote access

Remote access (web based)

- User can use preferred Operating System (only a web browser is needed)
- User no longer needs to have all the hardware with him
- For maintenance purposes, Software and firmware can be upgraded without any friction to the user


FTUNSHADES
Test Analysis Tools
User Friendly Framework

User name:

Password:

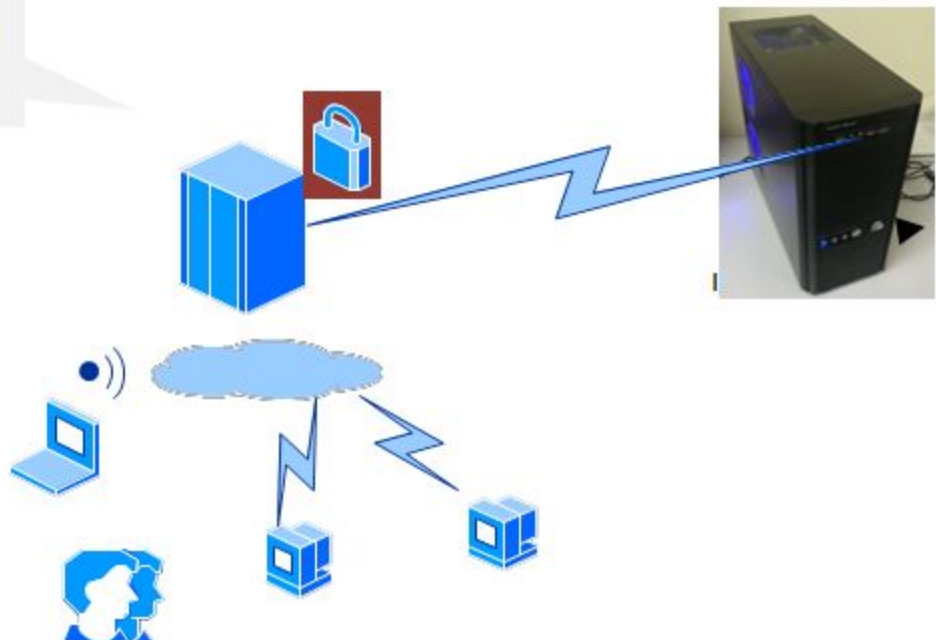
Users may only be created through the [administration page](#).
You need to enable cookies to log in.

Copyright © 2011-2013 Universidad de Sevilla – AICIA – ESA



The FTUNSHADES User Friendly Framework v. 1.0
commit: 7aa0dac602019973ca2bced3f51d5374d03c31
Please provide this ID number with all bug reports

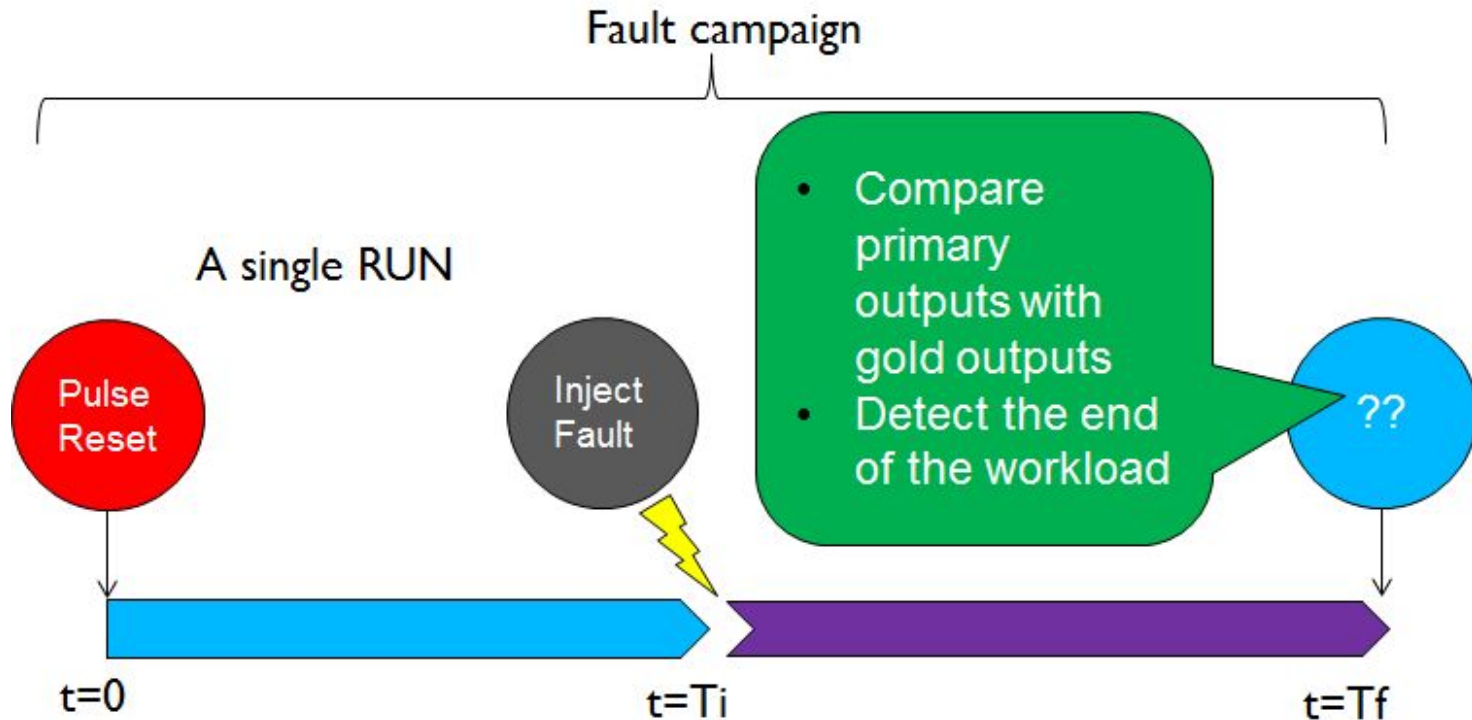
- Bitstream
- Injection Batch Workload



ASIC mode

Fault Injection is performed only targeting the USER REGISTERS.

- After an injection the result is recorded into a Fault Dictionary.
- The procedure is repeated a significant number of RUNS. This is called CAMPAIGN.



Settings for ASIC mode

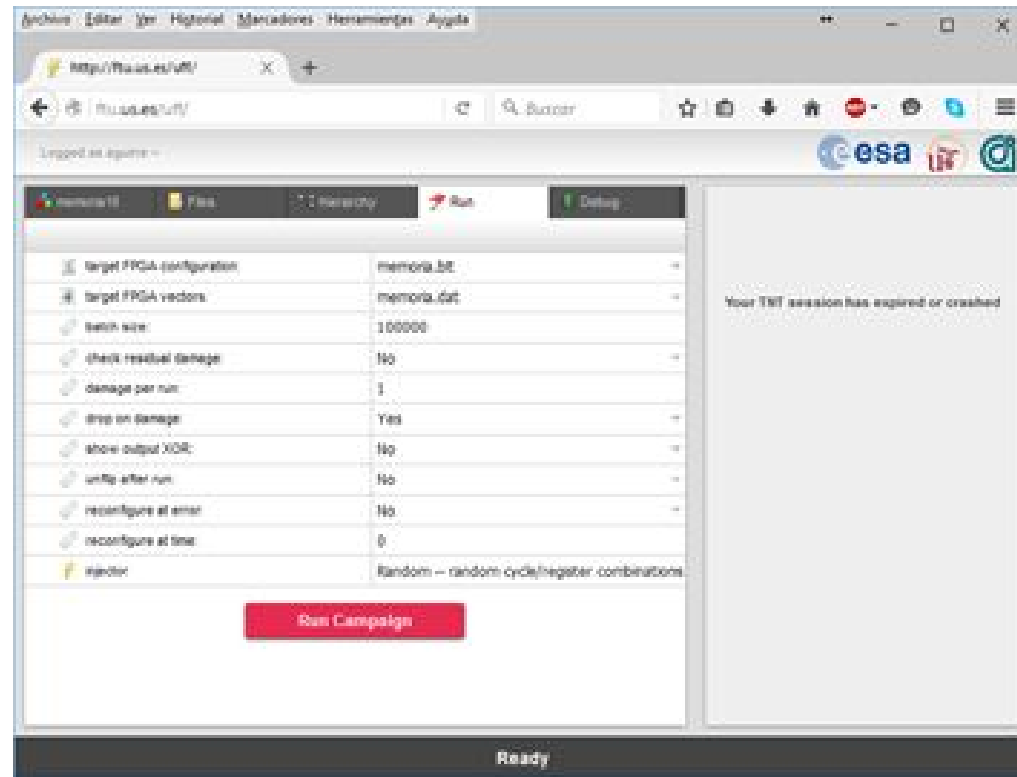
Number of RUNS

Target registers and
clk cycles

Type of campaign

- Random
- Exhaustive
- File
- Custom (tcl)
- Dummy

Number of injections
per run



FPGA mode

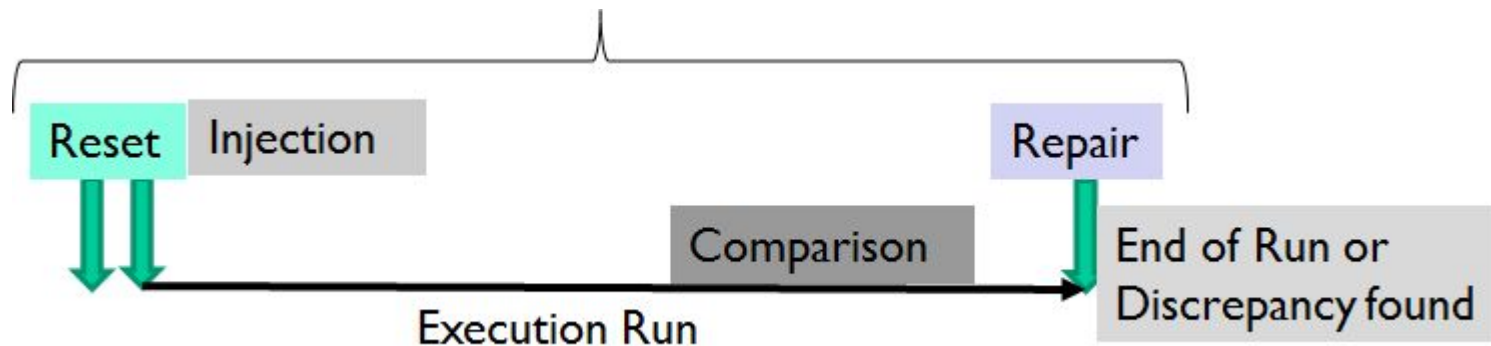
Fault Injection is performed targeting the CONFIGURATION BITS.

- Reuse the FT-UNSHADES2 hardware platform.

*A fault on one configuration bit will not propagate to any other configuration bit
(Is it totally true? Not really!)*

Of course some faults on configuration bit can propagate to the circuit logic

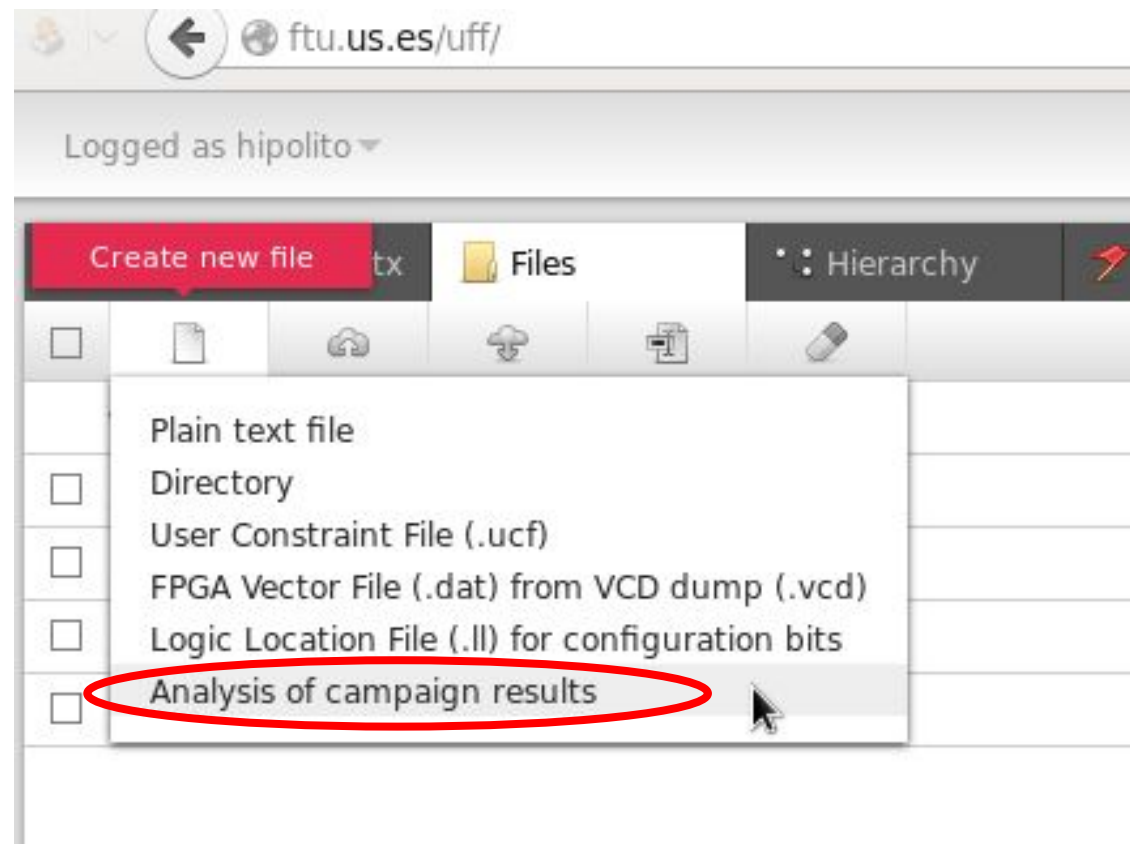
- Make a DYNAMIC injection (not only injections at time zero!)
- Technique: inject and repair (fast scrubbing)



Postprocessing

Multiple modes of operation force the system to log fault information in different files:

- injections.csv
- damages.csv
- reg_name.csv
- stats.txt



For advanced issues

- We strive to make our tool teach itself to the user: basic usage should prepare the designed to begin with advanced issues
 - i.e: Button clicks produce commands which can be reused
- User manuals available online
- Partial reconfiguration (scrubbing) through the user interface in interactive mode & scriptable
- Wanted: FTU2 application notes

Wishlist

- Integrate Implementation tools for FT-Unshades server-based FPGA implementation:
 - Not possible now because legal issues (would need custom software license)
 - Would simplify a lot design preparation process
 - Guarantee designs and bitstreams have compatible FTU2 configurations 100% of the time
- Provide script templates for some typical postprocessing operations
- FTU2 as a tool for radiation tests

Conclusions

A tool for Fault Injection is available, it uses
Xilinx SRAM-FPGAs

The software that interconnects with the
hardware is original and is determined to be an
aid to the designer

The application is linked to the platform and
provides information in ASIC, FPGA and
analysis mode

Let's see an example

Scrubbing emulation

Perform a partial (or even complete!) reconfiguration **maintaining** the current contents of user FFs

This way scrubbing schemas can be tested

Let's see a brief example

Logged as hipolito ▾

counter_8bit_partialreconf

Files

Hierarchy

Run

Debug



0 1 2 3 4 5 6 7 8 9 10

 /data_out

00 01 02 03 04 05 06

00 01 02 03 04 05 06

 /enable /reg

00 01 02 03 04 05 06

00 01 02 03 04 05 06

 /rst_high

Reconfigure target
FPGAs with one of
the bitstreams on
board

tialreconf
 Files
 Hierarchy
 Run
 Debug

	0	1	2	3	4	5	6	7	8	9	10
<input type="checkbox"/> counter_8bit_up.bit											
<input type="checkbox"/> up2down.bit		00			01	02	03	04	05	06	
<input type="checkbox"/> down2up.bit		00			01	02	03	04	05	06	
<input type="checkbox"/> /enable											
<input type="checkbox"/> /reg		00			01	02	03	04	05	06	
		00			01	02	03	04	05	06	
<input type="checkbox"/> /rst_high											

Logged as hipolito

counter_8bit_pa Step cycles Hierarchy Run Debug

1 cycle
5 cycles
10 cycles
other...
until next event

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
/data_out			00			01	02	03	04	05	06	05	04	03	02	01
/enable																
/reg			00			01	02	03	04	05	06	05	04	03	02	01
/rst_high																

Reconfigure target
FPGAs with one of
the bitstreams on
board

tialreconf
 Files
 Hierarchy
 Run
 Debug

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
<input type="checkbox"/> counter_8bit_up.bit																
<input type="checkbox"/> up2down.bit		00	01	02	03	04	05	06	05	04	03	02	01			
<input type="checkbox"/> down2up.bit		00	01	02	03	04	05	06	05	04	03	02	01			
<input type="checkbox"/> /enable																
<input type="checkbox"/> /reg		00	01	02	03	04	05	06	05	04	03	02	01			
<input type="checkbox"/> /rst_high																

Logged as hipolito

counter_8bit_pa Step cycles Hierarchy Run Debug

1 cycle
5 cycles
10 cycles
other...
until next event

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
/data_out		00	01	02	03	04	05	06	05	04	03	02	01	02	03	04	05	06			
/enable																					
/reg		00	01	02	03	04	05	06	05	04	03	02	01	02	03	04	05	06			
/rst_high																					

Conclusions

A tool for Fault Injection is available, it uses
Xilinx SRAM-FPGAs

A Brave FPGA (NanoXplore NXT) version is
intended for the future

The application is linked to the platform and
provides information in ASIC, FPGA and
analysis mode

**We need your feedback to make our tool
better ;)**