|  |
| --- |
| HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY  **SCHOOL OF ELECTRICAL**  **AND ELECTRONIC ENGINEERING**  logo_128  **NEURON GRID SIMULATION TUTORIAL**    **Instructor**: Assoc. Prof. Dr. NGUYEN DUC MINH  **Students**: NGUYEN LE TRUNG 20186076  VU HOANG LONG 20182926  PHAM HUY HOANG 20182544  Ha Noi, 02-2022 |

**PREFACE**

This project report is submitted in fulfillment of the requirements for the course Digital Design II of School of Electrical Engineering. We are required to make a project on “Neuron grid based on spiking neural network”. The basic objective behind carrying out this project requirement is to understand and apply the SNN theory to practical design.

In this report, we included various concepts, measurement step and simulation results on Modelsim and Vivado.

Through this progress of doing this project, we have come to enhance teamwork and problem-solving skills.

We are grateful to outstanding course instructor Nguyen Duc Minh. He was instrumental in preparing our class essential guidance and references to be capable of completing this assignment and has been helpful throughout. It's an honor to have completed this course under his direction, which aided in bringing out the best of our efforts to achieve our objective.

**TABLE OF CONTENTS**

[FIGURES i](#_Toc94826566)

[CHAPTER 2. C++ SIMULATION 3](#_Toc94826567)

[1.1 Generate input data 3](#_Toc94826568)

[1.1.1 Generate axon spikes information 3](#_Toc94826569)

[1.1.2 Generate neuron parameters 4](#_Toc94826570)

[1.2 Running simulation 5](#_Toc94826571)

[1.2.1 In 1 tick 6](#_Toc94826572)

[1.2.2 In many ticks 6](#_Toc94826573)

[CHAPTER 3. VERILOG SIMULATION 7](#_Toc94826574)

# FIGURES

[*Figure 1 Data flow* 2](#_Toc94826545)

[*Figure 2 Link to axon spikes files* 3](#_Toc94826546)

[*Figure 3 Functions to generate Axon Spikes in 1 tick and many ticks* 3](#_Toc94826547)

[*Figure 4 Link to neuron parameters file* 4](#_Toc94826548)

[*Figure 5: generate CSRAM function* 4](#_Toc94826549)

[*Figure 6 Function generateCSRAM() in main()* 5](#_Toc94826550)

[*Figure 7 Link to neuron instructions file and output file* 5](#_Toc94826551)

[*Figure 8 Function NeuronGrid() in main()* 6](#_Toc94826552)

[*Figure 9 Function ManyTicks() in main()* 6](#_Toc94826553)

[*Figure 10 Values need to be set before running Verilog simulation* 7](#_Toc94826554)

[*Figure 11 Successful Verilog simulation* 7](#_Toc94826555)

[*Figure 12 Error Verilog simulation* 7](#_Toc94826556)

**CHAPTER 1. OVERVIEW**

Code C++ (file neuron\_grid\_codeC.cpp):

* Generate text files that contain all the data in network. These files will be considered as the input file of simulation.
* The output files:
* Text files contain input data
* A text file contains output data from C++

Code Verilog:

* Use the data from the text file to simulate the operation of the Neuron Grid*.*
* The output file is a text file containing the simulation result.

Experiment flow:

Diagram

Description automatically generated

*Figure 1 Data flow*

As shown in Figure 1, both C++ and Verilog can take input from external text files containing data, but the C++ can itself generate text files containing data as input to Verilog and itself.

# CHAPTER 2. C++ SIMULATION



## Generate input data

C++ can create 3 types of data file:

* The file that contains all axon spikes information in 1 tick
* The file that contains all axon spikes information in many ticks
* The file that contains all the neuron parameters

### Generate axon spikes information

Link to the file that contains all axon spike information can be modified:



*Figure 2 Link to axon spikes files*

The 2 functions *generateAxonSpikes()* and *generateManyAxonSpikes(num\_tick)* need to be called at main() function before running the simulation. (num\_tick is the number of ticks in simulation)

Text

Description automatically generated

*Figure 3 Functions to generate Axon Spikes in 1 tick and many ticks*

Figure 3: The function *generateManyAxonSpikes()* which creates a text file with 16 lines, each line represents 1 tick and contains axon spikes data of 256 axons. Similarly, the function *generateAxonSpikes()* creates a text file with 1 line corresponding to 1 tick and contains axon spikes data of 256 axons.

These axon\_spikes informations are all randomly selected between 0 and 1.

### Generate neuron parameters

Link to the file that contains all the neuron parameters can be modified:



*Figure 4 Link to neuron parameters file*

In *generateCSRAM()* function, the generated text file has as many lines as the number of neurons (default is 256), each line is the parameters of each randomly generated neuron with a width that follows the RANC architecture (368 bits). The random interval can be changed in the *random(a, b)* function (where a is the minimum value and b is the maximum value in the range in decimal).

Graphical user interface, text

Description automatically generated

*Figure 5: generate CSRAM function*

NOTE:

* The synaptic connections value is generated only for the first 64 neurons and from axon 245th of each neuron because in the RANC architecture, potential and spike out information of a neuron are only considered after all 256 axons have been calculated, however the default range of *current\_potential* is only 9 signed bits => If there are too many synaptic connection or the weight is too high => The potential is integrated many time => The calculated potential value may exceed 8 bit that represent integer part => Wrong result. (Example: 25610 = 100000000­2 in unsigned = -25610 in signed)
* In fact, all the neuron parameters are generated based on a trained model, so this phenomenon may not happen

Text

Description automatically generated

*Figure 6 Function generateCSRAM() in main()*

## Running simulation

Before starting the simulation, make sure the following values are set properly:



*Figure 7 Link to neuron instructions file and output file*

These values are addresses of all needed files for the simulation:

* Output file
* Neuron instruction file: contains all the axon types information, it can be downloaded from

[RANC/hardware/src/simulations/memory\_files/vmm at master · UA-RCL/RANC · GitHub](https://github.com/UA-RCL/RANC/tree/master/hardware/src/simulations/memory_files/vmm)

### In 1 tick

Use function *NeuronGrid()* to start the simulation:

Text

Description automatically generated

*Figure 8 Function NeuronGrid() in main()*

After finish, a text file will be generated at *link\_to\_output* (*Figure 7*). This is the output of C++ simulation, this file contains 1 line with 256 bits, each bit represents an output of a neuron: 0 (there is no spike from that neuron) or 1 (there is a spike from that neuron)

### In many ticks

Use function *ManyTick(num\_tick)* to start simulation:

Text

Description automatically generated

*Figure 9 Function ManyTicks() in main()*

After finish, a text file will be generated at *link\_to\_output* (*Figure 7*). This is the output of C++ simulation, this file contains *num\_tick* line with 256 bits; each line is the output in a tick (line 1 is the output of the first tick, line 2 is the output of the second tick, etc.), each bit represents an output of a neuron: 0 (there is no spike from that neuron) or 1 (there is a spike from that neuron)

# CHAPTER 3. VERILOG SIMULATION

To run the Verilog simulation, in the file *tb\_neuron\_grid*, the number of ticks to be tested needs to be updated and the addresses of the following 6 files need to be created: file containing axon spikes in 1 tick, file containing all neuron parameter, file containing neuron instruction, file containing simulation results of C++, file containing axon spikes in many ticks and file containing simulation results of Verilog.

Text

Description automatically generated

*Figure 10 Values need to be set before running Verilog simulation*

After simulation, the output file of Verilog will be compared with the C++ output file. If they are the same, transcript window will display the message “Test pass without error”:

Text

Description automatically generated

*Figure 11 Successful Verilog simulation*

Else, it will notify where the errors are:

Table

Description automatically generated with medium confidence

*Figure 12 Error Verilog simulation*