

Project Number: FP7-611404

**D5.5 – Preliminary EDA tool-suite (draft)**

**Authors<sup>1</sup>**

A. Savino (POLITO), S. Di Carlo (POLITO), A. Vallero (POLITO), G. Politano (POLITO)

Version 1.0 – 19/03/2016

<b>Lead contractor:</b> Politecnico di Torino
<b>Contact person:</b> Alessandro Savino Control and Computer Engineering Dep. Politecnico di Torino, C.so Duca degli Abruzzi, 24 I-10129 Torino TO Italy E-mail: <a href="mailto:alessandro.savino@polito.it">alessandro.savino@polito.it</a>
<b>Involved Partners<sup>2</sup>:</b> POLITO
<b>Work package:</b> WP5
<b>Affected tasks:</b> T5.6

<b>Nature of deliverable<sup>3</sup></b>	R	P	D	O
<b>Dissemination level<sup>4</sup></b>	PU	PP	RE	CO

<sup>1</sup> Authors listed here only identify persons that contributed to the writing of the document.

<sup>2</sup> List of partners that contributed to the activities described in this deliverable.

<sup>3</sup> R: Report, P: Prototype, D: Demonstrator, O: Other

<sup>4</sup> **PU**: public, **PP**: Restricted to other programme participants (including the commission services), **RE** Restricted to a group specified by the consortium (including the Commission services), **CO** Confidential, only for members of the consortium (Including the Commission services)

## COPYRIGHT

© COPYRIGHT CLERECO Consortium consisting of:

- Politecnico di Torino (Italy) – Short name: POLITO
- National and Kapodistrian University of Athens (Greece) - Short name: UoA
- Centre National de la Recherche Scientifique - Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (France) - Short name: CNRS
- Intel Corporation Iberia S.A. (Spain) - Short name: INTEL
- Thales SA (France) - Short name: THALES
- Yogitech s.p.a. (Italy) - Short name: YOGITECH
- ABB (Norway and Sweden) - Short name: ABB
- Universitat Politècnica de Catalunya: UPC

### CONFIDENTIALITY NOTE

THIS DOCUMENT MAY NOT BE COPIED, REPRODUCED, OR MODIFIED IN WHOLE OR IN PART FOR ANY PURPOSE WITHOUT WRITTEN PERMISSION FROM THE CLERECO CONSORTIUM. IN ADDITION TO SUCH WRITTEN PERMISSION TO COPY, REPRODUCE, OR MODIFY THIS DOCUMENT IN WHOLE OR PART, AN ACKNOWLEDGMENT OF THE AUTHORS OF THE DOCUMENT AND ALL APPLICABLE PORTIONS OF THE COPYRIGHT NOTICE MUST BE CLEARLY REFERENCED

ALL RIGHTS RESERVED.

# INDEX

<b>COPYRIGHT</b> .....	<b>2</b>
<b>INDEX</b> .....	<b>3</b>
<b>Scope of the document</b> .....	<b>4</b>
<b>1. Introduction</b> .....	<b>5</b>
<b>2. RELTech Tools</b> .....	<b>7</b>
<b>2.1. Soft Error Rate Technology Analyzer</b> .....	<b>7</b>
<b>3. RelHW Tools</b> .....	<b>8</b>
<b>3.1. MaFIN - Microarchitecture Level Fault Injector for x86 Intel/AMD CPUs</b> .....	<b>8</b>
<b>3.2. GeFIN - Microarchitecture Level Fault Injector for ARM, Intel and AMD CPUs</b> ...	<b>9</b>
<b>3.3. GuFI - Microarchitecture Level Reliability Evaluation of NVIDIA GPUs</b> .....	<b>10</b>
<b>3.4. SIFI - Microarchitecture Level Reliability Evaluation of AMD Southern Islands</b>	
<b>GPGPUs</b> .....	<b>11</b>
<b>3.5. MASKIt - Soft Error Rate Predictor for Combination Circuits</b> .....	<b>12</b>
<b>3.6. NANDA - A tool for the reliability analysis of NAND Flash based SSDs</b> .....	<b>13</b>
<b>4. RELSw Tools</b> .....	<b>14</b>
<b>4.1. LIFILL - A LLVM-based software fault injector</b> .....	<b>14</b>
<b>4.2. LICFI - A Full features C-Based Fault Injector</b> .....	<b>15</b>
<b>4.3. ALIVE - A LLVM-based Lifetime Variable Analysis</b> .....	<b>16</b>
<b>4.4. BalTA - Bayesian Instruction Trace Analyzer for x86 Software</b> .....	<b>17</b>
<b>5. RELSys Tools</b> .....	<b>18</b>
<b>5.1. SyRA - A full System Reliability Analyzer</b> .....	<b>18</b>
<b>5.1. ReDO - A full System Reliability Design Optimizer</b> .....	<b>19</b>

## Scope of the document

This document is the main outcome of task T5.6 "**Preliminary EDA tool-suite**", elaborated in the Description of Work (DoW) of the CLERECO project under Work Package 5 (WP5).

This documents gives a commercial view of all developed tools updated at the date of submission of this document. This is a preliminary version of the deliverable and a consolidated document will be produced at the end of the project.



# 1. Introduction

The CLERECO EDA tool-suite offers tools, models and technologies that cover the four main design dimensions. Figure 1 provides a high-level view of the set of available tools.

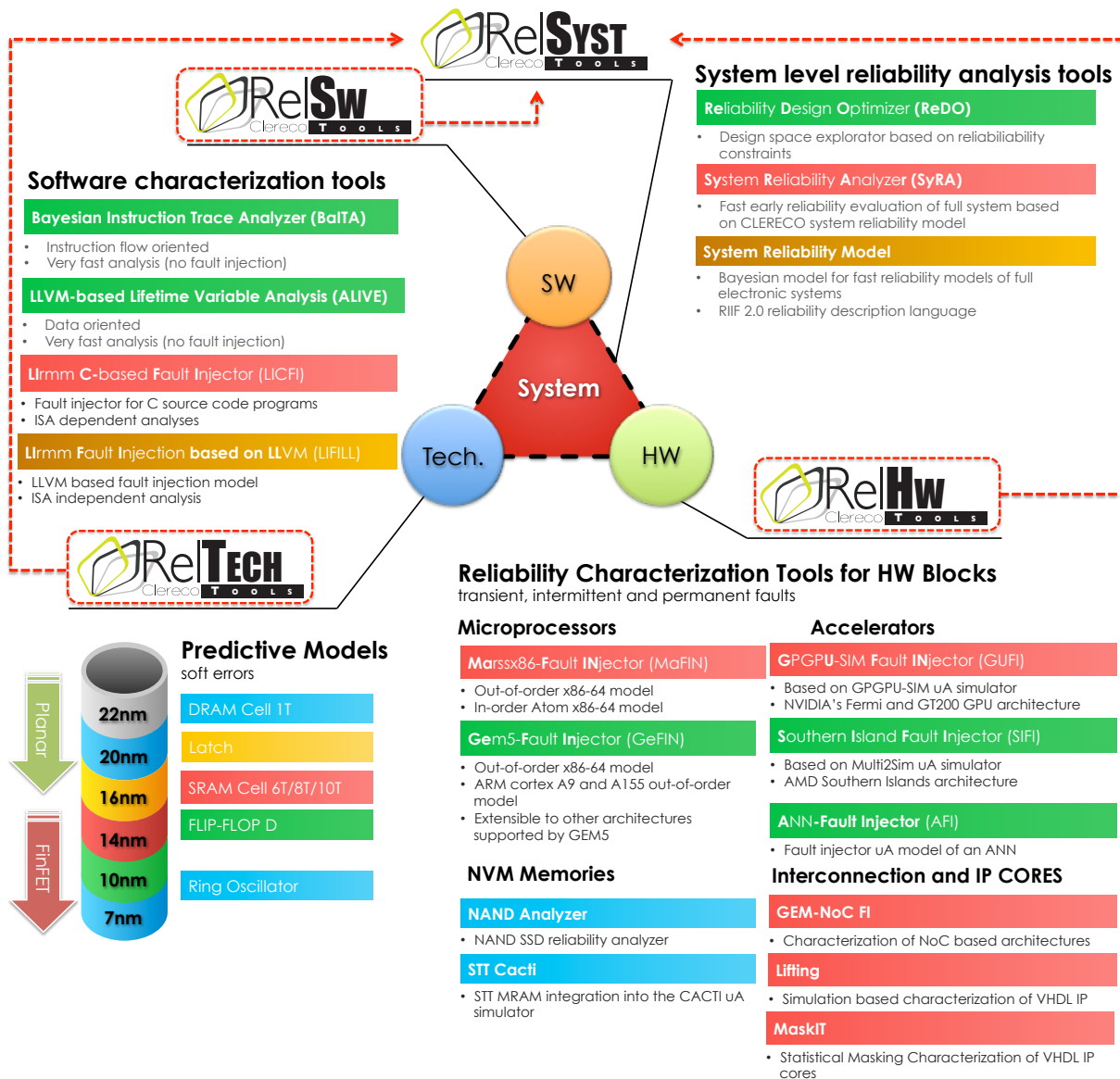


Figure 1: CLERECO EDA tool-suite

---

The tools are organized into four clusters, with each cluster identified by a dedicated logo:

- **RELTech Tools:** offers a set of predictive models to analyze the impact of future technology nodes on specific basic design blocks. Predictive models for technologies are very valuable for OEM and Tier 1 system designers that usually lack access to fab technology data.
- **RelHW Tools:** offers a set of tools for reliability analysis of different hardware architectures. These tools cover the analysis of all major hardware structures of a complex digital system (i.e., Microprocessors, Accelerators, Memories, Interconnections and Custom IP Cores). A very wide set of ICT players including Tier 2 technology providers, up to OEM system's integrators both in the embedded systems and HPC domain are potentially interested in these tools.
- **RelSW Tools:** offers capability to analyze software fault masking in isolation from the hardware architecture. Both static and dynamic analysis of the software is supported by our technology. These tools have a key value for OEM system designers that exploit software fault tolerance solutions to enhance the reliability of their systems.
- **RelSyst Tools:** is the core of the CLERECO design methodology. It integrates information from the other tools into a high-level system model and provides tools to perform early reliability evaluation at system level as well as tools to perform design space exploration in order to optimize the target system given the reliability constraints. This is specifically devoted to OEM system's designers that require the evaluation of the reliability of their products.

This document provides a commercial overview of each developed. The same descriptions are public on the CLERECO website. This preliminary version of the deliverable only includes the description of those tools that reached high-level of maturity. There is a set of minor tools that are still not included. We are currently deciding whether improving them as stand-alone tools or as functionalities of the already completed tools.

## 2. RELTech Tools

### 2.1. Soft Error Rate Technology Analyzer



**Soft Error Rate Technology Analyzer** March 2016

#### Product Overview

SERTA (SER Technology Analyzer) allows for a fast characterization of raw failure rates of current and future technologies for a variety of components such as memories (i.e. SRAMs) and the most common logic gates (i.e. NAND, NOR, NOT). It also provide a sensitivity analysis to operating conditions such as temperature, voltage and location.

*“Technology reliability is not only about the present, it is about the future too”*

- ARCO Research Group (UPC)

#### Supported Architectures

- Any technology based on SPICE description

#### Target Components

- Memories
- Logic Gates

#### Extensions & Tools

- Compliant with the Hazucha and Svensson model
- Fully parametrized analysis.
- Full Technology fair comparison available

#### Supported Fault Models

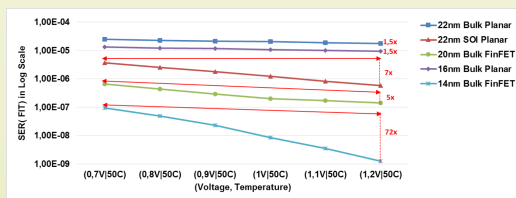
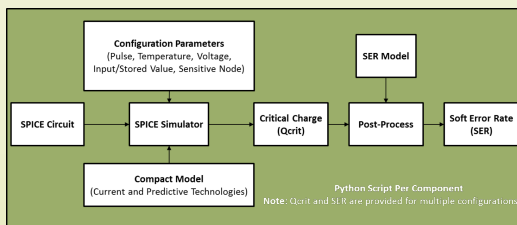
- ✓ **Soft Errors**

#### Measurements

- SER value across several environmental conditions

#### Extra Features

- The tool also allows to perform a fair comparison of these technologies and components using the same methodology to compute their SER.




**Contact Us**  
 Universitat Politècnica de Catalunya, Dep. of Computer Architecture  
 Campus Nord UPC, Cr. Jordi Girona 1-3, 08034 Barcelona (ES)

**Antonio González**  
 Phone: +34-934016988 Fax: +34-934017011  
 E-mail: antonio@ac.upc.edu


### 3. RelHW Tools

#### 3.1. MaFIN - Microarchitecture Level Fault Injector for x86 Intel/AMD CPUs


MaFIN



Clereco  
Cross-Layer Early Reliability Evaluation for the Computing Continuum



FP7-CLERECO  
Grant Agreement FP7-611404



**Microarchitecture Level Fault Injector for x86 Intel/AMD CPUs**
March 2016

#### Product overview

MaFIN is a complete microarchitecture level reliability evaluation framework for high performance computing systems. It is based on state-of-the-art statistical fault injection method or ACE analysis and built on MARSSx86 full-system simulator, providing accurate results for the entire CPU and all its components.

#### Supported Architectures

⇒ **Embedded and high performance x86-64 Intel and AMD architectures**

#### Extensions & Tools

- Caches extended with the data field (L1 data, L1 instruction and unified L2 cache)
- Prefetchers added for the first level caches
- Fully automated** tools for:
  - running the golden run
  - fault mask generation
  - fault injection in MARSSx86
  - Faults classification

#### Target Components

- Physical Register File (Int, FP)
- All fields of caches (L1 data and instruction, L2, L3)
- Prefetchers of L1 data, L1 instruction
- Load/Store Queue
- Load/Store Aliasing Table
- Issue Queue
- Branch Prediction Unit, RAS, BTBs

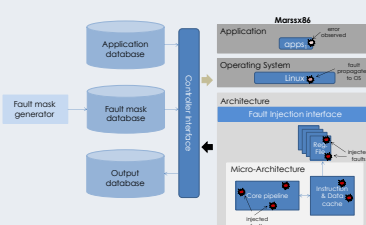
#### Supported Fault Models

⇒ **Transient** } any multiple combination of model, component, entry and cycle  
 ⇒ **Intermittent** }  
 ⇒ **Permanent** }


#### Measurements

- AVF/FIT, HVF
- Fault effect classification:**
  - Masked
  - Silent Data Corruption (SDC)
  - Crash
  - Assert
  - Timeout
  - DUE

*Flexible user extensible parser.*  
*Measurements in **any unmodified workload.***



**Application** (Marssx86) → **Operating System** (Linux) → **Architecture** (Fault Injection interface) → **Micro-Architecture** (Cache pooling, Instruction & Data caches) → **Output database**




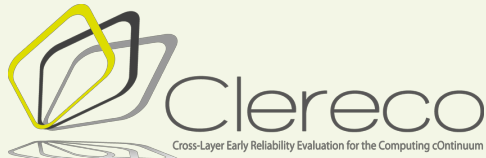
<https://twitter.com/CalDIua>  
**cal@di**

**Contact Us**  
 University of Athens, Department of Informatics and Telecommunications  
 Panepistimiopolis, Ilissia, GR 157 04, Athens, Greece  
 (Office A32, 1st floor, Computer Architecture Lab)


Dimitris Gizopoulos  
 Phone: +30 210 727 5145, Fax: +30 210 727 5214  
 Email: dgizop AT di DOT uoa DOT gr

## 3.2. GeFIN - Microarchitecture Level Fault Injector for ARM, Intel and AMD CPUs






Cross-Layer Early Reliability Evaluation for the Computing cOntinuum



FP7-CLERECO  
Grant Agreement FP7-611404



Microarchitecture Level Fault Injector for ARM, Intel and AMD CPUs
March 2016

### Product overview

GeFIN is a complete microarchitecture level reliability evaluation framework for high performance and embedded computing systems. It is based on state-of-the-art statistical fault injection and built or ACE analysis on Gem5 full-system simulator, providing accurate results for the entire CPU and all its components.

#### Supported Architectures

- ⇒ ARMv7, ARMv8, x86, Alpha
- ⇒ Comes with ARM Cortex-A15, Cortex-A9 and Intel Haswell presets
- ⇒ **Most commercial embedded and high performance** microarchitectures

#### Extensions & Tools

- **Fully automated** interface
  - ◇ Benchmark profiling and checkpointing
  - ◇ Fault-injection campaign
  - ◇ Result classification
- Extension with x86 Translation caches
- Graphical **web interface**
  - ◇ Live status monitoring
  - ◇ Early result classification

#### Target Components

- Physical Register File (Int, FP, CC)
- All fields of caches (L1 data and instruction, L2, L3)
- Prefetchers of L1 data, L1 instruction, L2
- Load/Store Queue (all data fields)
- Instruction Queue (all data fields)
- ROB (active list)
- Rename map
- TLB (Instruction and data)
- Branch Predictors, RAS, BTB
- Main memory

#### Supported Fault Models

- ⇒ **Transient**
- ⇒ **Intermittent**
- ⇒ **Permanent**

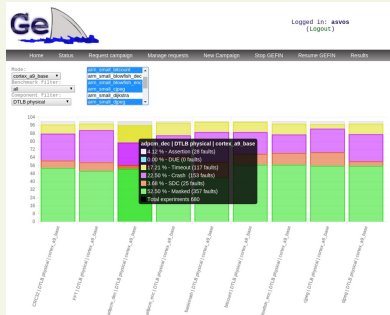
}

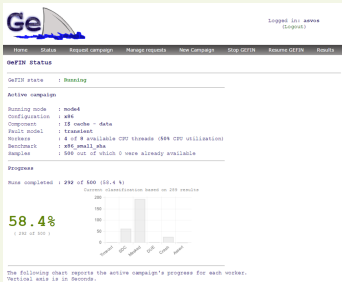
any multiple combination of model, component, entry and cycle


#### Measurements

- AVF/FIT, HVF
- **Fault effect classification:**
  1. Masked
  2. Silent Data Corruption (SDC)
  3. Crash
  4. Assert
  5. Timeout
  6. DUE


*Flexible user extensible parser.  
Measurements in **any unmodified workload**.*





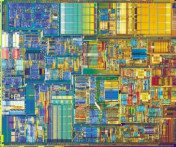


<https://twitter.com/CalDiUoa>



**Contact Us**  
University of Athens, Department of Informatics and Telecommunications  
Panepistimiopolis, Ilissia, GR 157 84, Athens, Greece  
(Office A32, 1st floor, Computer Architecture Lab)

Dimitris Gizopoulos  
Phone: +30 210 727 5145, Fax: +30 210 727 5214  
Email: dgizop AT di DOT uoa DOT gr



*"100 to 1000 times faster microarchitecture level reliability assessments for Intel/AMD x86 and ARM processors"*

- Computer Architecture Lab


### Acceleration with efficient driven simulation

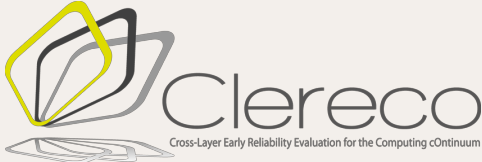
Intelligent **acceleration features:**

- **Workload analysis** - Initial analysis to effectively drive fault injection only to crucial parts - Introduces a novel grouping technique.
- **Simulation speedup** - Runtime simulation speedup with several acceleration techniques.
- Up to **1000x faster** compared to baseline fault-injection.


Results library

### 3.3. GuFI - Microarchitecture Level Reliability Evaluation of NVIDIA GPUs






Clereco  
Cross-Layer Early Reliability Evaluation for the Computing cOntinuum



FP7-CLERECO  
Grant Agreement FP7-611404



SEVENTH FRAMEWORK PROGRAMME

Microarchitecture Level Reliability Evaluation of NVIDIA GPUs
March 2016

#### Product overview

GUFU is a tool for comprehensive reliability assessments of NVIDIA GPU Architectures. It is built on top of a state-of-the-art micro-architectural simulator GPGPU-Sim. It reports the vulnerability of many on-chip hardware components based on **Fault Injection (FI)** or **Architectural Correct Execution (ACE)** analysis.

#### Supported Architectures

- ⇒G80 (Quadro FX 5600)
- ⇒GT200 (Quadro FX 5800)
- ⇒Fermi (GeForce GTX 480, Tesla C2050)

#### Extensions & Tools

**Fully automated** tools for:

- **Fault Injection**
  1. running the golden run
  2. fault mask generation
  3. actual fault injection in GPGPU-Sim
  4. fault classification (Configurable parser according to user needs)
- **ACE Analysis**
- **Both methodologies can be applied to:**
  - **the whole CUDA application**  
comprehensive reliability evaluation of a hardware component for an application
  - **a specific kernel invocation**  
reliability evaluation of a hardware component for a given invocation of a CUDA kernel

#### Target Components

- General Purpose Register file
- Shared Memory
- **Single Instruction Multiple Thread (SIMT) Stacks**
- Valid bit of Instruction buffer entries

#### Supported Fault Models

- Transient
- Intermittent
- Permanent

} any multiple combination of model, component, entry and cycle

#### Measurements

- **Architectural Vulnerability Factor (AVF)**
- **AVF of utilized resources (AVF util)**
- **Failures In Time (FIT)**
- **Mean Instructions to Failure (MITF)**
- **Fault effect classification:**
  1. **Masked**
  2. **Detectable Unrecoverable Error (DUE)**
  3. **Silent Data Corruption (SDC)**

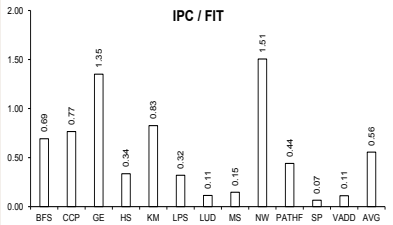
*“Microarchitecture Level Reliability Evaluation of NVIDIA GPU Architectures based on Fault Injection or ACE-Analysis”*

- Computer Architecture Lab


#### Ways to use GUFU

GUFU is a useful tool either for architects (early in the design phase) or programmers:


- Architects may evaluate the reliability of various GPU models.
  - Hardware based protection techniques may be incorporated and also evaluated in terms of performance and reliability.
- Programmers can break the vulnerability of an entire application down to the vulnerability of its kernels.
  - Adding software based error protection only to the most vulnerable kernel of an application can deliver remarkable improvements on its error resilience combined with low loss in performance.

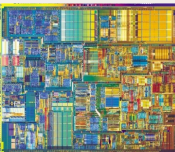


Component	Value
BFS	0.69
CCP	0.77
GE	1.35
HS	0.34
KM	0.83
LPS	0.32
LUD	0.11
MS	0.15
NW	1.51
PATHF	0.44
SP	0.07
VADD	0.11
AVG	0.26



<https://twitter.com/CalDIUoa>





**Contact Us**  
 University of Athens, Department of Informatics and Telecommunications  
 Panepistimiopolis, Ilissia, GR 157 84, Athens, Greece  
 (Office A32, 1st floor, Computer Architecture Lab)

Dimitris Gizopoulos  
 Phone: +30 210 727 5145, Fax: +30 210 727 5214  
 Email: dgizop AT di DOT uoa DOT gr

Version 1.0 – 25/03/2016

### 3.4. SIFI - Microarchitecture Level Reliability Evaluation of AMD Southern Islands GPGPUs



**Microarchitecture Level Reliability Evaluation of AMD Southern Islands GPGPUs March 2016**

#### Product Overview

SIFI is a tool for comprehensive reliability assessments of AMD Southern Islands GPGPU Architectures. It is built on top of a state-of-the art micro-architectural simulator *multi2sim*. It can analyze architectural vulnerability of many on chip hardware components by Fault Injection (FI) and Architectural Correct Execution (ACE) analysis.

*"SIFI is tailored on hardware architects and programmers"*

- Testgroup (Polito)

#### Supported Architectures

- AMD Southern Islands GPGPU Architectures

#### Target Components

- General Purpose Vector Register File
- General Purpose Scalar Register File
- Special Registers of the Scalar Register File
- Local Memory

#### How To Use SIFI

- Fault injection campaign can run in two different modes:
  - by components
  - by internal specific internal resources of each component

#### Extensions & Tools

- Fully automated tools for:
  - Fault injection**
  - ACE Analysis**
- Fully customizable GPGPU architectures for design exploration
- Reliability-related vulnerable code analysis

#### Supported Fault Models

- ✓ **Transient**
- ✓ **Intermittent**
- ✓ **Permanent**

#### Measurements

- Architectural Vulnerability Factor (AVF)
- AVF of utilized resources (AVF Util)
- Failure In Time (FIT)
- Mean Instruction To Failure (MITF)
- Fault effect classification:
  - Masked**
  - Detectable Unrecoverable Error (DUE)**
  - Silent Data Corruption (SDC)**



**POLITECNICO DI TORINO**

#### Contact Us

Politecnico di Turin, Department of Controls and Computer Engineering  
Corso Duca degli Abruzzi 24, 10129, Torino, Italy

**Stefano Di Carlo**

Phone: +39 011 0907080 Fax: +39 011 0907099  
Email: stefano.dicarlo@polito.it

# 3.5. MASKit - Soft Error Rate Predictor for Combination Circuits



**Soft Error Rate Predictor for Combination Circuits** March 2016

### Product Overview

MASKit is a tool that quickly and accurately predicts the Soft Error Rate in combinational circuits. It uses as inputs a netlist of the circuit and the signal probability distribution of its primary inputs to compute the circuit's vulnerability: the probability that a particle strike at any node of the circuit results in a bit flip in one or more primary outputs.

*"MASKit development has been made with a great tool flow in mind"*

*- ARCO Research Group (UPC)*

#### Supported Architectures

- Any combinational circuits described in the format produced by popular synthesis tools, such as **RTL Compiler** and **Yosys**

#### Target Components

- All** gates in the model
  - Gate models are automatically extracted from the technology library

#### Extensions & Tools

- Precise reliability estimation avoiding RTL fault-injection campaigns.
  - Speedup from 170x to 800x
- Estimation accounts Technology node, Supply voltage and Temperature

#### Supported Fault Models

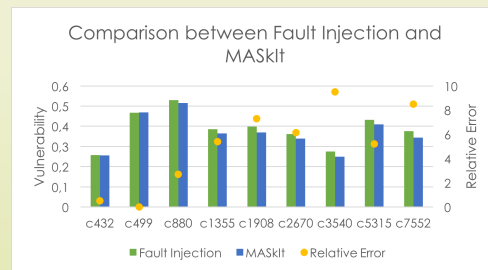
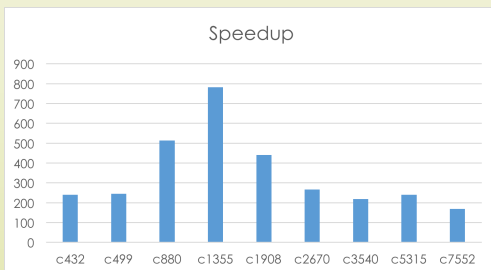
- ✓ **Transient**

#### Measurements

- Vulnerability factor** for each node of the circuit

#### Analysis Enhancements

MASKit can be connected to any architecture-level simulator tool, providing models of micro architectural components otherwise totally missing



**Contact Us**  
 Universitat Politècnica de Catalunya, Dep. of Computer Architecture  
 Campus Nord UPC, Cr. Jordi Girona 1-3, 08034 Barcelona (ES)

**Marif Anglada**  
 Phone: +34-934016988  
 Email: manglada@ac.upc.edu



# 3.6. NANDA - A tool for the reliability analysis of NAND Flash based SSDs.



**A tool for the reliability analysis of NAND Flash based SSDs.** March 2016

### Product Overview

NAND Analyzer is a product for design analysis of NAND Flash based Solid State Drives (SSDs). In includes models to assess:

- Flash memory error rate prediction based on the workload
- Wear-out analysis
- ECC scheme analysis.

*“Evaluate **error-rate** and **lifetime** of your SSD storage system “*

- Testgroup (Polito)

### Extensions & Tools

- Workload based characterization
- Different ISPP programming algorithms
- Different ECC configurations
- **Support for YAFFS2 file system**

### Target Components

- SLC NAND Flash Memories
- MLC NAND Flash Memories

### Supported Fault Models

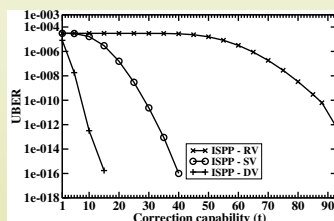
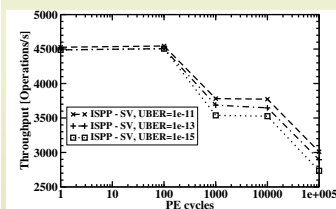
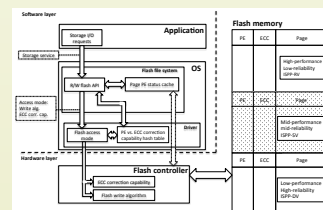
- ✓ **Intermittent**
- ✓ **Permanent**

### Measurements

- Bit Error Rates
- Timing
- Power consumption
- Full statistics report

### How To Use NAND Analyzer

- Configure your SSD characteristics
- Explore different design dimensions



**POLITECNICO DI TORINO**

**Contact Us**  
 Politecnico di Torino, Department of Controls and Computer Engineering  
 Corso Duca degli Abruzzi 24, 10129, Torino, Italy

**Stefano Di Carlo**  
 Phone: +39 011 0907080 Fax: +39 011 0907099  
 Email: stefano.dicarlo@polito.it

# 4. RELSw Tools

## 4.1. LIFILL - A LLVM-based software fault injector



**A LLVM-based software fault injector** March 2016

### Product Overview

LIFILL (Lirmm Fault Injection LLVM-based) is able to inject faults in both data and instructions of the LLVM code. The LLVM source code is modified by applying mutations that implement the effect of the fault on the variable or the instructions.

*"We provided you the **passcode** to the **reliability** of any software you develop"*

### Supported Architectures

Any language provided with a LLVM compiler.

### Target Components

- Any data (variables, vectors, etc.)
- Any **standard** LLVM instruction.

- LIRMM (CRNS)

### Extensions & Tools

- Fully Hardware independent
- Controllability on the fault location and its effects.

### Supported Fault Models

CLERECO developed Software Fault Models (SFM):

- ✓ **Wrong Data**
- ✓ **Instruction Replacement**

### System Requirements:

- OS:** Linux
- Tools:** clang/llvm
- RAM:** 4GB

### Measurements

- Masking probability**
- Fault Silent Violation (FSV)**
- Crashed**
- Detected Faults**



**Contact Us**  
 LIRMM - CNRS / Université Montpellier  
 UMR 5506 - CC 477,  
 161 rue Ada, 34095 Montpellier Cedex 5 France  
**Giorgio Di Natale**  
 Phone: +33 467 41 85 01  
 Email: giorgio.dinatale@lirmm.fr

## 4.2. LICFI - A Full features C-Based Fault Injector

# LICFI



### A Full features C-Based Fault Injector March 2016

#### Product Overview

LICFI (Lirrm C-Based Fault Injector) randomly inject faults in both data and instructions of a program written in C language. Injections are randomly and dynamically performed while the program is currently running.

*“The only feasible way to prove your C program is **reliable** is **testing it, quickly**”*

- LIRMM (CRNS)

#### Supported Architectures

The tool supports **all** C language programs.

#### Target Components

- Any data (variables, vectors, etc.)
- Any **standard** C instruction.

#### Extensions & Tools

- Hardware independent.
- Instrumented at the original source code, which offers an efficient observability of the software components.
- Execute on the final executable file.
- Easy fault injection mechanism.
- Multi-Thread implementation.

#### Supported Fault Models

CLERECO developed Software Fault Models (SFM):

- ✓ **Wrong Data**
- ✓ **Instruction Replacement**

#### Measurements

- **Masking probability**
- **Fault Silent Violation (FSV)**
- **Crashed**
- **Detected Faults**

#### Key Concepts

Instrumentation of the original code allows a selective analysis of the code.

#### System Requirements:

- **OS:** Linux
- **Tools:** clang/llvm
- **Libraries:** pthread
- **RAM:** 4GB



**Contact Us**  
 LIRMM - CNRS / Université Montpellier  
 UMR 5506 - CC 477,  
 161 rue Ada, 34095 Montpellier Cedex 5 France

**Giorgio Di Natale**  
 Phone: +33 467 41 85 01  
 Email: giorgio.dinatale@lirmm.fr

## 4.3. ALIVE - A LLVM-based Lifetime Variable Analysis



### A LLVM-based Lifetime Variable Analysis March 2016

#### Product Overview

ALIVE evaluates the effect of faults in all variables of a generic software, by analyzing the variable lifetime and its propagation to the output of the program.

*“Before asking if a single fault will **impact** on your system, ask if it will be **seen at all**”*

#### Supported Architectures

The tool supports **all** programming languages included in the **LLVM** set of compilers.

#### Target Components

- Single variables
- Basic Structures (i.e., vectors and matrix)
- Advanced structures (i.e., unions, multi-type containers)

- LIRMM (CRNS)

#### Extensions & Tools

- Very fast evaluation: only **one** run is required to provide effective results,
- Time accurate,
- Accounts for all possible making effects,
- Support Software Error Protection strategies,

#### Supported Fault Models

CLERECO developed Software Fault Models (SFM):

- ✓ **Wrong Data**
- ✓ **Instruction Replacement**

#### Measurements

- **Masking probability**
- **Fault Silent Violation (FSV)**
- **Crashed**
- **Detected Faults**

#### Key Concepts

Variable **Lifetime analysis**:





- A variable is **alive** from the first write to the last read (before next write)
  - A fault in an alive variable can have influence on the program execution
  - A fault in a dead variable is masked (will be either re-written or never used again)



**Contact Us**  
 LIRMM - CNRS / Université Montpellier  
 UMR 5506 - CC 477,  
 161 rue Ada, 34095 Montpellier Cedex 5 France

**Giorgio Di Natale**  
 Phone: +33 467 41 85 01  
 Email: giorgio.dinatale@lirmm.fr

## 4.4. BalTA - Bayesian Instruction Trace Analyzer for x86 Software

**Bayesian Instruction Trace Analyzer for x86 Software** March 2016

### Product Overview

BalTA is a reliability instruction trace analyzer for softwares based on bayesian network. It provides a very fast analysis of each x86 Instruction Set Architecture (ISA) based software exploring real executable traces of the software without the need of the original sources.

*"The **only** way to prove your running software is really **reliable**"*

- Testgroup (Polito)

### Supported Architectures

The tool is able to parse:

- ✓ x86 standard instructions
- ✓ **AMD** extensions
- ✓ **SSE1** & 2 extensions
- ✓ **MMX** instructions

### Extensions & Tools

- Fully automated analysis
  - Data propagation
  - Control flow generation
- Internal parser fully customizable
- Multi-thread analysis capability
- Reliability model for further investigation provided as output

### Target Components

- System Registers
  - ES, SS, DS, CS, ...
  - EIP, EDI, ...
  - ...
- General Purpose Registers
  - EAX, EBX, ...
  - r1x, r2x, ...
- Floating Point Registers
- MMX registers
- **All** addressable Memory Locations

### Supported Fault Models

- ✓ **Transient**
- ✓ **Intermittent**
- ✓ **Permanent**

### Measurements

- AVF/FIT
- Single target error probability

### Extra Features

- Cross-Platform Implementation
- Easy compilation using CMake
- Fully customizable parser
- Extendible Target component description
- **Compatible with CLERICO MaFIN and GeFIN tools**

### System Requirements

- **OS:** Linux, OS X 10.8 or later
- **Libraries:** SMILE
- **RAM:** 4GB
- **Tools:** CMake, Bison, Flex



**POLITECNICO  
DI TORINO**

#### Contact Us

Politecnico di Torino, Department of Controls and Computer Engineering  
Corso Duca degli Abruzzi 24, 10129, Torino, Italy

**Stefano Di Carlo**

Phone: +39 011 0907080 Fax: +39 011 0907099

Email: stefano.dicarlo@polito.it

## 5. RELSys Tools

### 5.1. SyRA - A full System Reliability Analyzer





FP7-CLERECO  
Grant Agreement FP7-611404



**A full System Reliability Analyzer** March 2016

#### Product Overview

SyRA automates reliability analysis of complex electronic systems by means of component based statistical reliability models. SyRA enables to model the target system in terms of components (technology, hardware and software) and resorting to the CLERECO Bayesian reliability engine can efficiently analyze how faults and errors propagate through components, accounting for complex interactions among them that are not modeled with simpler statistical models.

*“You don't need to know that your system is reliable, you need to **prove it!**”*

*- Testgroup (Polito)*

#### Supported Architectures

- Supported microprocessors architectures through other CLERECO tools (ARM Cortex A9, ARM Cortex A15, x86\_64)
- Single/Multicore architectures
- Single/Multithread applications

#### Target Components

- All hardware components and subcomponents.
- All functions of the OS and the Software.

#### Key Features

- The model is highly parameterized. It enables to include any factor that can potentially affect the reliability of the system (e.g., environmental factors such as location and temperature) by simply adding new variables to the model.
- **Full GUI available**

#### Supported Fault Models

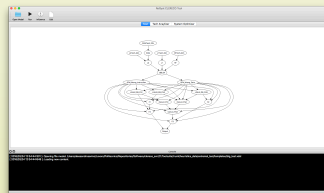
- ✓ **Transient**
- ✓ **Permanent**

#### Extensions & Tools

- Full system stack analyzed (from technology to the application software)
- Detailed hardware and software description
- Montecarlo simulation to account for uncertainty on reliability parameters of the single components
- Very fast analysis for early design exploration.

#### Measurements

- AVF/FIT
- Influence Probability



#### System Requirements

- **OS:** Linux, OS X 10.8 or later
- **Libraries:** SMILE, QT, Boost
- **RAM:** 4GB



**POLITECNICO DI TORINO**

#### Contact Us

Politecnico di Torino, Department of Controls and Computer Engineering  
Corso Duca degli Abruzzi 24, 10129, Torino, Italy

**Stefano Di Carlo**

Phone: +39 011 0907080 Fax: +39 011 0907099  
Email: stefano.dicarlo@polito.it

## 5.1. ReDO - A full System Reliability Design Optimizer

# ReDO



FP7-CLERECO  
Grant Agreement FP7-611404



**A full System Reliability Design Optimizer**

March 2016

### Product Overview

ReDO improves the reliability of your system design by selecting the best combination components (technology, hardware and software) to meet your design constraints. ReDO let you explores hundreds of design alternatives automatically. ReDO features an advanced optimization algorithm inspired by the Extremal Optimization evolutionary strategy, and it is based on the CLERECO Bayesian reliability engine.

*“There is only one way to optimization: be sure you are getting only the **best of all.**”*

- Testgroup (Polito)

### Supported Architectures

- Supported microprocessors architectures through other CLERECO tools (ARM Cortex A9, ARM Cortex A15, x86\_64)
- Single/Multicore architectures
- Single/Multithread applications

### Extensions & Tools

- Full system stack optimized (from technology to the application software)
- Very fast design exploration
- Full Design exploration logged
- Multi-objective optimization functions
- Maximum optimization time definable by the user based on early stop conditions

### Target Components

- Full optimization for:
- All hardware components and subcomponents.
  - All functions of the OS and the Software.

### Optimization Parameters:

- ✓ **Reliability**
- ✓ **Time**
- ✓ **Area**
- ✓ **Power Consumption**
- ✓ ... any parameter that can be described and evaluated.

### Measurements

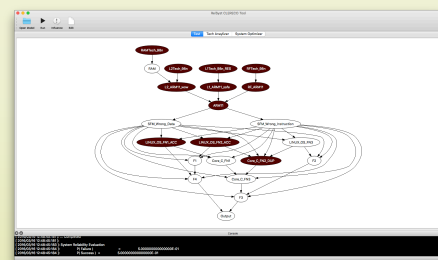
- AVF/FIT
- Percentages of improvement.
- Influence Probability

### Key Features

- Fully design optimization via support of users defining objective functions.
- Full GUI available**

### System Requirements

- OS:** Linux, OS X 10.8 or later
- Libraries:** SMILE, QT, Boost
- RAM:** 4GB



**POLITECNICO DI TORINO**

#### Contact Us

Politecnico of Turin, Department of Controls and Computer Engineering  
Corso Duca degli Abruzzi 24, 10129, Torino, Italy

**Stefano Di Carlo**

Phone: +39 011 0907080 Fax: +39 011 0907099  
Email: stefano.dicarlo@polito.it