IIS5008
Hardware Security

User's Guide

Andy, Yu-Guang Chen
Assistant Professor, Department of EE
National Central University
andyygchen@ee.ncu.edu.tw

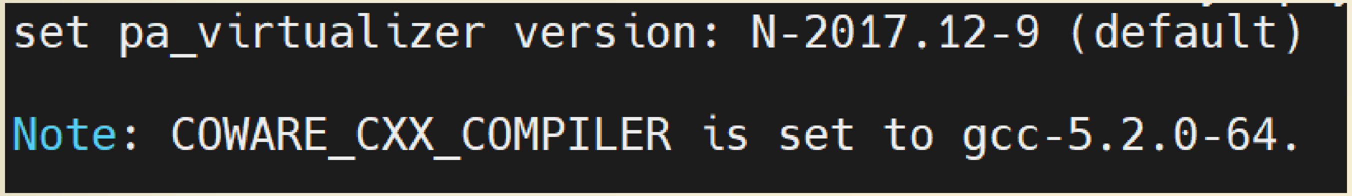
Slides Credit: TA Chi-Tse Pai

Outline

- ◆Workstation
- ◆Project Construction
- ◆PA3 Introduction

Workstation

- ◆ Login in to the server.
- ◆ Source PA
  - ➢ source /usr/cad/synopsys/CIC/pa_virtualizer.cshrc
  - ➢ You’ll see the information like :



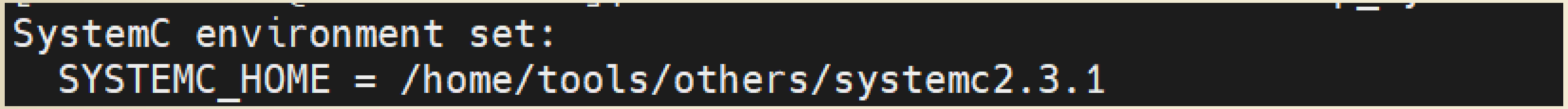
```
set pa_virtualizer version: N-2017.12-9 (default)

Note: COWARE_CXX_COMPILER is set to gcc-5.2.0-64.
```

- ◆ Extract file
  - ➢ Extract the compressed file HW3.tar.gz downloaded from eeclass.

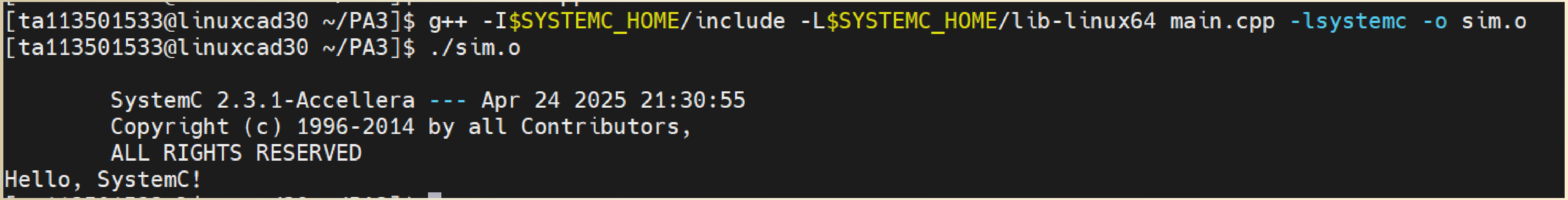
Workstation

- Source systemC
  - source /home/tools/others/setup_systemc.csh 2.3.1
  - You’ll see the information like :



```
SystemC environment set:
  SYSTEMC_HOME = /home/tools/others/systemc2.3.1
```

- Compile and run systemC file, you can test it by provided main.cpp.



```
[ta113501533@linuxcad30 ~/PA3]$ g++ -I$SYSTEMC_HOME/include -L$SYSTEMC_HOME/lib-linux64 main.cpp -lsystemc -o sim.o
[ta113501533@linuxcad30 ~/PA3]$ ./sim.o

        SystemC 2.3.1-Accellera --- Apr 24 2025 21:30:55
        Copyright (c) 1996-2014 by all Contributors,
        ALL RIGHTS RESERVED
Hello, SystemC!
```

Workstation

- Using the Platform Architecture to simulate output result.
- SystemC simulation
  - cd <your_directory>/PA3
  - pct&

Workstation

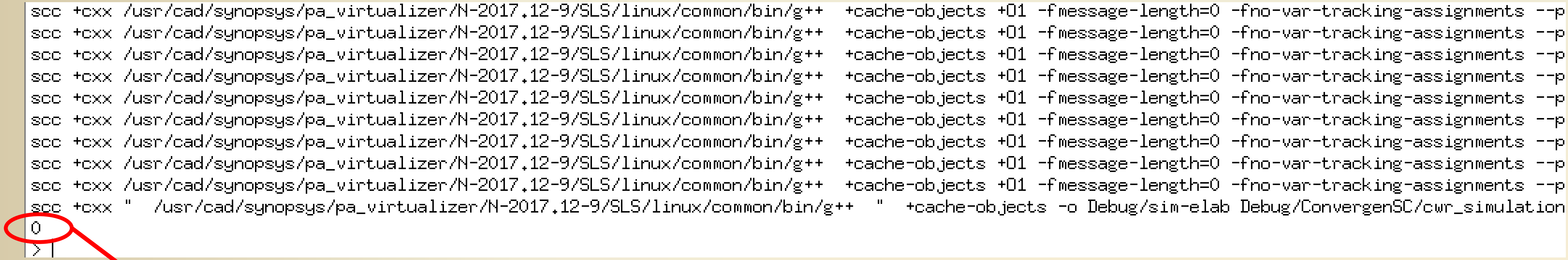
- vpsession provides a Tcl command shell used for construction, configuration, simulation, and export.
- scc is Synopsys’ platform-specific packaging compiler.
- sim-elab is an intermediate file generated during the elaboration stage.
- sim is the final simulation executable.
- After connecting block diagram, run simulation.
  - The entire SystemC project is compiled within the interactive shell of vpsession in Virtualizer.
  - Use Synopsys' packaged scc command along with the GNU g++ compiler.

Workstation

- ◆ ::scsh::open-project
- ◆ ::scsh::cosim::enable_hdl_sdi
- ◆ ::scsh::build-options -skip-elab on
  - ➢ Skip elaboration and build directly.
- ◆ ::set_maf mem_map
- ◆ ::scsh::build
  - ➢ Generate sim.exe and prepare for simulation.

Workstation

- After running simulation or sim.tcl script, console:

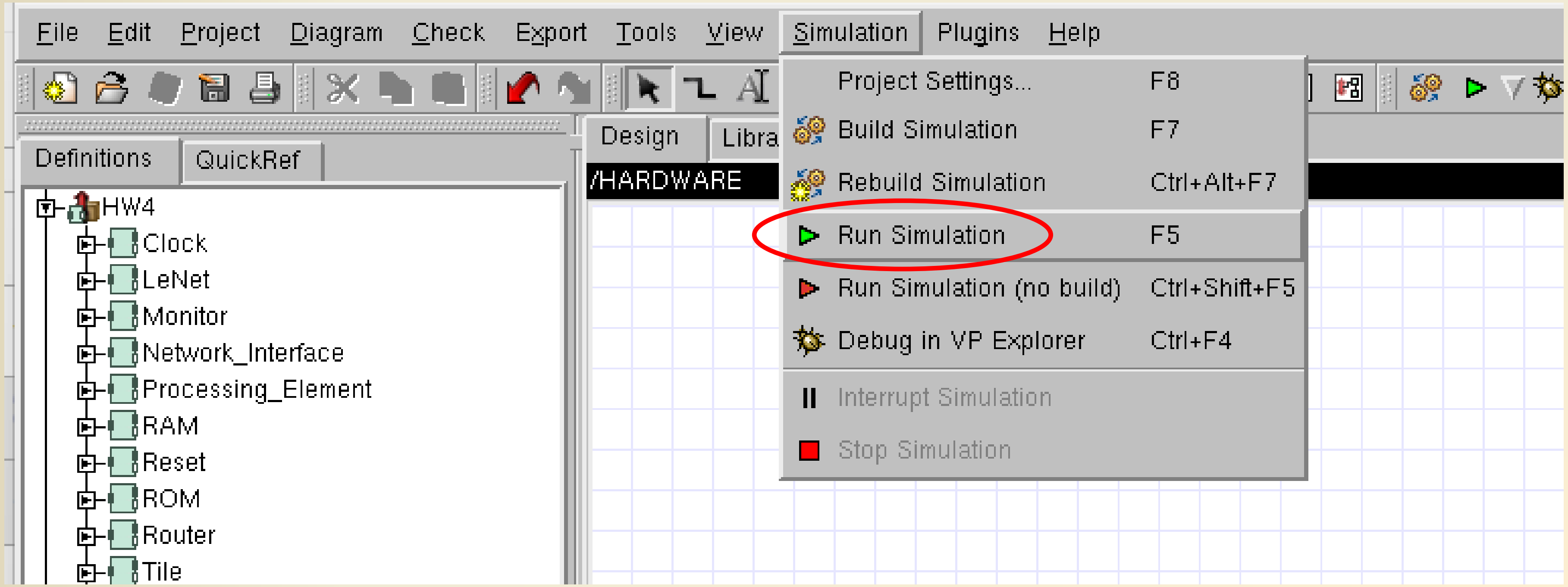


```
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  +cache-objects +O1 -fmessage-length=0 -fno-var-tracking-assignments --p
scc +cxx "  /usr/cad/synopsys/pa_virtualizer/N-2017.12-9/SLS/linux/common/bin/g++  "  +cache-objects -o Debug/sim-elab Debug/ConvergenSC/cwr_simulation
0
>
```

means complete successfully
(exit code is 0)

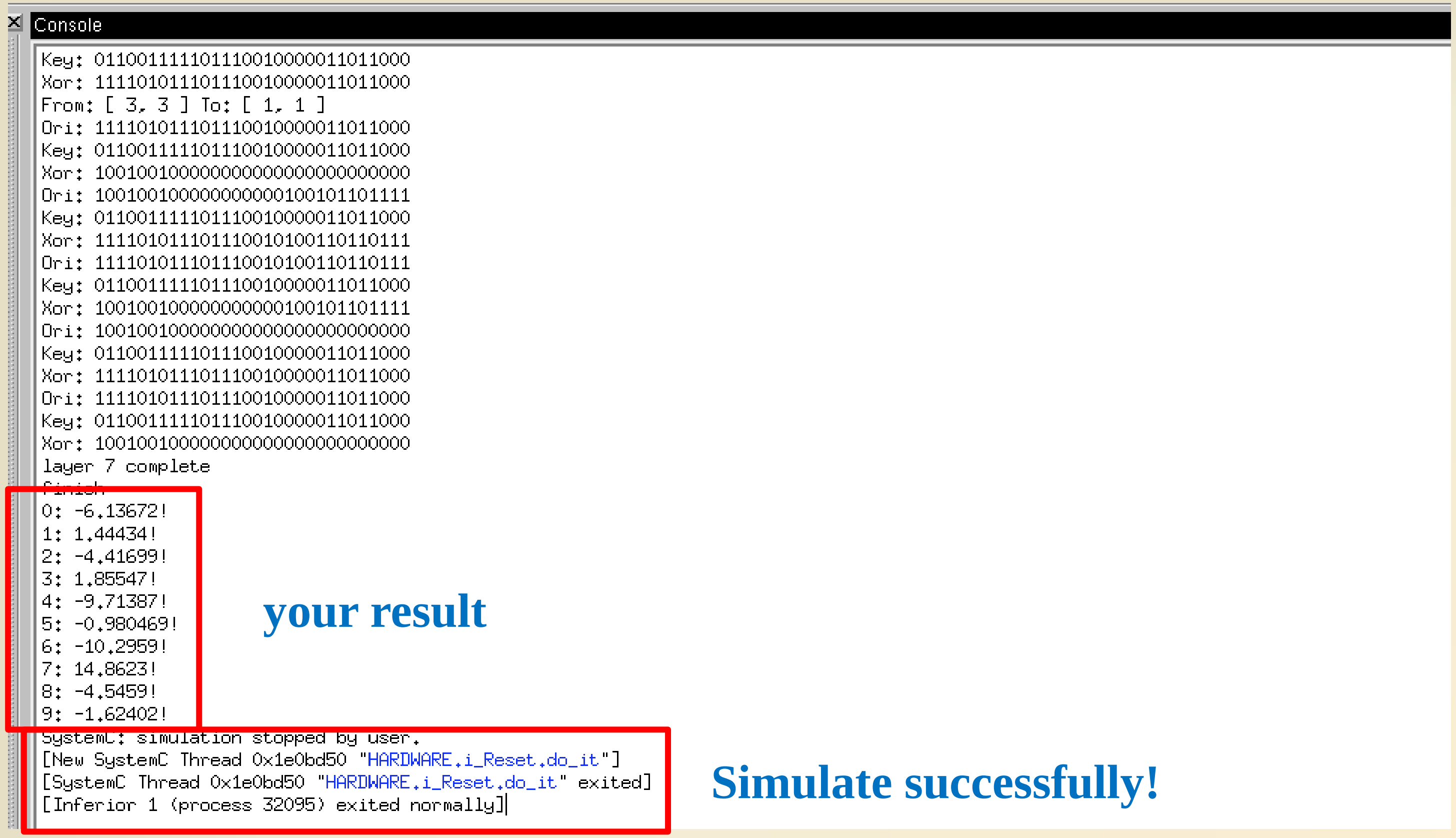
Workstation

- Simulation > Run simulation.



Workstation

- Console:



```
Key: 01100111110111001000001101100 0
```

your result

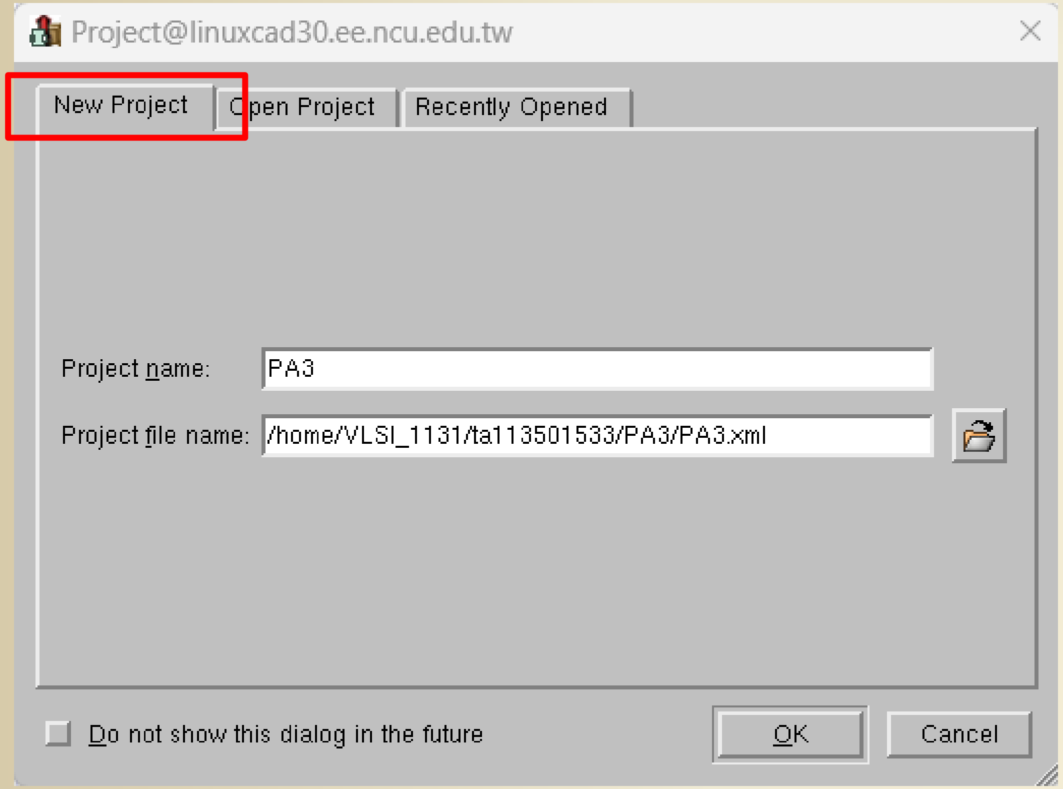
Simulate successfully!

Outline

- Workstation
- Project Construction
- PA3 Introduction

Project Construction

◆Create new project or open existed project.

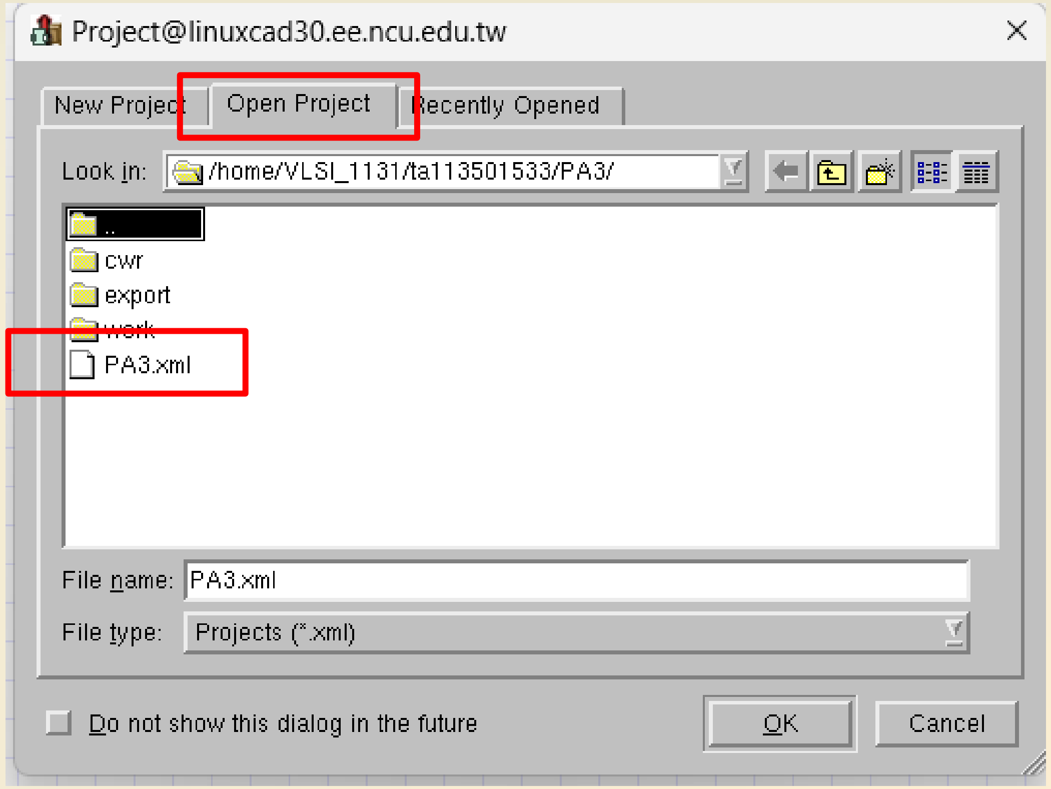


Project@linuxcad30.ee.ncu.edu.tw

Project name: PA3

Project file name: /home/VLSI_1131/ta113501533/PA3/PA3.xml

□ Do not show this dialog in the future



Project@linuxcad30.ee.ncu.edu.tw

Look in: /home/VLSI_1131/ta113501533/PA3/

```
..
cwr
export
work
PA3.xml
```

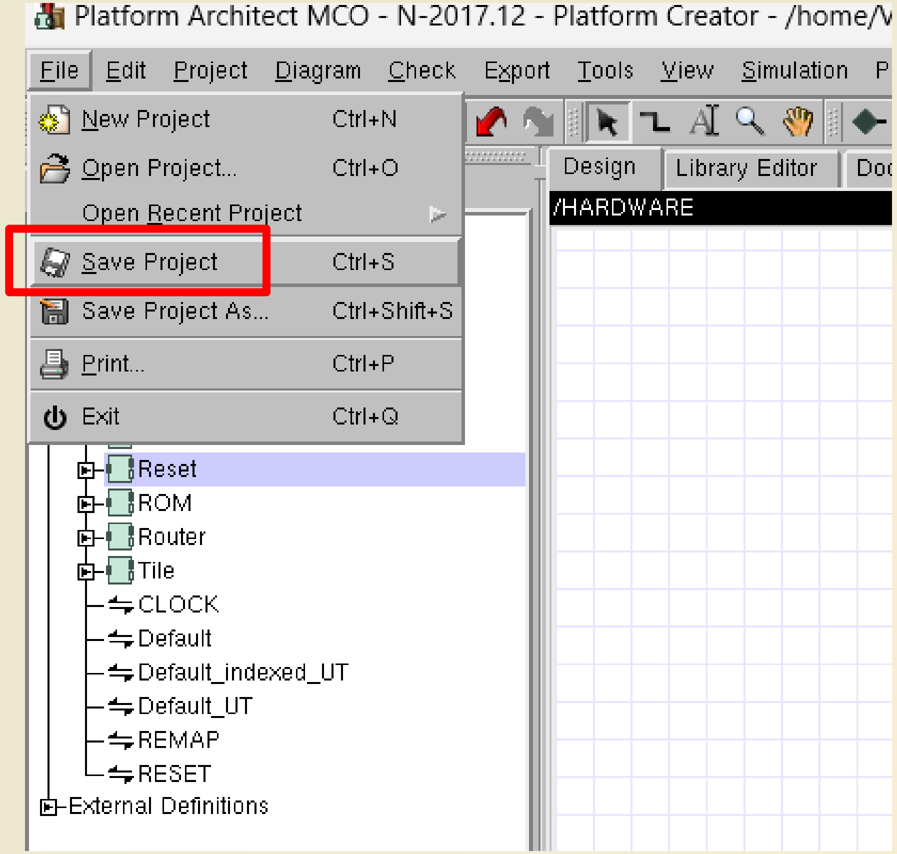
File name: PA3.xml

File type: Projects (*.xml)

□ Do not show this dialog in the future

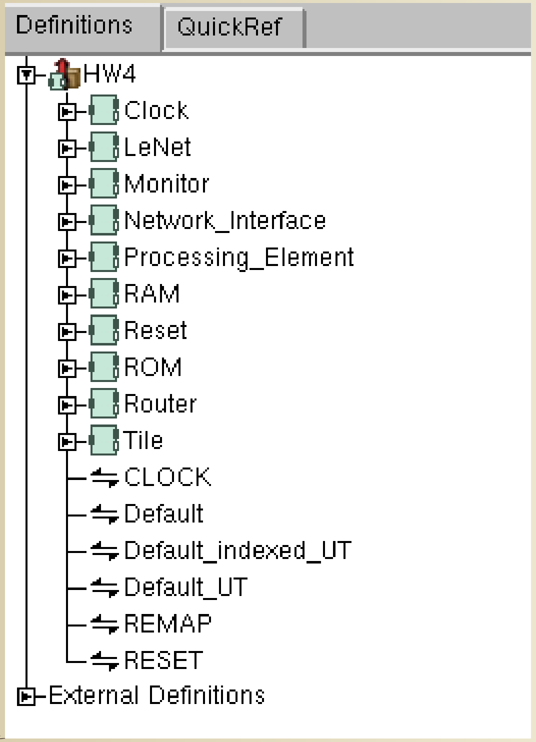
Project Construction

- ◆Remember to save project after modifying.



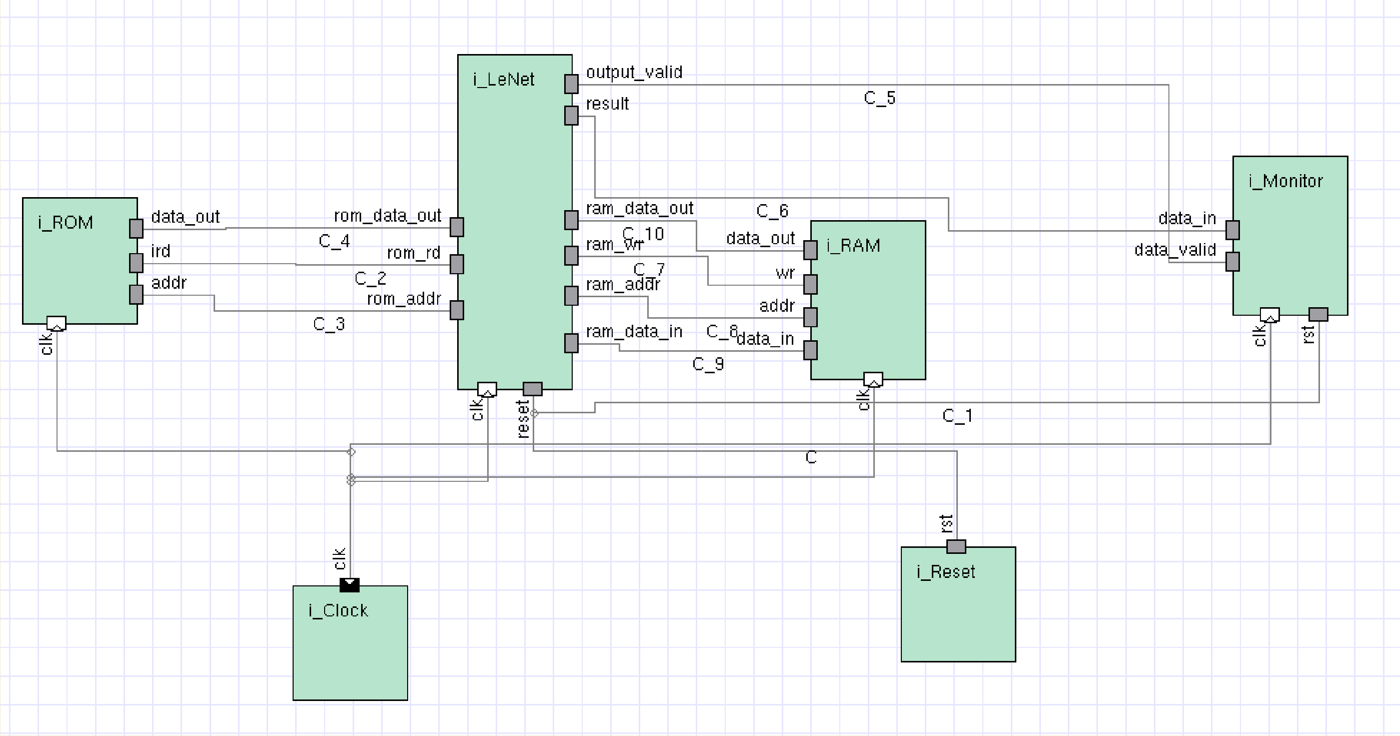
Project Construction

- ◆Create PA3 block diagram
  - ➢ You should construct it step by step as following slides.



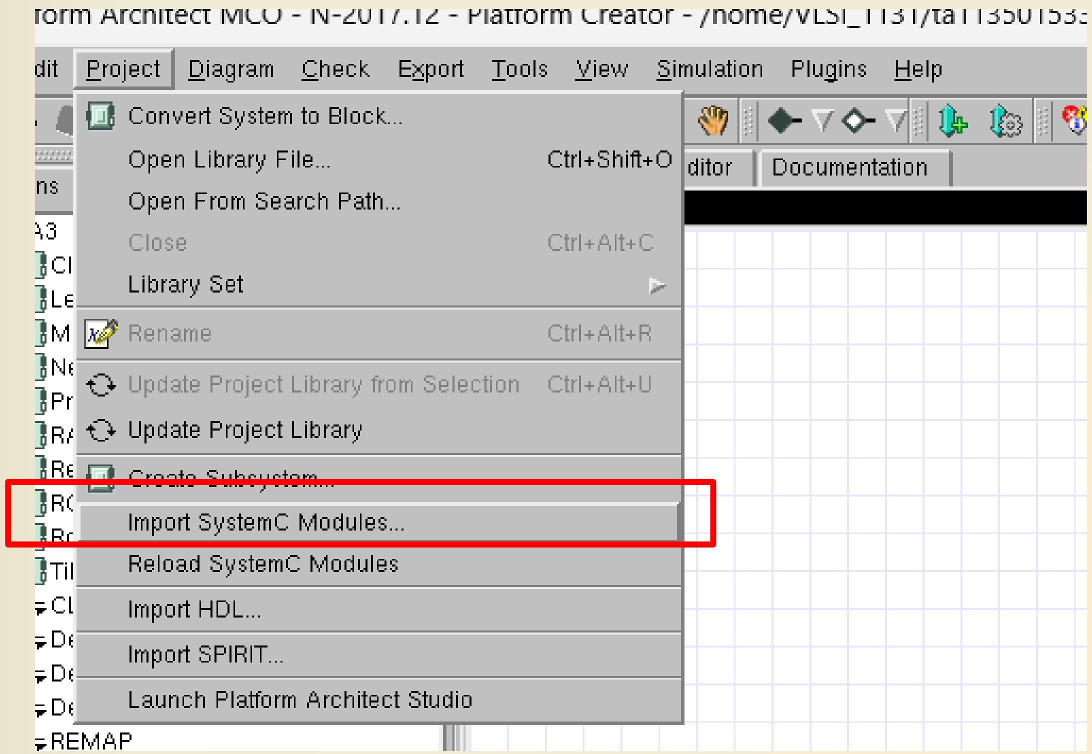
Definitions | QuickRef

- HW4
  - Clock
  - LeNet
  - Monitor
  - Network_Interface
  - Processing_Element
  - RAM
  - Reset
  - ROM
  - Router
  - Tile
  - CLOCK
  - Default
  - Default_indexed_UT
  - Default_UT
  - REMAP
  - RESET
- External Definitions



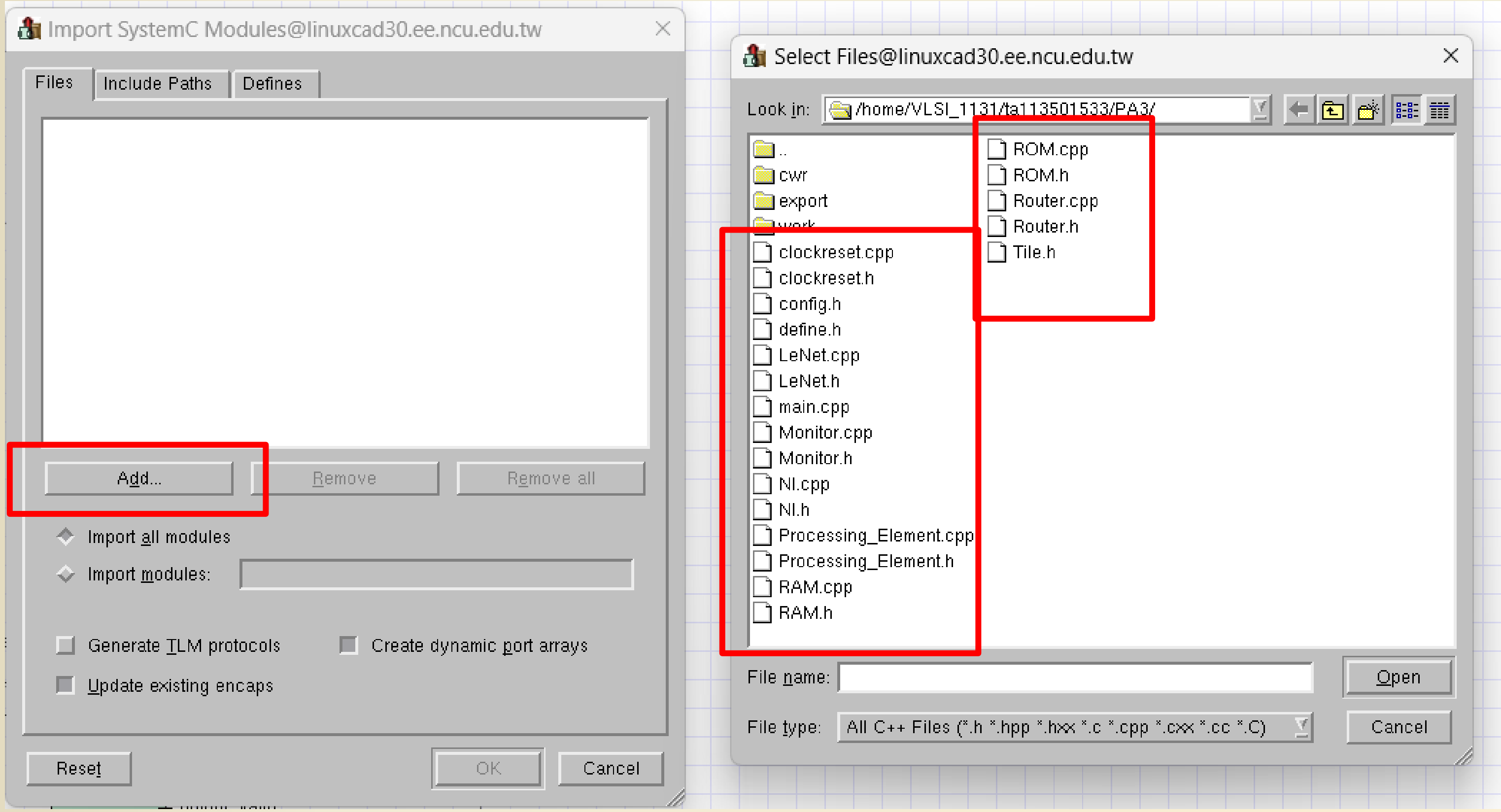
Project Construction

- ◆Import systemC source files
  - ➢ Project > Import SystemC Modules…



Project Construction

- Import systemC source files
  - Add > select all source files



Look in: /home/VLSI_1131/ta113501533/PA3/

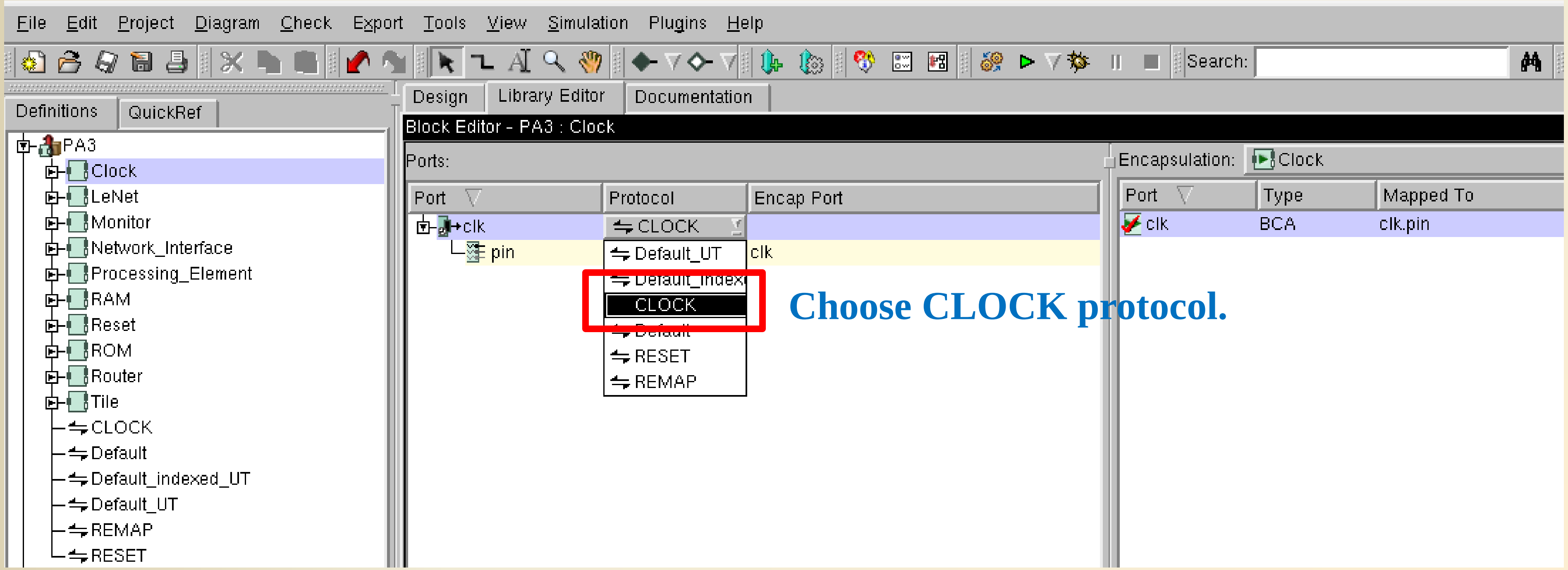
..
cwr
export
work
clockreset.cpp
clockreset.h
config.h
define.h
LeNet.cpp
LeNet.h
main.cpp
Monitor.cpp
Monitor.h
NI.cpp
NI.h
Processing_Element.cpp
Processing_Element.h
RAM.cpp
RAM.h
ROM.cpp
ROM.h
Router.cpp
Router.h
Tile.h

File name:

File type: All C++ Files (*.h *.hpp *.hxx *.c *.cpp *.cxx *.cc *.C)

Project Construction

- ◆Edit protocol of clock port.
  - ➢ Double-click the design in left definition window.
  - ➢ Select CLOCK protocol for those designs with clock port.



Block Editor - PA3 : Clock

Ports:

Port	Protocol	Encap Port
clk	CLOCK	
pin	Default_UT	clk

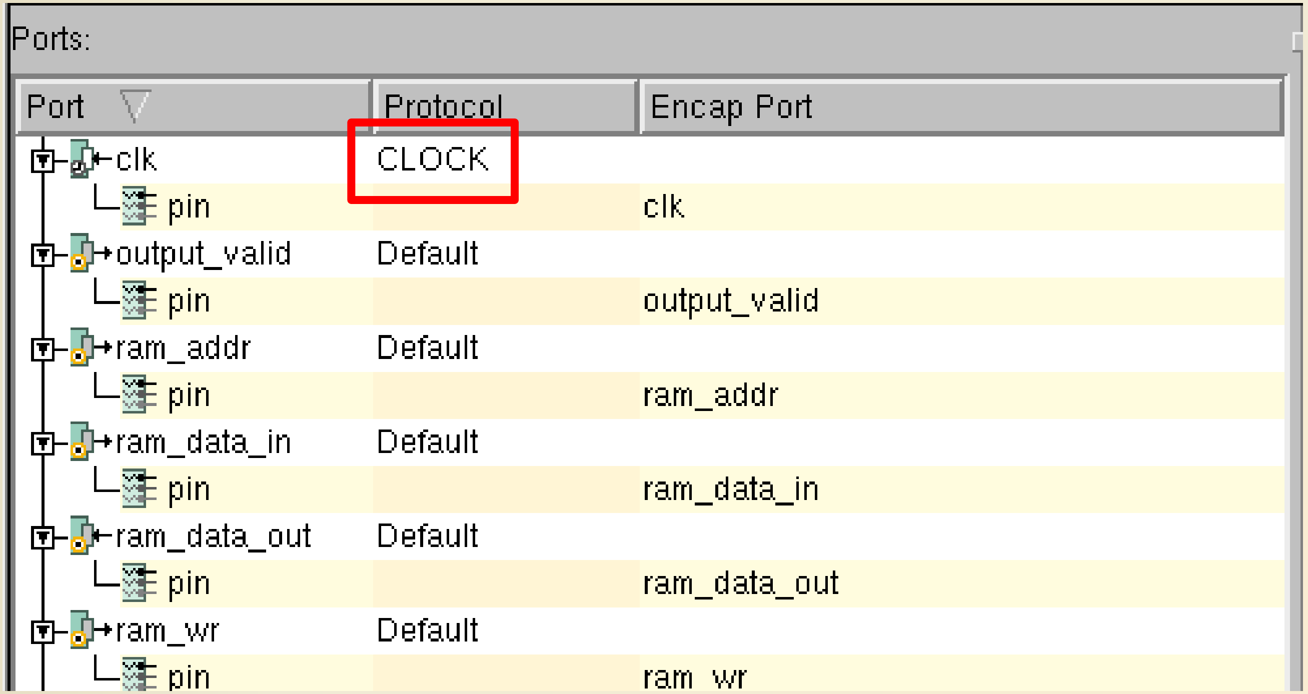
Encapsulation: Clock

Port	Type	Mapped To
clk	BCA	clk.pin

Choose CLOCK protocol.

Project Construction

- ◆For example, you also need to change protocol of LeNet clock port.

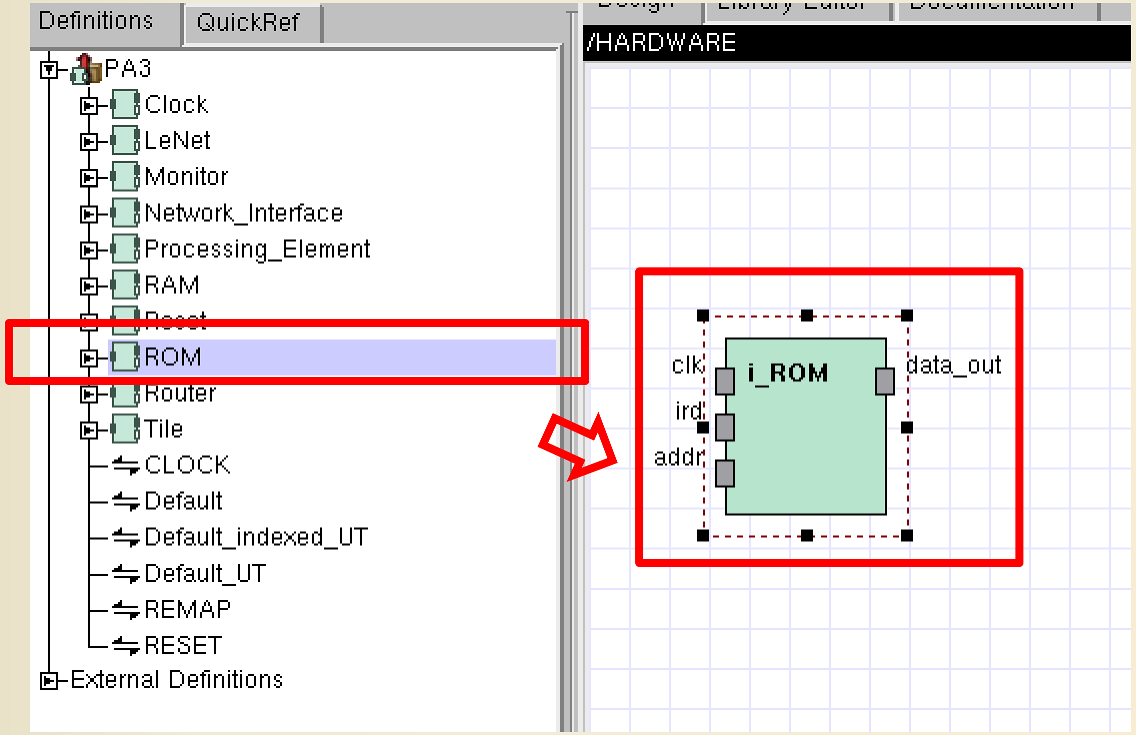


Ports:

Port	Protocol	Encap Port
clk	CLOCK	
pin		clk
output_valid	Default	
pin		output_valid
ram_addr	Default	
pin		ram_addr
ram_data_in	Default	
pin		ram_data_in
ram_data_out	Default	
pin		ram_data_out
ram_wr	Default	
pin		ram_wr

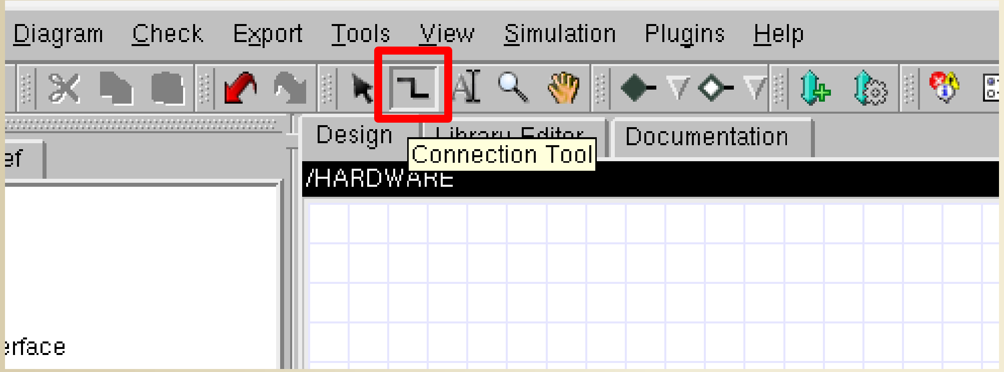
Project Construction

- Create the block
  - Click and drag to the design window.

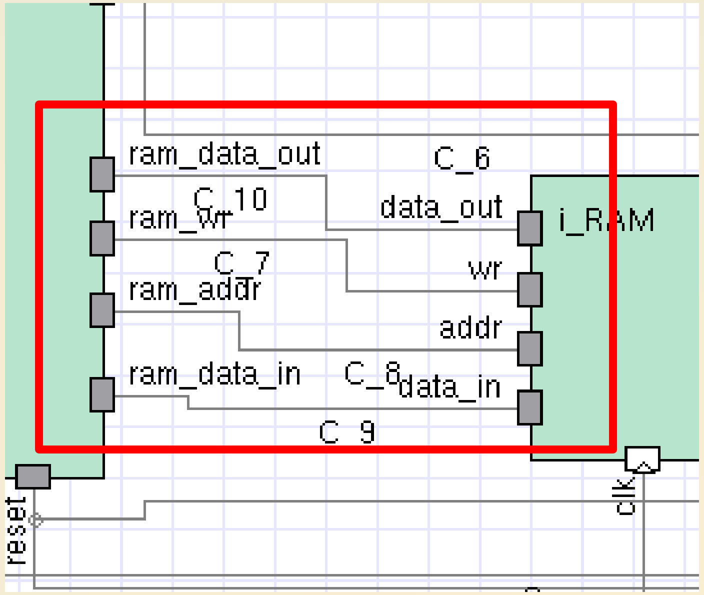


Project Construction

- Use connection tool to connect blocks.

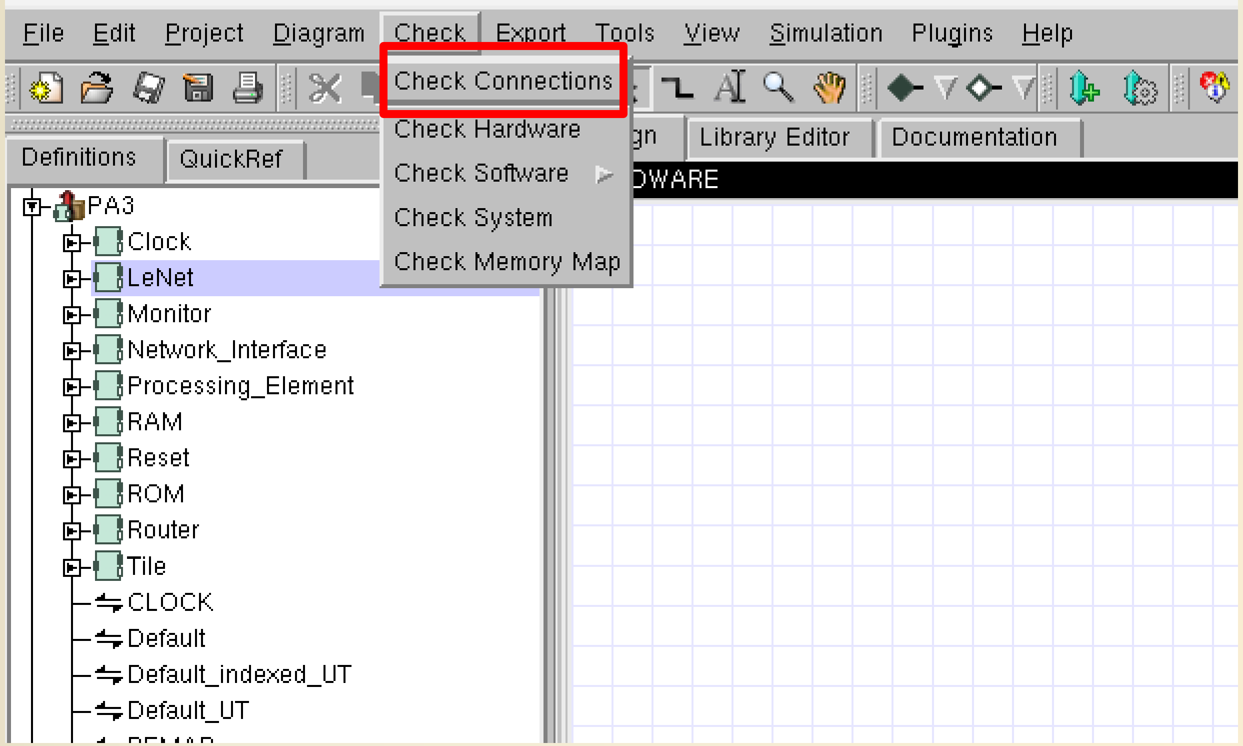


Connect ports based on your design.cpp.



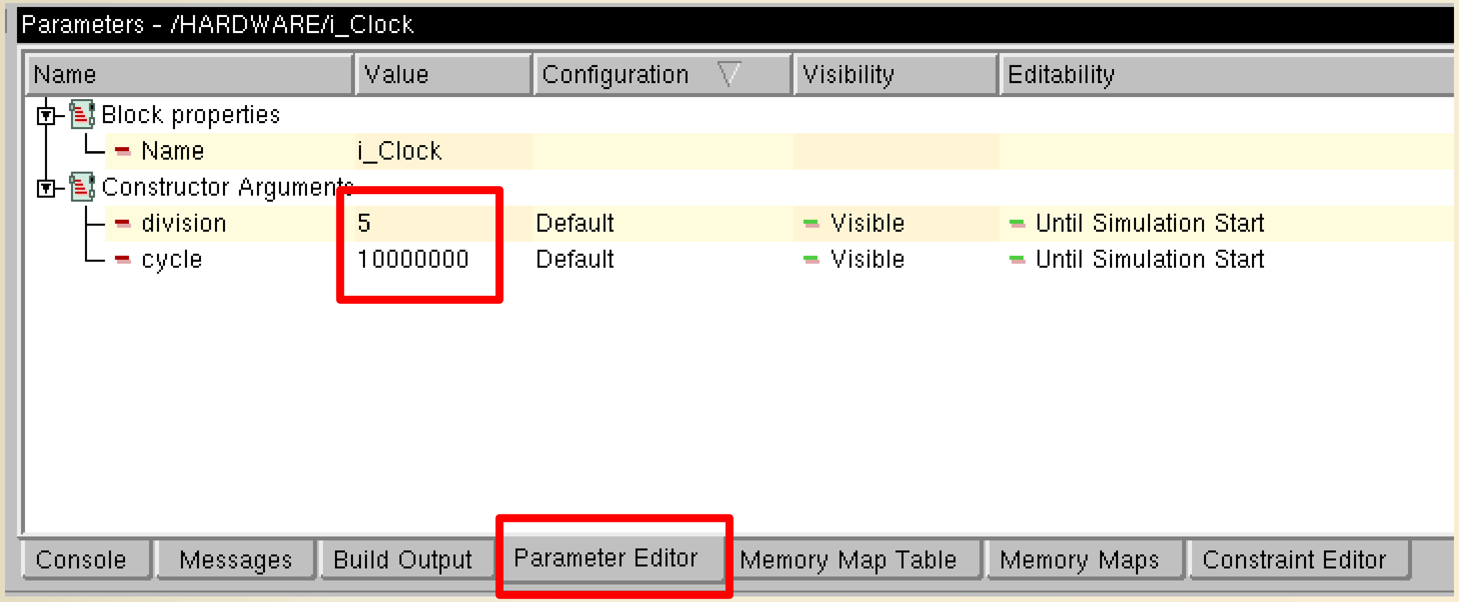
Project Construction

- ◆After connecting, you can use check tool to check connection correctness.



Project Construction

- ◆Set clock parameters.
  - ➢ Double-click the design block (Clock).
  - ➢ Set division value and cycle value as below figure.



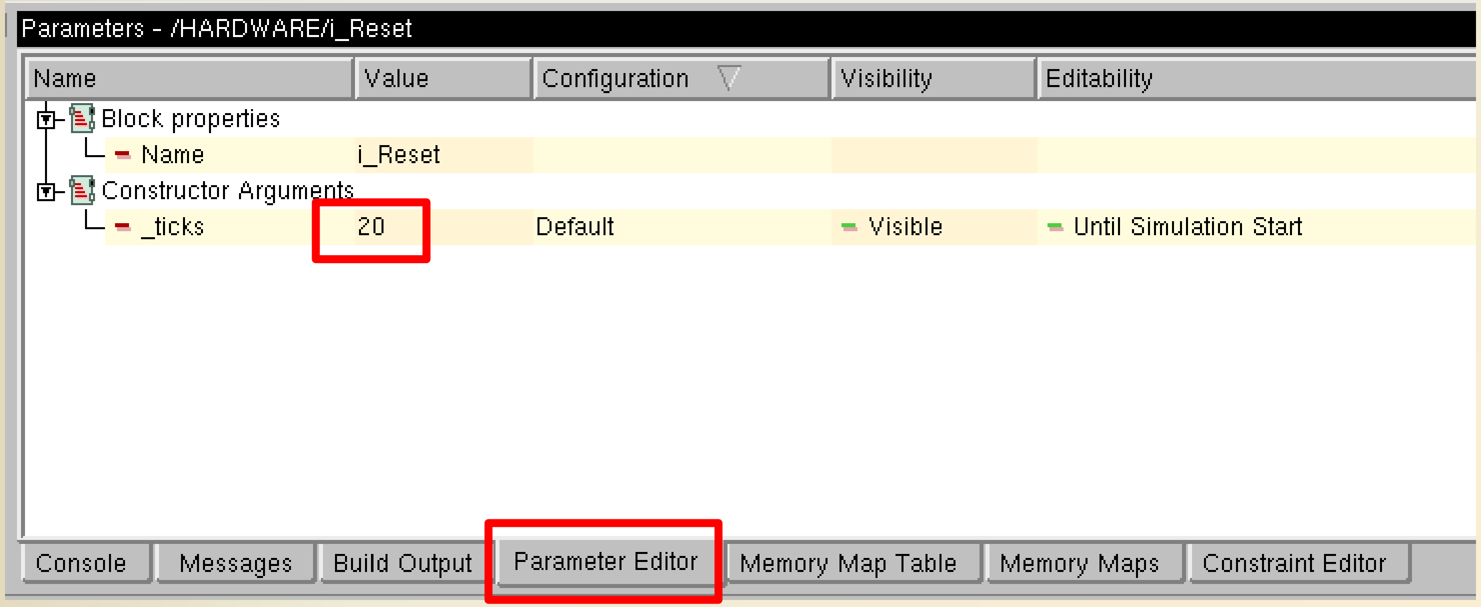
Parameters - /HARDWARE/i_Clock

Name	Value	Configuration	Visibility	Editability
Block properties				
Name	i_Clock			
Constructor Arguments				
division	5	Default	Visible	Until Simulation Start
cycle	10000000	Default	Visible	Until Simulation Start

Project Construction

◆Set reset parameter.

- ➢ Double-click the design block (Reset).
- ➢ Set tick value as below figure.

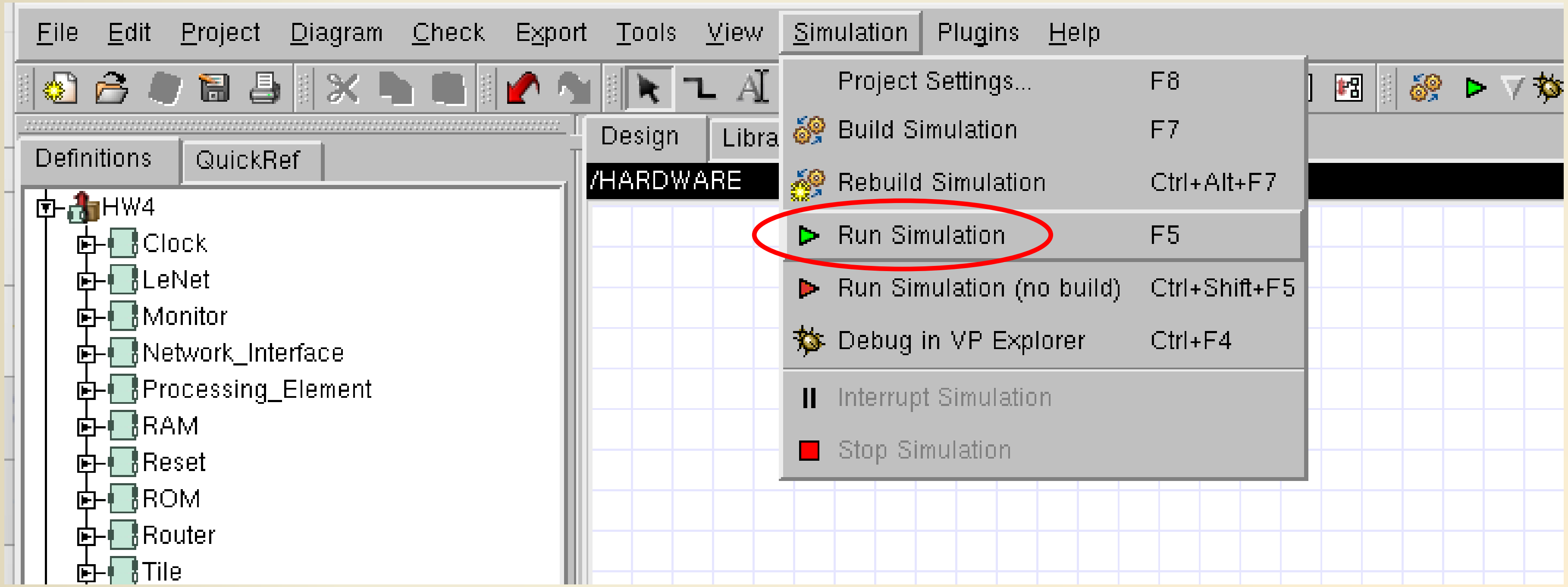


Parameters - /HARDWARE/i_Reset

Name	Value	Configuration	Visibility	Editability
Block properties				
Name	i_Reset			
Constructor Arguments				
_ticks	20	Default	Visible	Until Simulation Start

Project Construction

- ◆Simulation > Run simulation.



Outline

- ◆Workstation
- ◆Project Construction
- ◆PA3 Introduction

PA3 Introduction

- A 3×3 arrangement of nine Processing Elements (PEs) under a Network-on-Chip (NoC) architecture.
- Each tile consists of the following components:
  - Router：Handles data transmission between tiles over the NOC.
  - Processing_Element (PE)：Executes computations such as CNN convolution, pooling, dense operations, etc.
  - Network_Interface (NI)：Serves as the bridge between the Router and the PE, managing data transfers.

PA3 Introduction

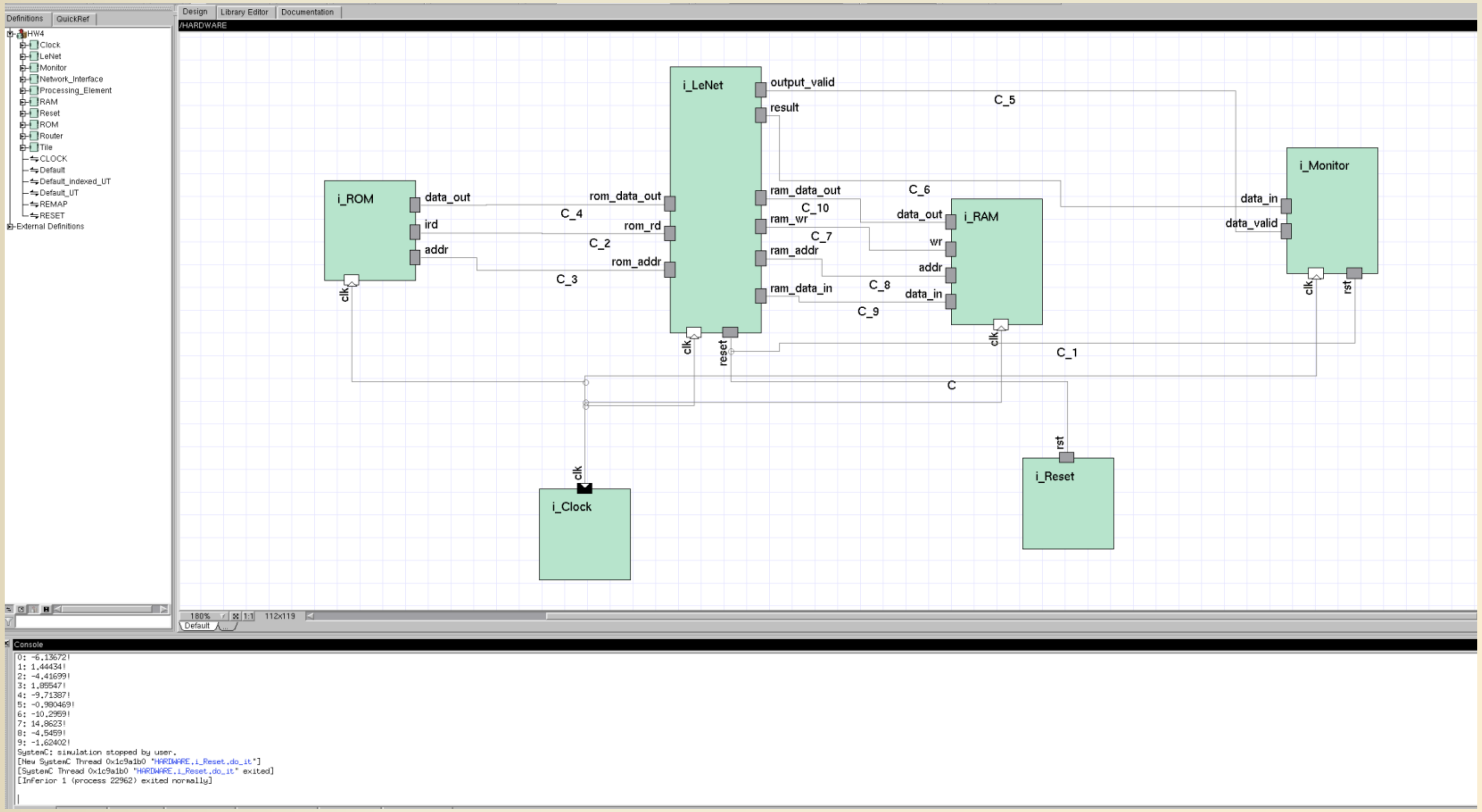
- PE attack on new stochastic computing.
- Example: Attack PE4, let one bit of the random pixel flip.
  - First step
    - Transfer the binary value to stochastic bit stream.
  - Second step (error injection)
    - Let one bit which in bit stream flip.
  - Third step
    - Transfer the stochastic bit stream to binary value.

PA3 Introduction

- Complete your design including error_injection.h and sc_function_bipolar.h in systemC.
- error_injection.h:
  - Simulate fault tolerance analysis in stochastic computing through fault injection.
  - Simulate the process of converting an integer into a stochastic bit stream, randomly injecting bit-flip errors, and then reconstructing the binary value.
- sc_function_bipolar.h:
  - Convert the input into a stochastic bit stream. (ex. using random number comparison with a threshold)
  - Convert the result back into a bipolar real value in the range of -1 to 1.
- You can test your files by designing your own main.cpp!!!

PA3 Introduction

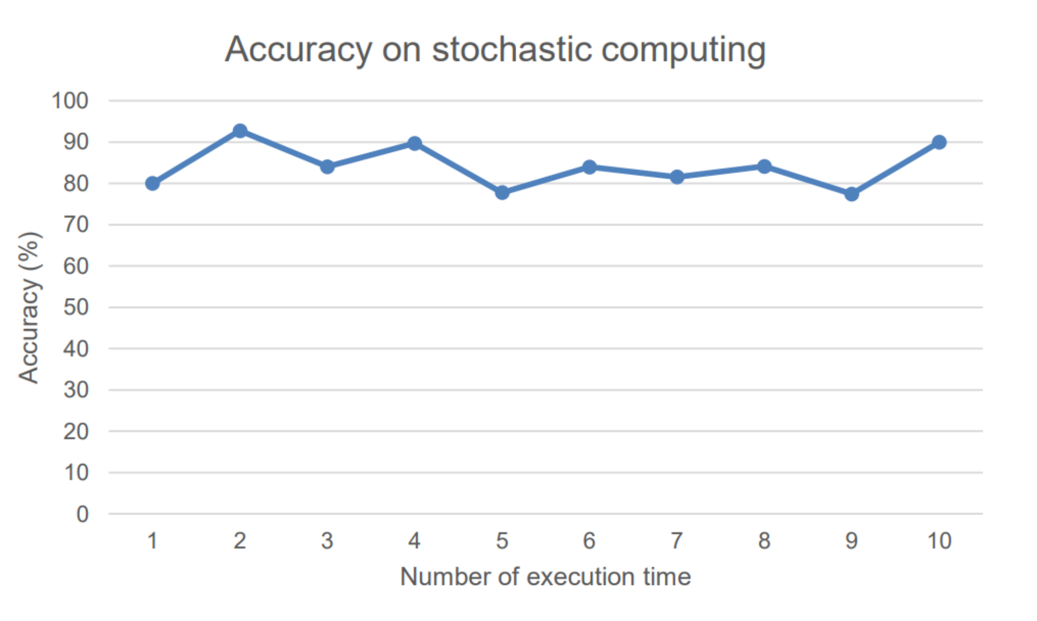
- Paste your screenshot in your report.

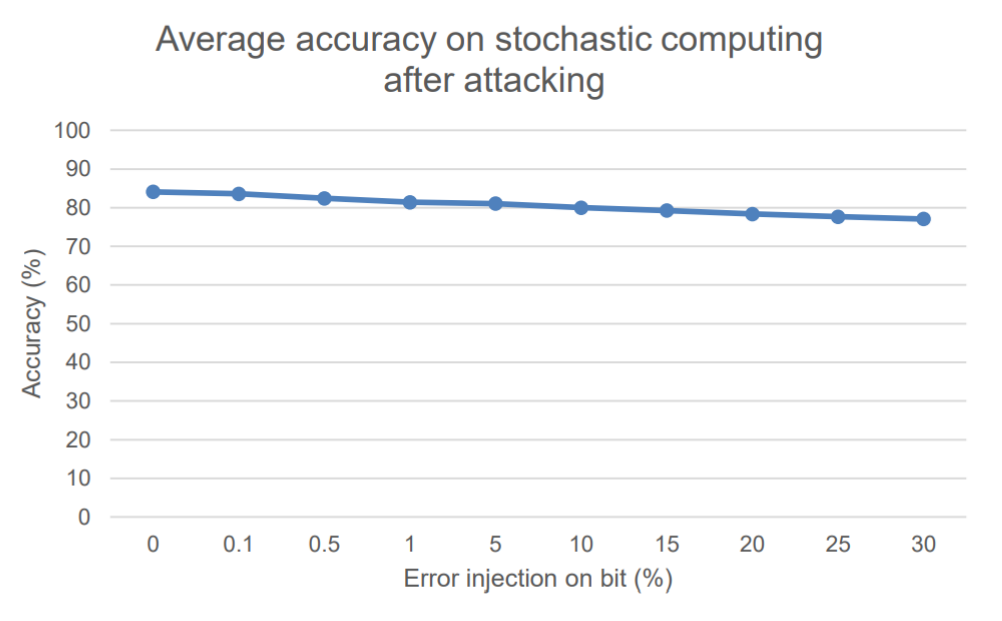


```
0: -6.13672!
1: 1.44434!
2: -4.41699!
3: 1.85547!
4: -9.71387!
5: -0.980469!
6: -10.2959!
7: 14.8623!
8: -4.5459!
9: -1.62402!
SystemC: simulation stopped by user.
[New SystemC Thread 0x1c9a1b0 "HARDWARE.i_Reset.do_it"]
[SystemC Thread 0x1c9a1b0 "HARDWARE.i_Reset.do_it" exited]
[Inferior 1 (process 22962) exited normally]
```

PA3 Introduction

- Compare results of one before attacking and one after attacking.
- Use graphs to present your experiment data as below figure.





Good Luck For Your PA3!!!