

How to quickly deploy a SoC on FPGA to evaluate security solutions for communicating embedded systems?

International Winter School on Microarchitectural Security 2022

Philippe TANGUY
December 06, 2022
Lab-STICC



Introduction

Objectives :

- ▶ Explained our needs and requirements related to our research topic

Objectives :

- ▶ Explained our needs and requirements related to our research topic
- ▶ Present a tool to generate a System-On-Chip quickly for FPGA :
RETX about LiteX

Objectives :

- ▶ Explained our needs and requirements related to our research topic
- ▶ Present a tool to generate a System-On-Chip quickly for FPGA :
RETX about LiteX
- ▶ Introduce our hands-on : introduce LiTeX and how it could be usefull for security evaluation of embedded system software/hardware counter measures.

Problems Statements and Requirements

- ▶ An embedded systems involves skills at the hardware level and software level

General Problems Statements

- ▶ An embedded systems involves skills at the hardware level and software level
- ▶ Developing for an embedded system needs to manage several tools : toolchain, several language, scripting, debug tools, JTAG, ...

General Problems Statements

- ▶ An embedded systems involves skills at the hardware level and software level
- ▶ Developing for an embedded system needs to manage several tools : toolchain, several language, scripting, debug tools, JTAG, ...
- ▶ How to go from idea to proof-of-concept quickly : for demonstration, etc.

General Problems Statements

- ▶ An embedded systems involves skills at the hardware level and software level
- ▶ Developing for an embedded system needs to manage several tools : toolchain, several language, scripting, debug tools, JTAG, ...
- ▶ How to go from idea to proof-of-concept quickly : for demonstration, etc.
- ▶ A PhD thesis is only 3 years (in France)

General Problems Statements

- ▶ An embedded systems involves skills at the hardware level and software level
- ▶ Developing for an embedded system needs to manage several tools : toolchain, several language, scripting, debug tools, JTAG, ...
- ▶ How to go from idea to proof-of-concept quickly : for demonstration, etc.
- ▶ A PhD thesis is only 3 years (in France)
- ▶ How to manage research artifacts certified as reproducible

- ▶ At the hardware level a System On Chip can be complex and at some point we contribute to a specific part of the overall system

- ▶ At the hardware level a System On Chip can be complex and at some point we contribute to a specific part of the overall system
- ▶ Hardware description language

- ▶ Open source tools as much as possible

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community
- ▶ Do not reinvent the wheel

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community
- ▶ Do not reinvent the wheel
- ▶ Build automation

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community
- ▶ Do not reinvent the wheel
- ▶ Build automation
- ▶ Support several FPGA, devkit boards, ...

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community
- ▶ Do not reinvent the wheel
- ▶ Build automation
- ▶ Support several FPGA, devkit boards, ...
- ▶ Make comparison of several implementation without developping everything from scratch

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community
- ▶ Do not reinvent the wheel
- ▶ Build automation
- ▶ Support several FPGA, devkit boards, ...
- ▶ Make comparison of several implementation without developing everything from scratch
- ▶ Have examples or support about deploying RTOS OS and/or Linux

Requirements

- ▶ Open source tools as much as possible
- ▶ Benefits of community
- ▶ Do not reinvent the wheel
- ▶ Build automation
- ▶ Support several FPGA, devkit boards, ...
- ▶ Make comparison of several implementation without developing everything from scratch
- ▶ Have examples or support about deploying RTOS OS and/or Linux

Which tool to use?

We choose to use LiteX!

Quick demo!

FPGA 101 (in 5min)

Terminology

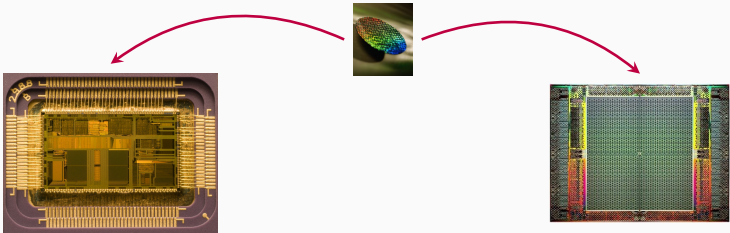
- ▶ ASIC : Application-specific integrated circuit
- ▶ PLD : Programmable Logic Device

ASIC vs PLD

- ▶ VHDL (or Verilog, ...) allows to describes hardware for ASIC or PLD
- ▶ ASIC are dedicted component
- ▶ PLD are programmable component

Au commencement ...

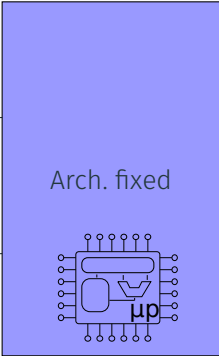
Microprocessor and FPGA are dedicated component.



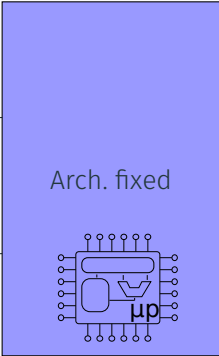

- ▶ First **microprocessor** (1971)
- ▶ Allows to execute instruction and process program data

- ▶ First **FPGA** (field programmable gate array) (1985)
- ▶ Allows to define a circuit to realize a function (Xor, FFT, CPU)

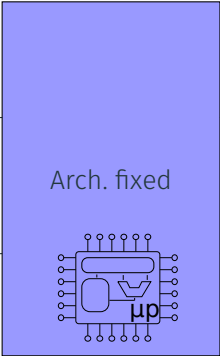


From transistor to algorithm

Niveau	Microprocessorr	FPGA
Transistor	 <p>Arch. fixed</p>	
Logic		
Architecture		
Algorithm		

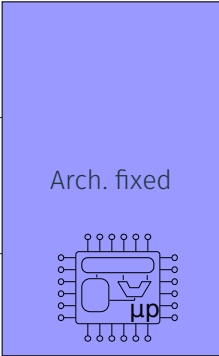

From transistor to algorithm

Niveau	Microprocessorr	FPGA	
Transistor			
Logic		Arch. fixed	
Architecture			
Algorithm			

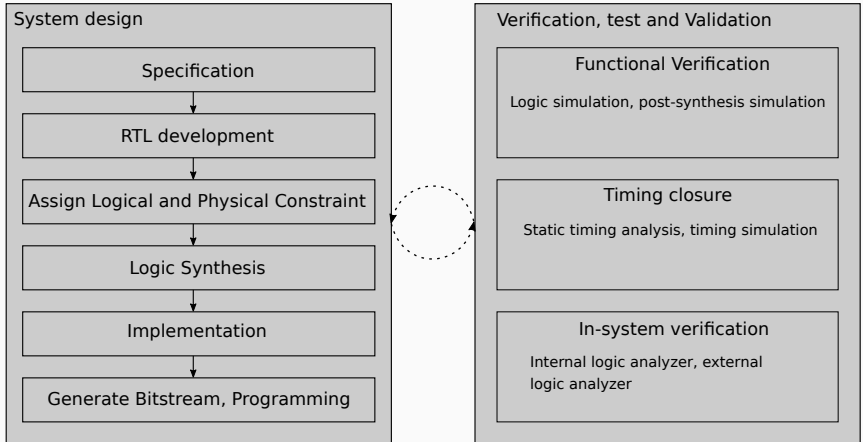
From transistor to algorithm

Niveau	Microprocessorr	FPGA
Transistor		
Logic		
Architecture		
Algorithm		

From transistor to algorithm

Niveau	Microprocessorr	FPGA
Transistor		Arch. fixed 
Logic		Netlist
Architecture		HDL prog.
Algorithm	Soft for CPU	Soft SoC

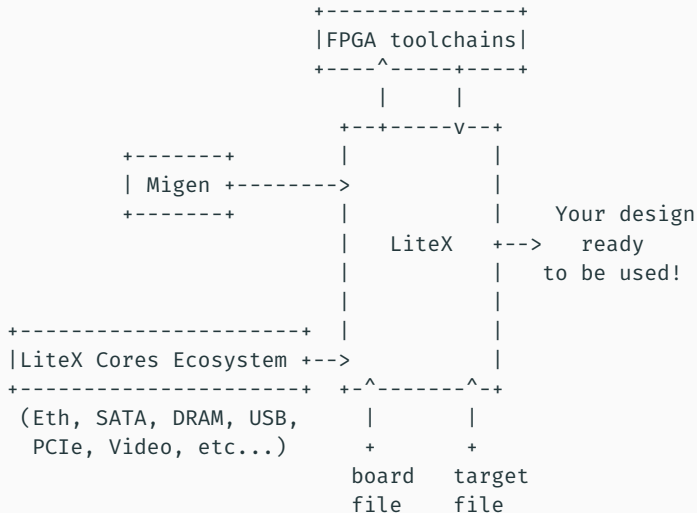
Overview



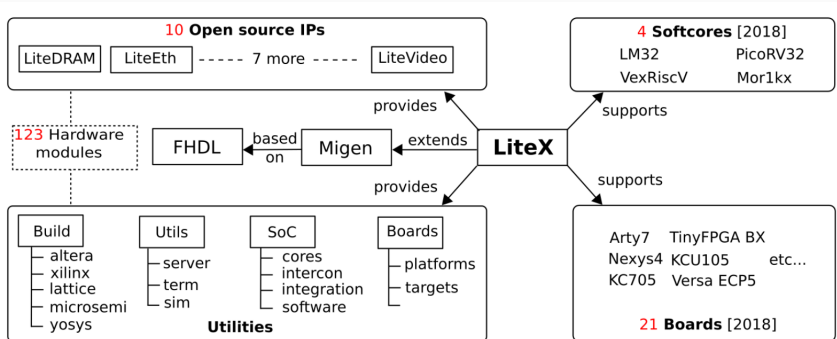
Example on the demo with Vivado!

LiteX 101

LiteX?



LiteX and component ?

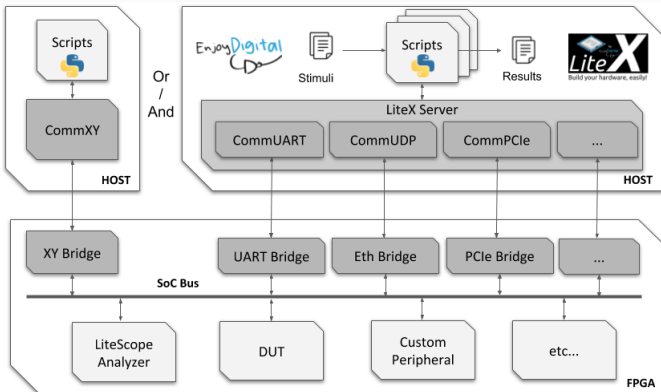


Img credits : <https://pcotret.github.io/ENSTAB-RISCV/>

- ▶ Support a lot of FPGA chip, boards and dev kit!

```
1 (pyenv-litex)> ls litex-boards/litex_boards/targets/  
2 zsh: do you wish to see all 132 possibilities (66 lines)?  
3 (pyenv-litex)> ls -l litex-boards/litex_boards/targets/digilent_*  
4 litex-boards/litex_boards/targets/digilent_arty.py  
5 litex-boards/litex_boards/targets/digilent_basys3.py  
6 litex-boards/litex_boards/targets/digilent_nexys4.py  
7 litex-boards/litex_boards/targets/digilent_arty_s7.py  
8 litex-boards/litex_boards/targets/digilent_cmod_a7.py  
9 litex-boards/litex_boards/targets/digilent_nexys_video.py  
10 litex-boards/litex_boards/targets/digilent_arty_z7.py  
11 litex-boards/litex_boards/targets/digilent_genesys2.py  
12 litex-boards/litex_boards/targets/digilent_pynq_z1.py  
13 litex-boards/litex_boards/targets/digilent_atlys.py  
14 litex-boards/litex_boards/targets/digilent_nexys4ddr.py  
15 litex-boards/litex_boards/targets/digilent_zedboard.py
```

LiteX debug infrastructure

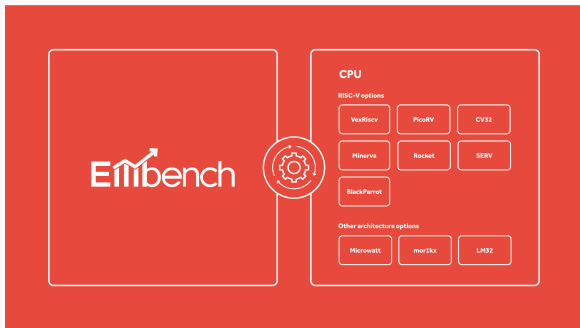


LiteX Remote Control/Debug Infrastructure

Img credits : LiTeX Github

Interesting project/paper using LiteX i

- ▶ LiteX embench tester
<https://github.com/antmicro/embench-tester>

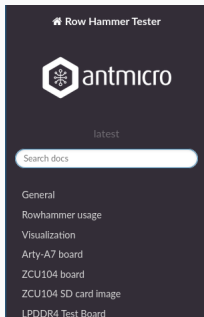


img credits : antmicro

Interesting project/paper using LiteX ii

▶ LiteX rowhammer tester

<https://github.com/antmicro/rowhammer-tester>



Img credits : antmicro

[Docs](#) » Welcome to Row Hammer Tester!

[View repository on GitHub](#)

Welcome to Row Hammer Tester!

- **General**
 - Architecture
 - Installing dependencies
 - Packaging the bitstream
 - Local documentation build
 - Tests
- **Rowhammer usage**
 - Network USB adapter setup
 - Controlling the board
 - Hammering
 - Utilities
 - Simulation

Conclusion

- ▶ Usefull for constraint embedded system : simple SoC, simple Core, ...
- ▶ It is not an industrial framework but it works ...

- ▶ Labs available online

<https://sourcesup.renater.fr/www/mic-sec-2022/>

- ▶ 3 labs :

- ▶ Lab 01 : Getting Started with LiteX
- ▶ Lab 02 : Create a minimal SoC with LiTeX
- ▶ Lab 03 : Software app for a SoC with LiTeX

Questions/Discussions?

(Hands-on after the break!)

Acknowledgements

- ▶ LiteX community
- ▶ Thesis Mohamed El-bouazzati