Role of Fault Injection in Digital Design for Space

Hipólito Guzmán Miranda
Miguel A. Aguirre
David Merodio Codinachs
Agustin Fernandez-Leon

SEFUW
16-18th September 2014

Outline

1. Integrating Fault Injection in the Digital Design Flow: Benefits and Howto

2. A Fault Injection Technique oriented to SRAM FPGAs

Integrating Fault Injection in the Digital *Design* Flow: Benefits and Howto

SEFUW, 16-18th September 2014

Fault Injection, in theory:

- Big number (total sensitivity)
- Detect most sensitive regions
- Hierarchical analysis
- Verification of inserted protections
- Detect collapsed TMRs
- Defects in reset strategy
- Quality of workloads
- Fault diagnosis
- …

Fault Injection, in practice:

What's the problem?

With fault injection you can "identify if your circuit output is sensitive to SEUs in a given Flip-flop, for a given workload"

Duh!... that is probably why I have a Flip-flop there in the first place!!

If I'm actually using the Flip-flop, the output will be sensitive to SEUs there

So, what do we do?

Think **ENVIRONMENT**, not just CIRCUIT

- All circuits are at least a bit sensitive to radiation effects
- Probably you can live with some SEUs once in a while, but some others will break your circuit (SEFI)
- What is expected from your circuit? What can (and cannot) be tolerated?

Can we answer those two questions in a systematic way?

How to do it?

Divide your circuit FFs into:

- Configuration
  - baudrate, IO standards, … , configuration registers in general
- Control-flow
  - FSMs, data/cycles counters, …
- Datapath
  - the real data you are processing

1. Configuration FFs

- Most likely to cause SEFI
- Should be first choice of FFs to protect
  - TMR/EDAC/...
- These should be clear from the functional specs!
  - Configuration options and modes
- For each of these, it's interesting to ask:
  - do we really need this configurability?
  - do we really need to store it on a Flip-flop?
- Less configurability -> less SEFIs
- You can (and should!) Inject Faults there
  - But expect wrong behaviour

2. Control-flow FFs

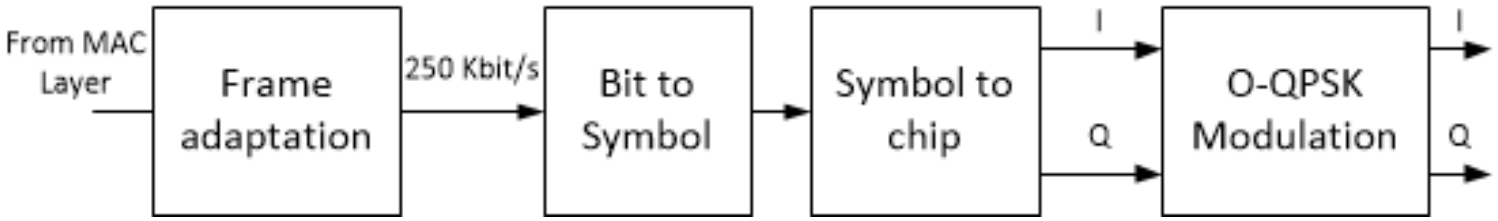
- Faults here may break your circuit in different ways
  - Wrong data at output
  - Wrong timing of output
  - Hang / SEFI
- ‘`when others =>`’ controversial but helpful
  - You must still know what you are doing
- How will you reset the FSMs?
  - Inadequate reset strategy will be seen in the fault injection campaign (unresettable SEU -> SEFI)
  - Global reset easy (but sensitive to SETs)

3. Datapath FFs

- Least critical
- Faults in processed data can sometimes be corrected at higher level
  - EDAC/ECC/Hamming/…
- Cycle-by-cycle comparison is too demanding
- Cook your own comparison model!
  - You will need a flexible Fault Injection tool that gives you more information than damage/no-damage
  - Ideally, total faulty output sequence (or diff with golden)

What we did

- Design an 802.15.4 (Zigbee) baseband transmitter
  - For intra-satellite communications
  - Avoid inserting any protections in the sections where SEUs did not matter

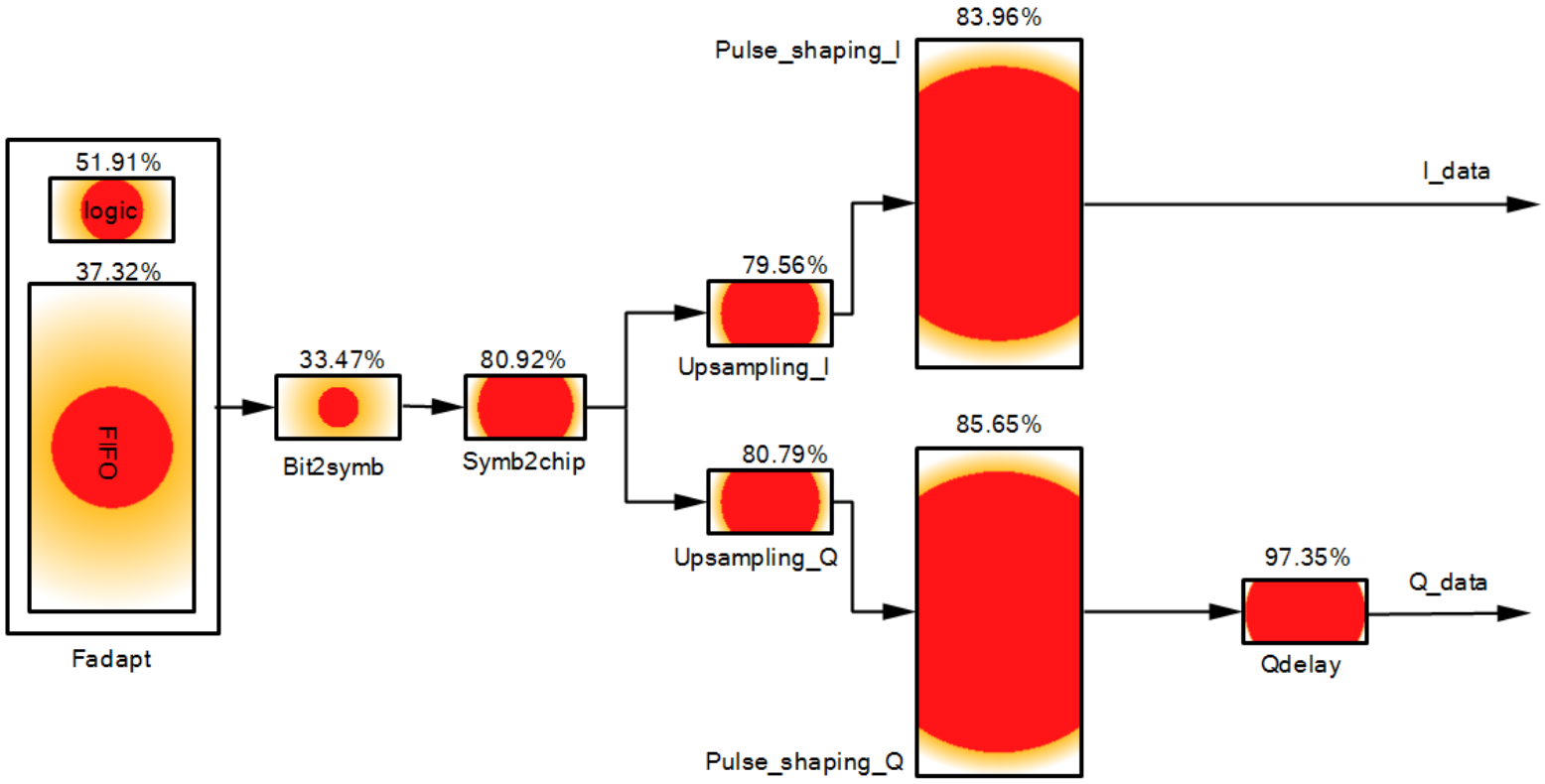


How we did it

- Reduced configurability
  - i.e: FRAME_SIZE became a GENERIC
- Consistent use of '`when others =>`' in FSMs
- Cooked our own comparison model to know if outputs were acceptable
  - Dump all differences for all cycles in a ~8.9GB log
  - Python postprocessing
  - Detect if data was aligned to expected (13 clk cycles per sample)
  - Considered unaligned data unrecoverable
  - Aligned erroneous data was put through a Matlab model of the receiver

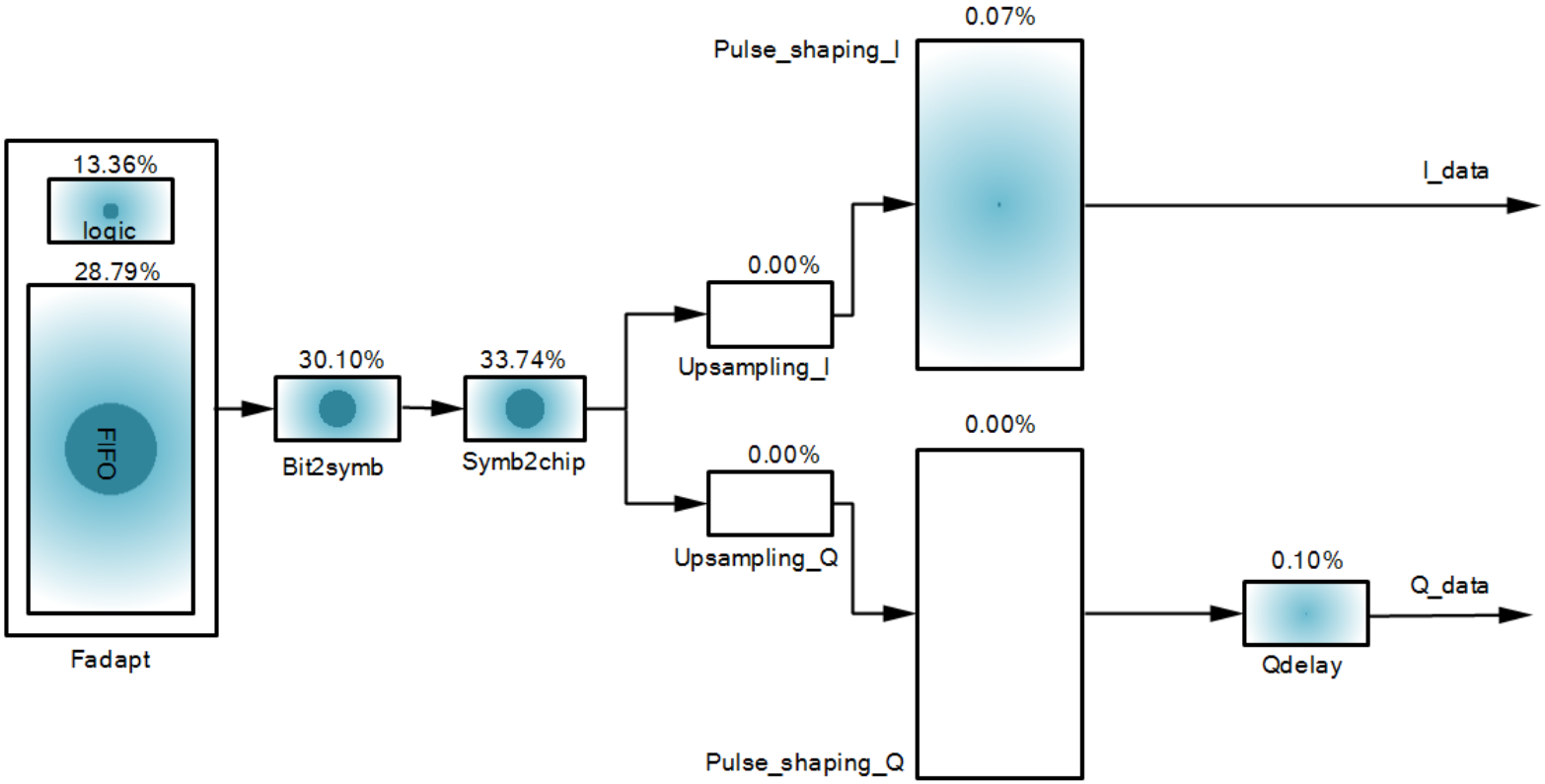
The Results

While cycle-by-cycle comparison suggests system is very sensitive...



The Results (think ENVIRONMENT!)

… the receiver can recover correct data from “wrong” transmitted frames:



Question time

Now it’s time to talk :)

Fault Injection was performed with FT-Unshades2 and yes, you can use it too :)

hipolito@gie.esi.us.es

http://ftu.us.es

A Fault injection technique oriented to SRAM-FPGAs.

Hipólito Guzmán-Miranda, Javier Barrientos-Rojas, Miguel A. Aguirre

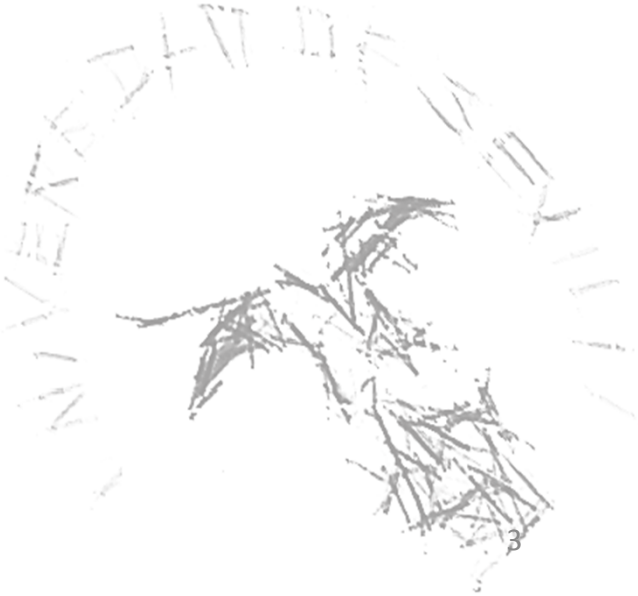
Universidad de Sevilla
Dpto. Ingeniería Electrónica. Escuela Superior e Ingeniería
C/Camino de los Descubrimientos s/n
41092 Sevilla

Motivation

- SRAM-FPGAs are very attractive solution for space applications
- Radiation environments affects them in different way than other electronic circuits, even when they implement common digital circuits.
- We offer TOOLS towards the emulation of the behavior of the SRAM-FPGA and the behavior of the circuit in fault in CBs.
- Check if scrubbing strategies are correct and enough
- The number of configuration points is relevant
- Functional fault assessment and configuration fault assessment are strongly related.

Summary

I. Introduction
II. Fault injection in SRAM-FPGA
III. The FT-UNSHADES2 approach
IV. Experience, conclusions and future work

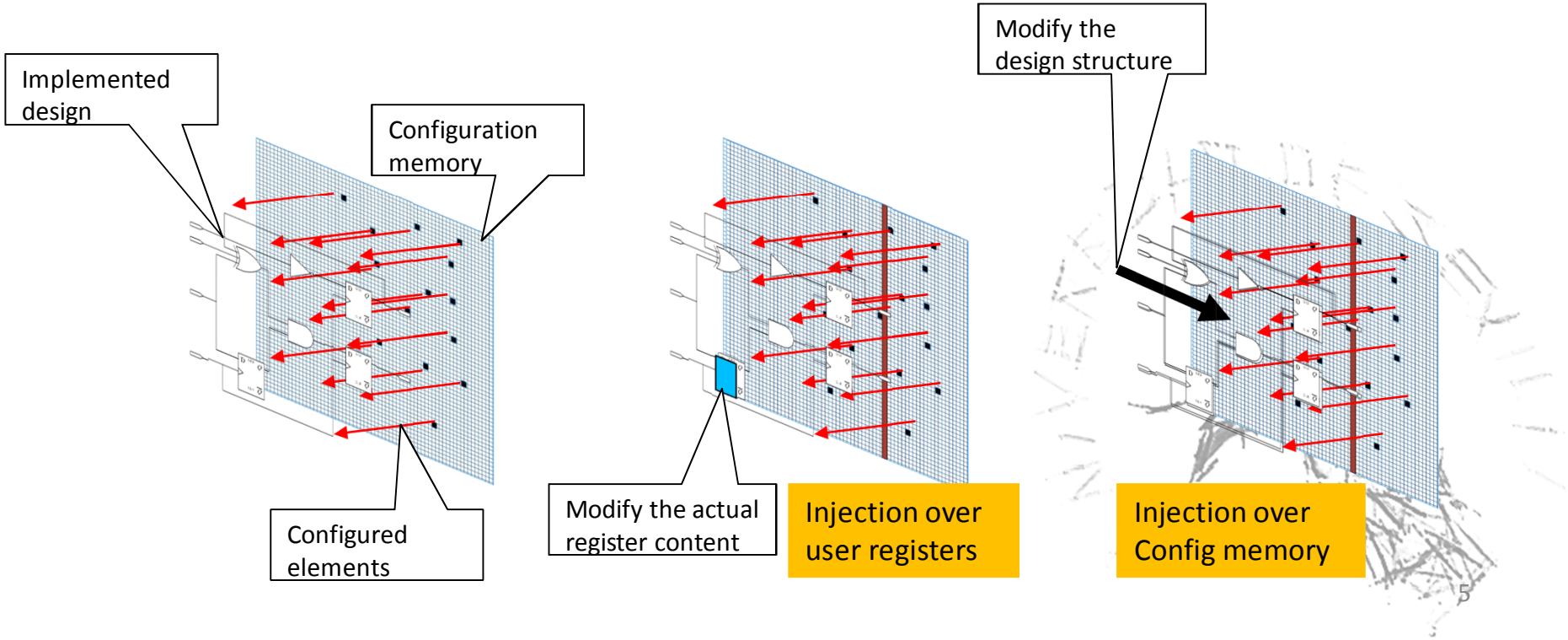


I. Introduction

- SRAM FPGA is an attractive solution for aerospace electronic systems.
- Fault injection techniques represent a good solution for emulating the radiation environment.
- The functional approach consists of analyzing the reliability making injections over USER REGISTERS:
  - Dynamic testing
  - Technology independent
- The present approach targets the FPGA configured with the functional design:
  - Target the FPGA technology
  - Inject over the CONFIGURATION BITS
  - Dynamic testing?

Using Run-Time reconfiguration for FAULT INJECTION

- Faults are propagated through **the logic** to PRIMARY OUTPUTS
- The injection is performed using the configuration circuit
- The injection is made modifying the THE CONFIGURATION BITS instead of REGISTER CONTENTS.



II. Fault injection over SRAM-FPGAs

- Target technology is the SRAM-FPGA
- Injection over the CONFIGURATION BITS
- Objective:
  - Check if faults propagate to the logic
  - Check if redundancy and scrubbing protections are enough
  - Check for simultaneous affection of several clock domains
  - Define PAR rules for design reliability improvement
- Few approaches: FLIPPER (Politecnico di Milano), STAR/RoRa (Politecnico di Torino), BYU injector (from test board and using Xilinx procedures)

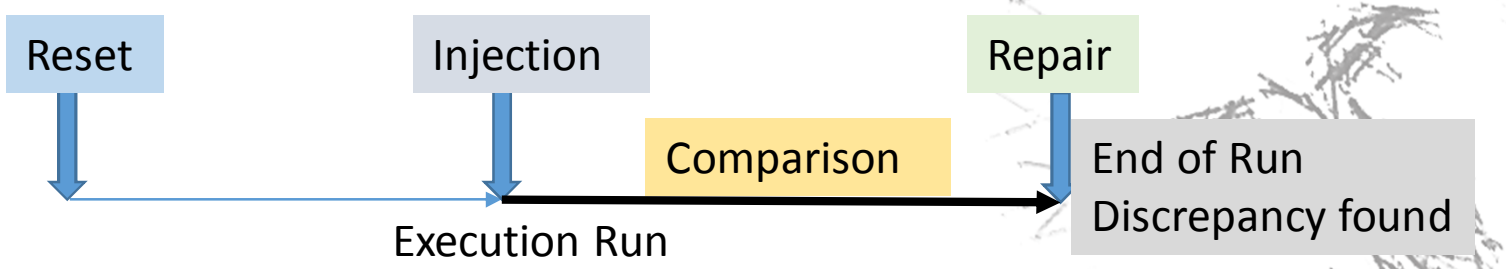
III. The FT-UNSHADES2 approach

- Reuse the FT-UNSHADES2 hardware platform
- The principles:

A fault on one configuration bit will not propagate to any other confguration bit (Is it totally true?)

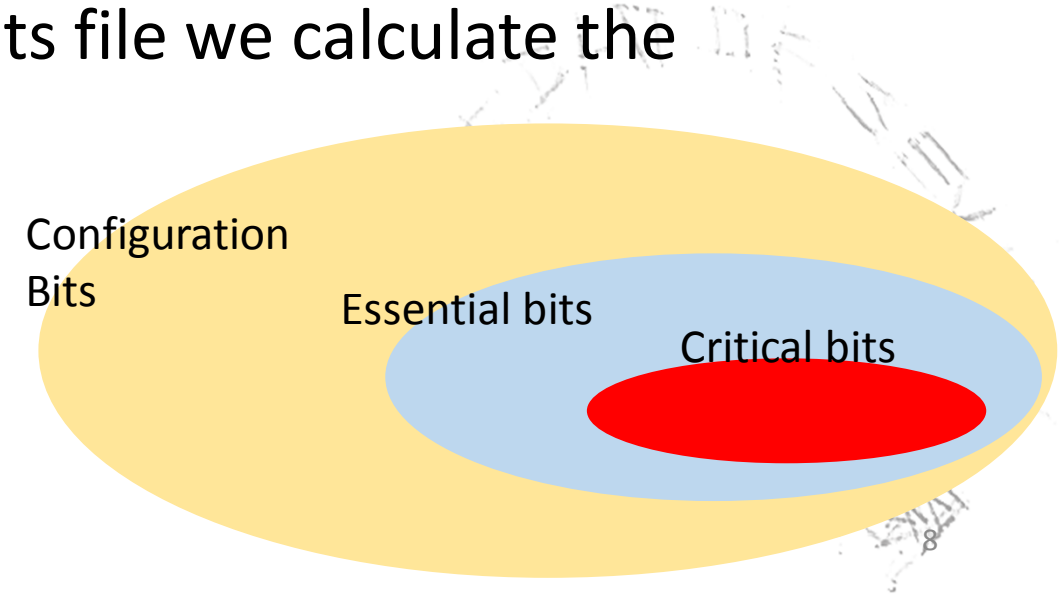
A fault in a configuration bit can propagate to the circuit logic

- Make the DYNAMICAL injection
- Technique: inject and repair



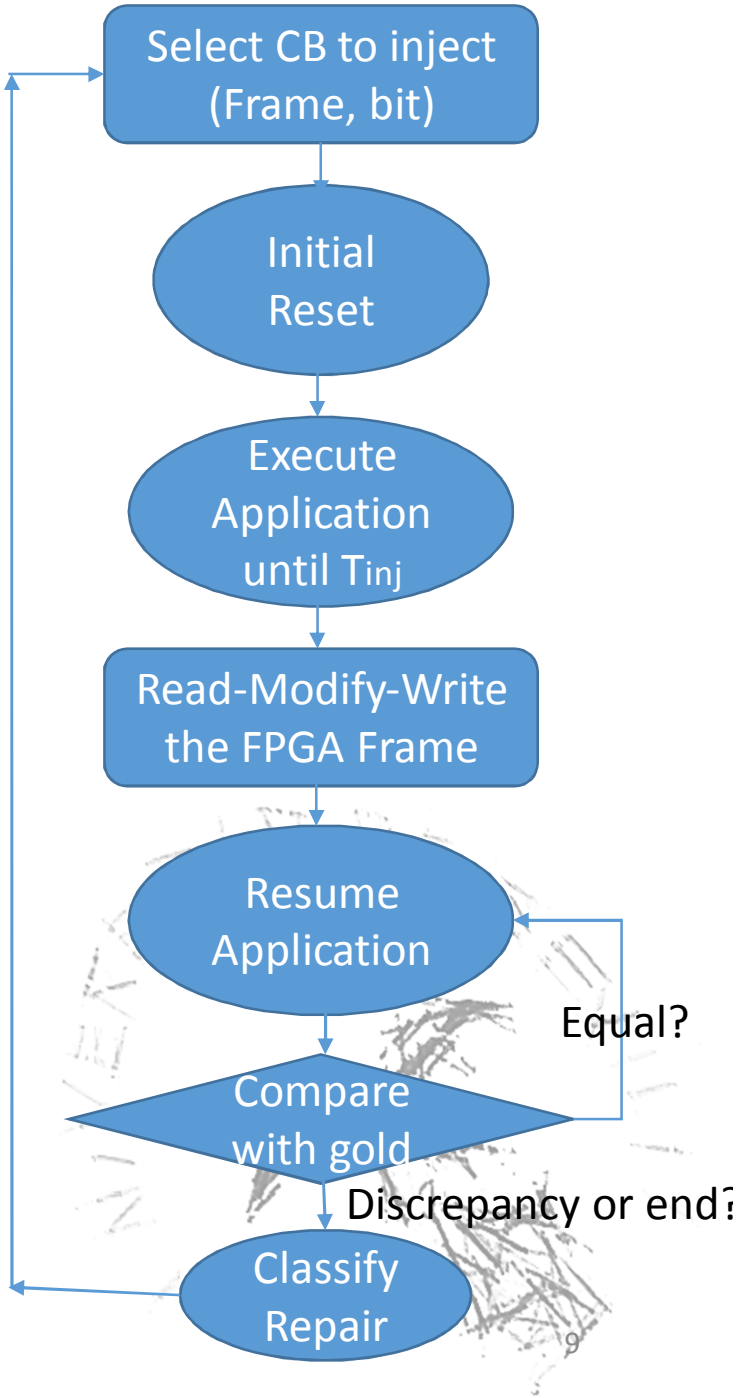
Essential bits

- *Essential bits are defined here as those bits associated with the circuitry of the design, and are a subset of the device configuration bits.*
- *If an essential bit is upset, it changes the design circuitry. However, the upset might not affect the function of the design.*
- They are calculated from BitGen utility (v 13.4)
- From the essential bits file we calculate the sentitive bits map



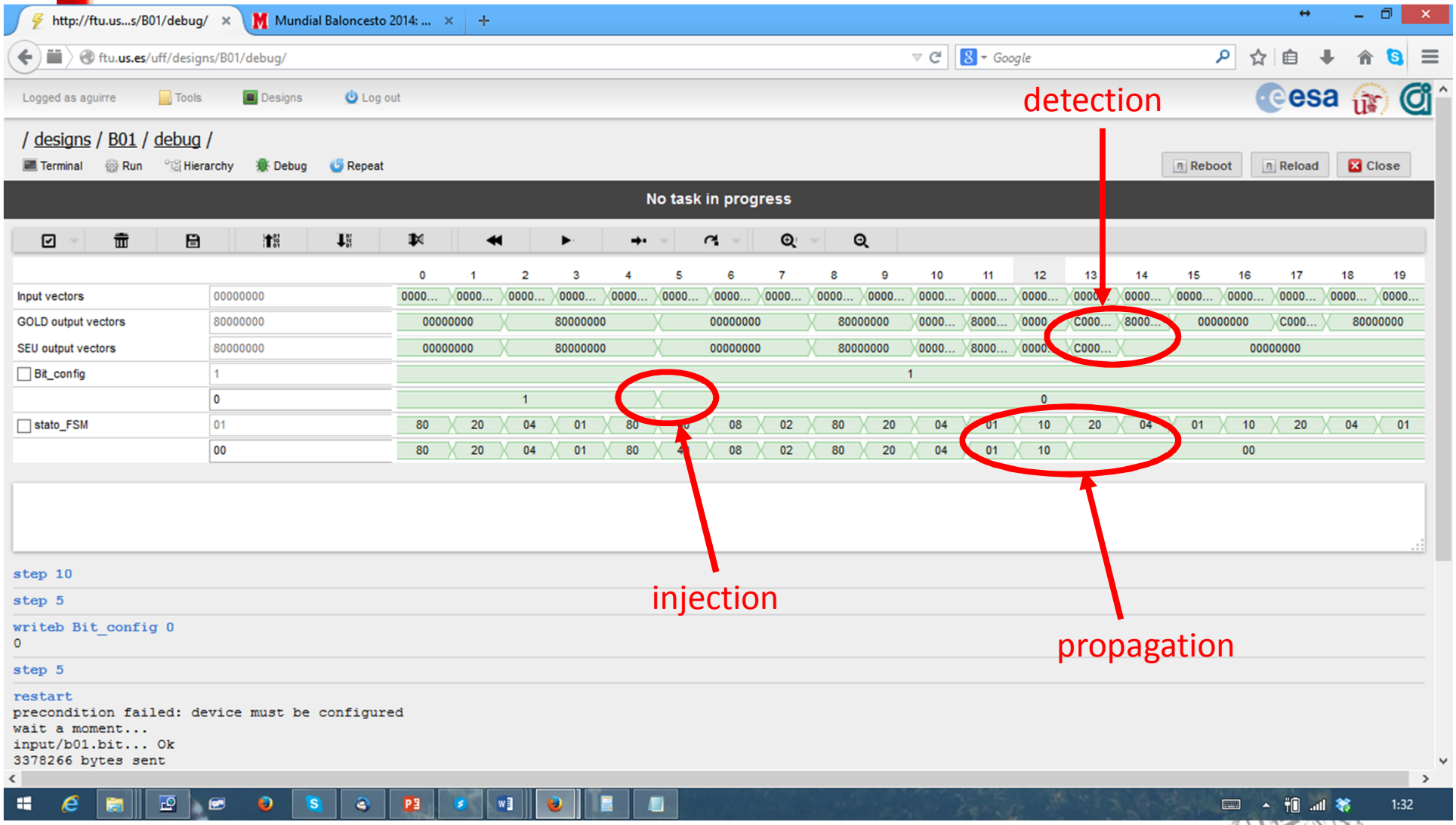
FT-UNSHADES2 FPGA mode

- Same injection technique than ASIC mode
- Detects critical and non-critical bits
- Analysis of the internal propagation of the faults to user logic
- Fast injection
- Dynamic or time zero analysis
- VIRTEX 5



FT-UNSHADES2 in debug mode

(ITC'99, b01)



V. Conclusions

- There are faults that do not affect the circuit
- There are faults that do not affect the logic behaviour
- Faults in CBs can be propagated to user logic
  - Data have to be repaired using traditional techniques (TMR, EDAC,…)
  - If one fault in CB propagates to other CBs:
    - Global scrubbing is required
    - Global reset is required

Can we compute the cross section of these faults?

Thank you. Q & A

aguirre@gie.esi.us.es

hipolito@gie.esi.us.es

New users are welcome to FT-Unshades2 !!

http://ftu.us.es