POLITECNICO DI TORINO

Dipartimento di Automatica e Informatica

PhD in Computer and Control Engineering

29th cycle

Advisor

Ernesto Sanchez

New techniques for functional test of processor-based systems

PhD Candidate: *Riccardo Cantoro*

Introduction

Electronic devices may be affected by **faults** as an effect of **physical defects**. These defects may be introduced:

- during the manufacturing process
- while the device is operative (due to aging).

Safety-critical applications do not tolerate errors due to faults: this is the reason why **testing** such devices is needed so to guarantee a correct behavior at any time. Testing is performed with different approaches.

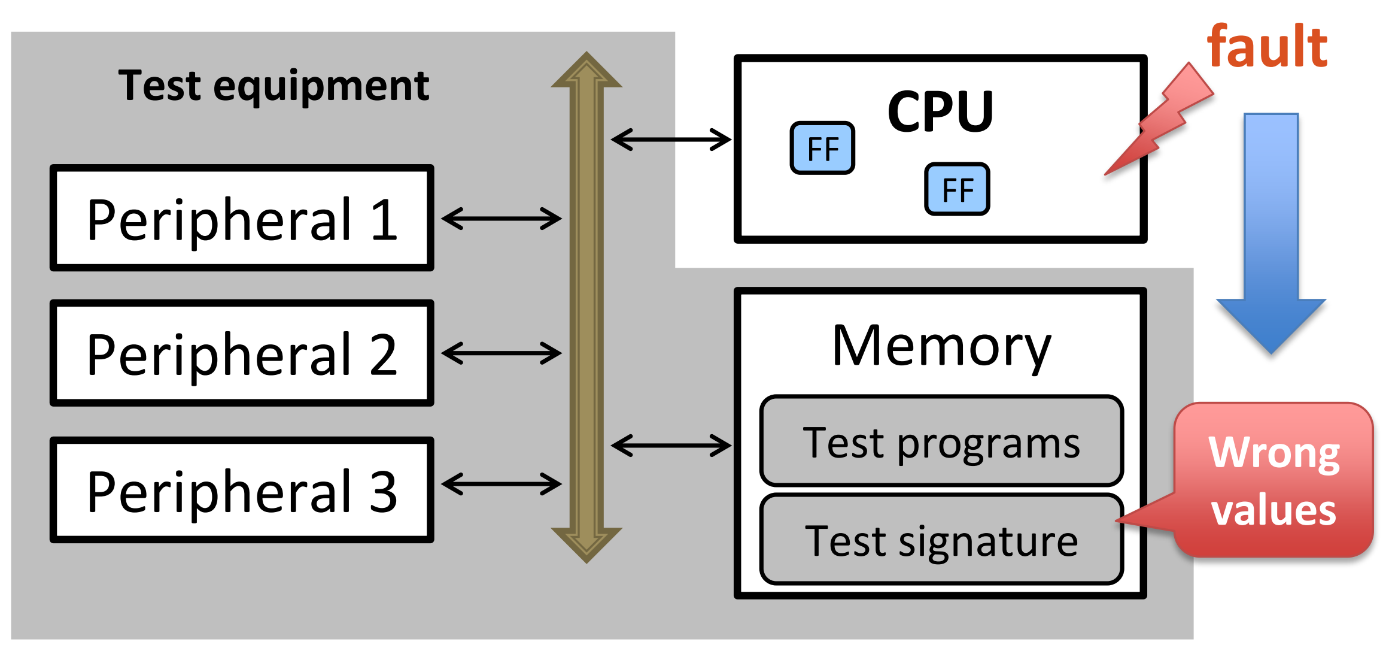
Design for Testability – the original design is equipped with special hardware devoted to testing

- Scan-chains
- Logic/memory built-in self-test (LBIST, MBIST).

Software-Based Self-Test – a suite of test programs is stored in a memory accessible by the processor

- The processor executes the test program
- Results are gathered and compared with the expected ones.

In-field SBST scenario



Research questions

How to improve current SBST fault coverage?

- ✓ **New systematic SBST algorithms**
- ✓ **Investigation of observability issues**

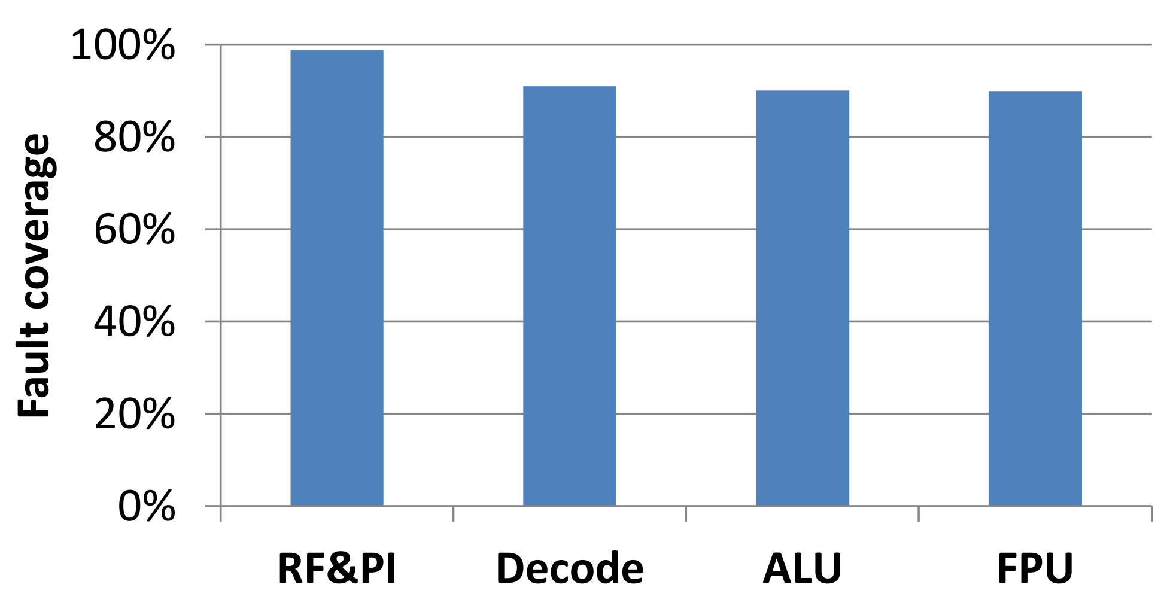
How to deal with expensive SBST development flow?

- ✓ **Automatic test programs generation**
- ✓ **Effective industrial development flow**

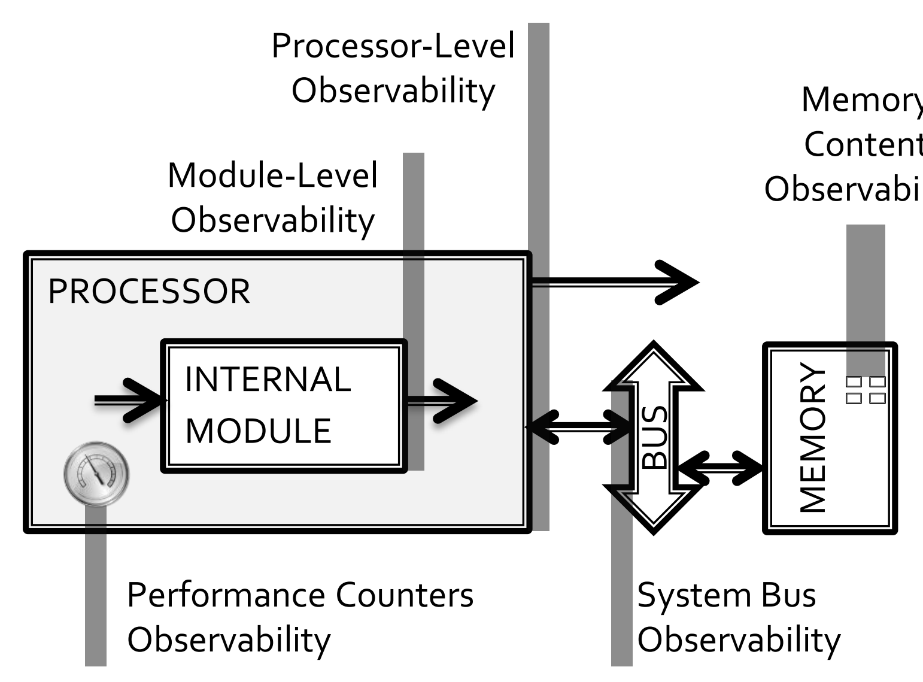
New systematic SBST algorithms

In my research, the fault coverage of several processor sub-modules have been increased by means of systematic SBST algorithms

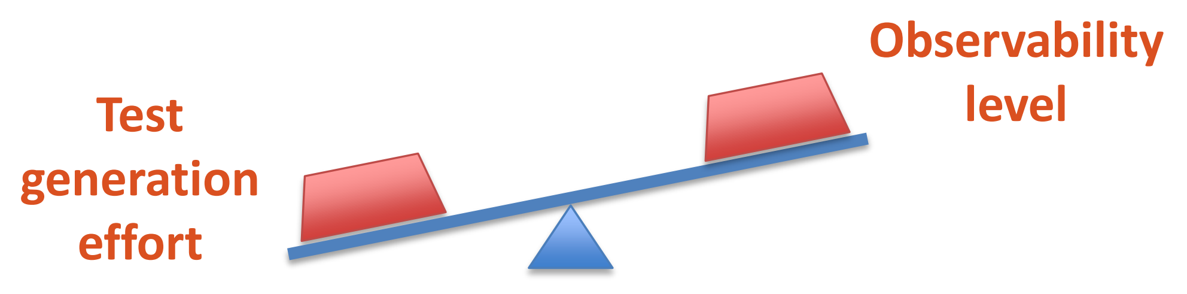
- Register forwarding & pipeline interlock [MTV'13]
- Decode units [DFT'14]
- ALU modules [EST'15]
- Cache coherency logic [LASCAS'15]
- Embedded floating-point units [DFT'16]



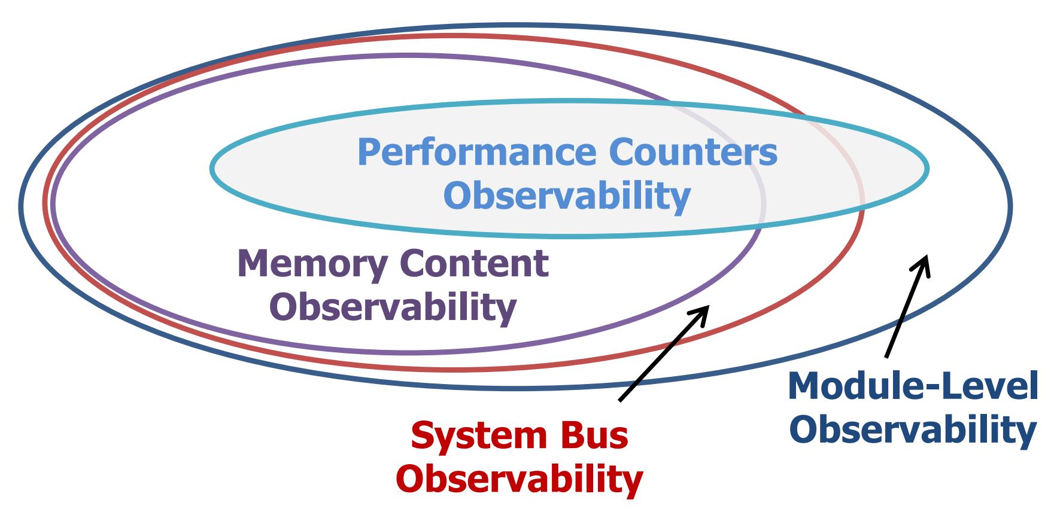
Investigation of observability issues [2]


Processor-Level Observability
Module-Level Observability
Memory Content Observability
PROCESSOR
INTERNAL MODULE
BUS
MEMORY
Performance Counters Observability
System Bus Observability

The **effectiveness** of test programs depends, among other factors, on the mechanisms adopted to observe the behavior of the system.

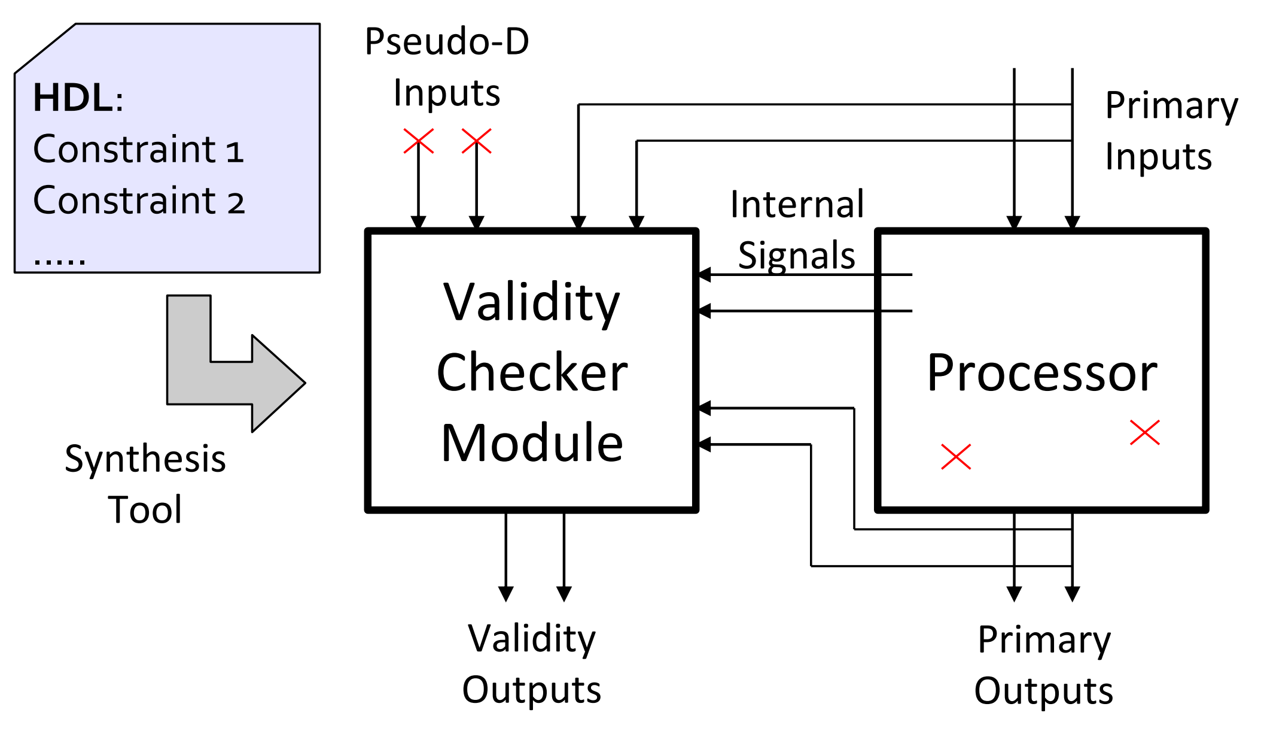


Part of my research focused on the quantitative evaluation of the **drop in fault coverage** coming from the adoption of the alternative approaches.


Performance Counters Observability
Memory Content Observability
System Bus Observability
Module-Level Observability

[2] J. Perez Acle, R. Cantoro, E. Sanchez, M. Sonza Reorda, G. Squillero, "Observability solutions for in-field functional test of processor-based systems: A survey and quantitative test case evaluation," *Microprocessors and Microsystems*, Volume 47, Part B, November 2016, Pages 392-403.

Automatic test programs generation [1]



The test environment is represented with formal constraints written in hardware-description language (HDL). A satisfiability (**SAT**) solver is used to generate test programs able to cover all detectable faults and to respect the constraints.

Results

- The framework is able to prove that certain faults are **untestable** by means of functional programs.
- The testable fault coverage is superior to manual generation techniques (up to 98% coverage on a MIPS-like processor).

[1] A. Riefert, R. Cantoro, M. Sauer, M. Sonza Reorda and B. Becker, "A Flexible Framework for the Automatic Generation of SBST Programs," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 10, pp. 3055-3066, Oct. 2016.

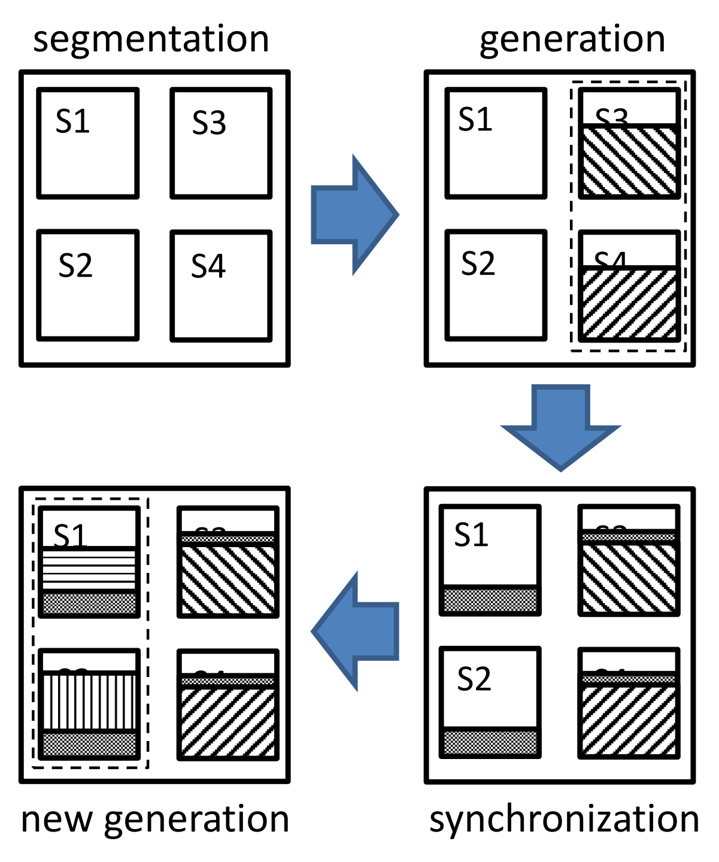
Effective industrial development flow [3]

Industrial processors are typically too large to test as a unique module. In my research, a general SBST development flow has been implemented. The overall flow is based on the following principles:

Modularity – the fault list is split in **sub-modules** by taking into account structural and functional aspects.

Parallelization – test programs are developed in parallel on **orthogonal** fault lists.

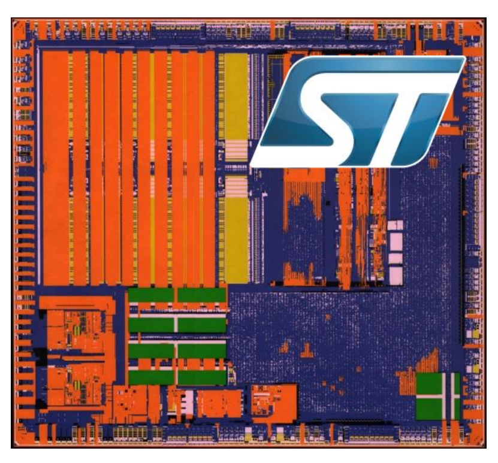
Positive side-effects – previously generated test programs are **evaluated** on a larger set of faults before starting a new generation phase.



Module	#faults	Single 1A FC [%]	Synchro 1A FC [%]	Single 1B FC [%]	Synchro 1B FC [%]	Single 2A FC [%]	Single 2B FC [%]	Synchro 2A+2B FC [%]	Single 3 FC [%]	Synchro 3 FC [%]
Functional Units	140k	**86.89**	89.21	--	--	--	--	89.51	--	91.78
Branch Units	72k	--	--	**75.07**	75.52	--	--	76.43	--	78.28
Register Bank	210k	--	<u>70.19</u>	--	--	**89.54**		93.53	--	95.38
Addressing modules	31k	--	--	--	<u>66.29</u>		**80.34**	81.59	--	82.33
Pipeline modules	278k	--	--	--	--	--	--	<u>64.59</u>	**79.91**	81.10
Glue logic	19k	--	--	--	--	--	--	--	--	<u>63.36</u>
TOTAL	**760k**	**--**	**36.07**	**--**	**9.74**	**--**	**--**	**76.87**	**--**	**87.23**

Additional aspects are considered when dealing with industrial test programs development:

- Time and memory **constraints**
- Coexistence with the final application (**mission**) and the Operating System
- Robust execution.



- Results refer to a SoC employed in safety-critical automotive embedded systems, such as airbag, ABS, and EPS controllers. SoC is currently manufactured by STMicroelectronics.
- 73 test programs written in assembly
- Full execution time: 0.8 ms (@150 MHz).

[3] P. Bernardi, R. Cantoro, S. De Luca, E. Sánchez and A. Sansonetti, "Development Flow for On-Line Core Self-Test of Automotive Microcontrollers," in *IEEE Transactions on Computers*, vol. 65, no. 3, pp. 744-754, March 1 2016.