Spiking Neural Network implementation for IoT Devices

Sondos Shahin 1200166 [1], *Ahmed Sayyad 1190855* [2]

1200166@student.birzeit.edu[1], 1190855@student.birzeit.edu[2]

Abstract

Spiking Neural Networks (SNN) have optimal characteristic for hardware implementation. Neurons communicate together using spikes, which in terms of logic resources means a single bit. This feature reduces the logic required for the device. This paper shows an implementation of a spiking neural network architecture, using 32nm process suitable for IoT devices.

1. Introduction

In order to make the internet of things (IoT) hardware more intelligent, artificial intelligence (AI) is being employed in the IoT devices. Spiking neural network (SNN) is one of the used AI algorithms in IoT hardware. SNN proposes high energy efficiency, and achieve better performance and lower area and power consumption compared to other implementations, since the spiking neural network-based systems can perform parallel signal processing [1].

Several 32nm processes have been introduces by different manufacturing companies. This technology address different applications depending on the manufacturer, but all share some improvements such as increasing performance and decreasing power consumption [2].

2. Literature Review

The implementation of a spiking neural network system includes modeling of the dynamics of individual neurons, and the interconnections between these neurons. The interactions between neurons, called synaptic, are modeled to transmit signals between the neurons. Each synaptic weight determines the strength of the connection [4].

The behavior of neurons including their membrane potential, synaptic interactions and spike generation can be described mathematically using ordinary differential equations or difference equations. The neurons would generate spikes after their membrane potential reaches a specified threshold, and these spikes are sent to another neuron through the synaptic connection [4].

One of the most interesting properties of SNN are the delays involved in the system. Both weights and delays can be adjusted to provide adequate behavior of the network. This flexibility led to developing different applications using SNN, including image recognition and operation, retinal coding and image processing and filtering. In addition to its wide usage in the hardware field, in robotics for movement control and other fields [3].

After the implementation of SNN in hardware succeeded, implementing SNN in real-time and embedded systems and IoT devices became the new goal because of the improvements the SNN would add to such devices [3].

The 32nm process technology propose smaller cell sizes than other technologies, leading to higher integration density, and better power and area optimization. These features make the use of 32nm process a good choice to implement the SNN in the IoT hardware [2].

3. System Architecture

The architecture designed (SNN) system designed for the IoT devices contains multiple layers of synapses and neurons, hierarchically organized to optimize energy and space consumption. To create an efficient and scalable system, the architecture leverages digital design principles

3.1. Overview

Spiking Neural Networks (SNNs) “stand as a transformative approach in the realm of artificial intelligence (AI), offering solutions to the pressing challenges faced by traditional AI models, including large language models (LLMs). Here we delve into the technical and conceptual advancements brought about by SNNs, highlighting their potential to revolutionize AI through energy efficiency, advanced hardware compatibility, temporal processing capabilities, and alignment with biological neural networks”. [4]

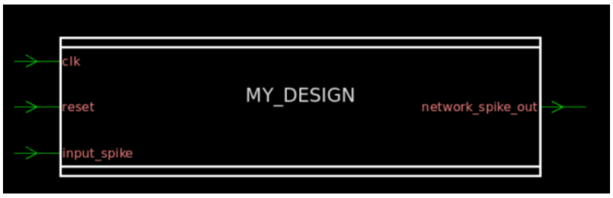


Figure 1: Block Diagram

3.2. Components

Synapse Module:

Based on the synaptic weights, the communication between neurons and the modulation of spike transmission will be handled by the synapse. Each synapse will increase its internal spike counter upon receiving a pre-synaptic spike, and once the counter reaches the synaptic weight, it will generate a post-synaptic spike.

Neuron Module:

Each neuron in the network when its membrane potential reaches a certain threshold, it will combine incoming impulses and generate an output spike.

Network Controller:

To manage the time and the coordination for the spikes all over the network the network controller is used for. It makes sure that the spikes spreaded correctly through the layers in the modules (Synapse and Neuron).

4. Design Methodology

To implement a SNN system for the IoT devices the design methodology should focuses to get a practical efficient low power solution. This may make a process that involves a lot of stages from design, verification to the optimizing.

4.1. Challenges of Spiking Neural Networks

SNN is considered to be the most powerful comparing to any current generation of the NNs. However, there is two main serious challenges that is need to be solved:

1) "The lack of the learning method developed specifically for the SNN training." [5]
2) Hardware: the working with the SNN requires a lot of solving differential equation so this can be considered to be computationally expensive, "so it is not possible to effectively work locally without having specialized hardware" [5]

This is making it gets a lot of requirements which include low power consumption, high integration that fit within small form factors, scalability and real time processing capability.

4.2. Conceptual & hardware Design

For the conceptual design "the basic building block of an SNN is the spiking neuron, which models the behavior of a biological neuron that fires an action potential, or spike when it receives enough input. SNNs also use mathematical models to describe the connectivity between neurons". [6]
Network Architecture: where is the network is structured to layers that each layer having neurons and synapses and as mentioned before the architecture is hierarchical.

For the hardware design Verilog which is a hardware language where used the design was parameterized that allows for easy modification.

4.3. Optimization & Verification

4.3.1. Power Optimization:

- Clock Gating**:** To reduce dynamic power consumption the Unused neurons and synapses got their clocks gated off.
- Simplified Neuron Model**:** for reducing both area and power usage. The integrate and fire model will minimize the logic that is required for each neuron

4.3.2. Area Optimization:

- High Integration Density: using of 32nm process technology "enables a high integration density, fitting more neurons and synapses into a smaller area".
- Hierarchical Design: The modular approach simplifies the design and allows for efficient use of space.

4.3.3. Functional Verification:

Using testbench that simulates inputs from different spikes and for correctness it will check the outputs

4.3.4. Performance Evaluation:

Based on the metrics like the spike processing latency the performance of the SNN will be evaluated that also beside the power consumption and area usage which are measured and compared to the design requirements to make sure that the SNN will be equal to the goal specified

5. Power Analysis

Power analysis is very important to the design of the SNN for the IoT devices, to make sure that power consumption of the SNN meets the low power requirements that is necessary to an efficient operation it will be analyzed.

5.1. Power Measurement

Using simulation tools the power estimation is performed. For the simulation setup to simulate typical usage scenarios the testbench will create realistic input spike pattern. Tools like primetime PX is used to estimate power consumption according to the simulation results.

"post-synthesis power estimation is conducted. This provides a more accurate measure of power consumption by considering the actual implementation of the design".

5.2. Power Optimization Techniques

1) Clock Gating: to reduce the dynamic power consumption it will make sure that no unnecessary switching activity will happen

2) Voltage Scaling: it reduces the voltage level when it in the same time maintaining performance so the dynamic and static power will both be reduced

3) Low-Power Design Techniques: MTCMOS and power gating both can be used to reduce the static power consumption

6. Performance Evaluation

6.1. Evaluation Metrics

Processing Latency: the lower latency the faster processing which is important to real time applications.

Spike Throughput: higher throughput gives the better performance which is the number of spikes processed per unit of time.

Accuracy: which is measured by comparing outputs with the expected results.

6.2. Testing Methodology

To measure the latency, throughput, and accuracy the SNN will be simulated with different input spikes.

- Test Scenarios: Include different spike rates, input patterns, and network sizes.
- Tools**:** Simulation tools provide detailed reports on timing, resource usage, and functional correctness.

7. Integration with IoT devices

For the Integration to the SNN with IoT devices this contain and involves many steps to make sure the efficiency and the compatibility to the real-world applications. This integration needs a handling input/output signals.

“The SNN needs to interact with various sensors and actuators commonly used in IoT devices. Sensors provide input spikes based on environmental changes, such as temperature or motion, while actuators receive output spikes to perform actions such as activating motors or alarms. Ensuring seamless data transfer between the SNN and these components is crucial, which is achieved through standard communication protocols. Common protocols such as I2C, SPI, and UART are used to exchange data between SNN and other IoT components, ensuring compatibility and efficient communication. Simple data formats, such as binary or JSON, are used to facilitate this exchange”.

To make sure that the SNN works within the power constraints to the IoT devices it should have an Efficient power management . Dynamic voltage scaling to reduce the power consumption and sleep modes to power down the idle components and reducing the energy uses.

8. Verification

8.1. Simulation-Based Verification

Testbenches that were running by the functional simulation that creates Verilog models of the SNN will generate input spike pattern then checking the response to verify both the correctness and the functionality from the SNN. Making sure testing all the possible scenarios tested. Time analysis is also crucial simulation part, the tools making a verify where that the signals to meet the setup as required and holding times avoiding any timing violations.

8.2. Hardware Testing

On the FPGA the SNN is implemented, evaluating the real-world performance for it. This will be involving real time testing in order to measure the performance metrics like latency.

Checking the resource utilization to make sure it has an efficient implementation.

8.3. Benchmark Validation

Making a test to the SNN facing the industry standard benchmarks making sure that it’s meeting the accuracy and performance that was expected.

MNIST is a benchmark where the datasets providing a basis to compare it with other AI algorithms showing the SNN strength to the IoT devices

All of this work together to make sure that the verification process to the SNN implementation is efficient suitable and reliable and meets the demands to the real-world application.

9. Results and Discussion

The first step in the physical design process was the floor planning, which determines the physical placements of the design components to optimize performance and power. And the power planning that provides power for all cells.

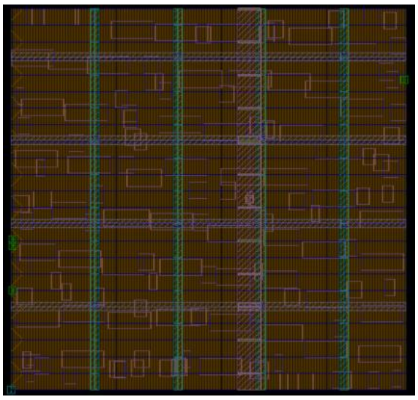


Figure 2-floor planning and power planning

Then in the placement step, the design components were placed on the die in a way that optimizes area, power and timing.

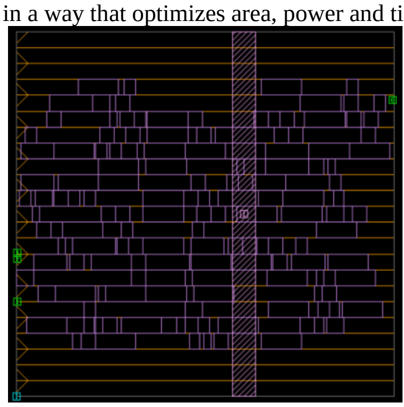
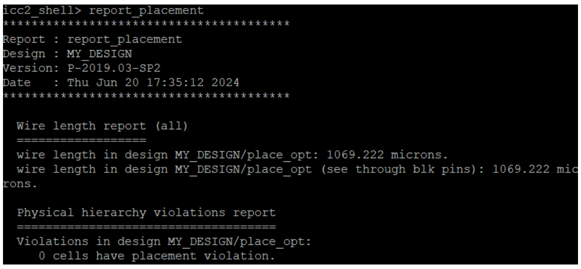


Figure 3- placement



```
icc2_shell> report_placement
****************************************
Report : report_placement
Design : MY_DESIGN
Version: P-2019.03-SP2
Date   : Thu Jun 20 17:35:12 2024
****************************************

  Wire length report (all)
  ==================
  wire length in design MY_DESIGN/place_opt: 1069.222 microns.
  wire length in design MY_DESIGN/place_opt (see through blk pins): 1069.222 mic
rons.

  Physical hierarchy violations report
  ====================================
  Violations in design MY_DESIGN/place_opt:
     0 cells have placement violation.
```

Figure 4- placement report

After placement was done successfully, clock synthesis and timing analysis were performed on the design.

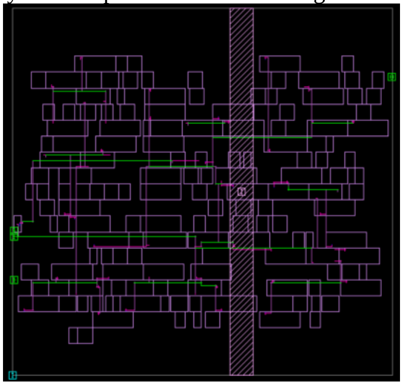
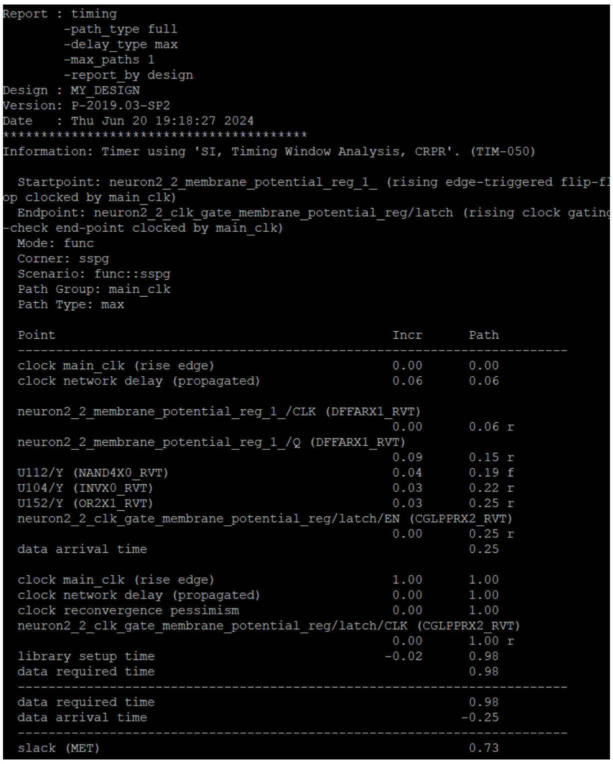


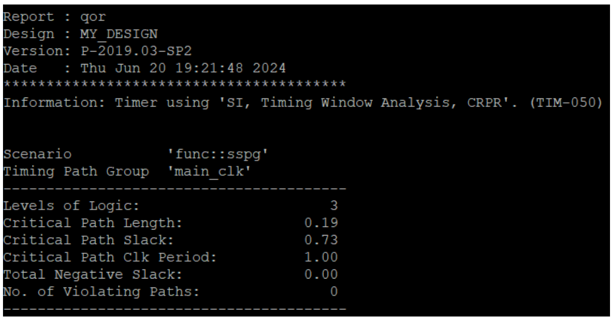
Figure 5- clocking

Many timing analysis reports were performed to make sure the design has no errors, in addition to area reports.



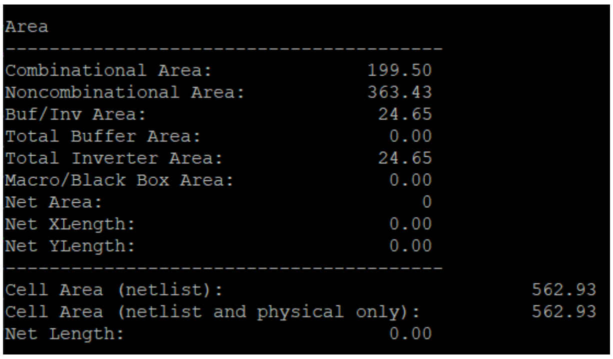
```
Report : timing
        -path_type full
        -delay_type max
        -max_paths 1
        -report_by design
Design : MY_DESIGN
Version: P-2019.03-SP2
Date   : Thu Jun 20 19:18:27 2024
****************************************
Information: Timer using 'SI, Timing Window Analysis, CRPR'. (TIM-050)

  Startpoint: neuron2_2_membrane_potential_reg_1_ (rising edge-triggered flip-fl
op clocked by main_clk)
  Endpoint: neuron2_2_clk_gate_membrane_potential_reg/latch (rising clock gating
-check end-point clocked by main_clk)
  Mode: func
  Corner: sspg
  Scenario: func::sspg
  Path Group: main_clk
  Path Type: max

  Point                                                 Incr      Path
  ------------------------------------------------------------------------
  clock main_clk (rise edge)                            0.00      0.00
  clock network delay (propagated)                      0.06      0.06

  neuron2_2_membrane_potential_reg_1_/CLK (DFFARX1_RVT)
                                                        0.00      0.06 r
  neuron2_2_membrane_potential_reg_1_/Q (DFFARX1_RVT)
                                                        0.09      0.15 r
  U112/Y (NAND4X0_RVT)                                  0.04      0.19 f
  U104/Y (INVX0_RVT)                                    0.03      0.22 r
  U152/Y (OR2X1_RVT)                                    0.03      0.25 r
  neuron2_2_clk_gate_membrane_potential_reg/latch/EN (CGLPPRX2_RVT)
                                                        0.00      0.25 r
  data arrival time                                               0.25

  clock main_clk (rise edge)                            1.00      1.00
  clock network delay (propagated)                      0.00      1.00
  clock reconvergence pessimism                         0.00      1.00
  neuron2_2_clk_gate_membrane_potential_reg/latch/CLK (CGLPPRX2_RVT)
                                                        0.00      1.00 r
  library setup time                                   -0.02      0.98
  data required time                                              0.98
  ------------------------------------------------------------------------
  data required time                                              0.98
  data arrival time                                              -0.25
  ------------------------------------------------------------------------
  slack (MET)                                                     0.73
```

Figure 6- timing report



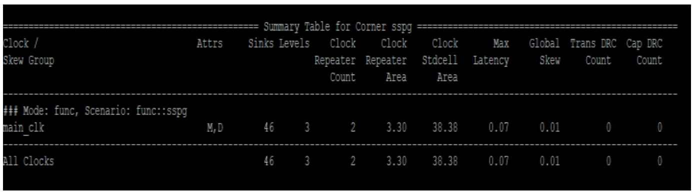
```
Report : qor
Design : MY_DESIGN
Version: P-2019.03-SP2
Date   : Thu Jun 20 19:21:48 2024
****************************************
Information: Timer using 'SI, Timing Window Analysis, CRPR'. (TIM-050)


Scenario           'func::sspg'
Timing Path Group  'main_clk'
----------------------------------------
Levels of Logic:                      3
Critical Path Length:              0.19
Critical Path Slack:               0.73
Critical Path Clk Period:          1.00
Total Negative Slack:              0.00
No. of Violating Paths:               0
----------------------------------------
```

Figure 7- qor report



```
Area
----------------------------------------
Combinational Area:               199.50
Noncombinational Area:            363.43
Buf/Inv Area:                      24.65
Total Buffer Area:                  0.00
Total Inverter Area:               24.65
Macro/Black Box Area:               0.00
Net Area:                              0
Net XLength:                        0.00
Net YLength:                        0.00
----------------------------------------
Cell Area (netlist):                              562.93
Cell Area (netlist and physical only):            562.93
Net Length:                         0.00
```

Figure 8- area report



```
=============================================== Summary Table for Corner sspg ================================================
Clock /                          Attrs    Sinks Levels    Clock    Clock    Clock      Max   Global Trans DRC Cap DRC
Skew Group                                              Repeater Repeater Stdcell  Latency    Skew     Count   Count
                                                           Count     Area    Area
------------------------------------------------------------------------------------------------------------------------------
### Mode: func, Scenario: func::sspg
main_clk                           M,D       46      3        2     3.30    38.38     0.07     0.01        0       0
------------------------------------------------------------------------------------------------------------------------------
All Clocks                                   46      3        2     3.30    38.38     0.07     0.01        0       0
```

Figure 9- clock report

Then, routing was performed, which establish an electrical connection between the design components, while adhering to the design constrains of timing, chip area and power consumption.

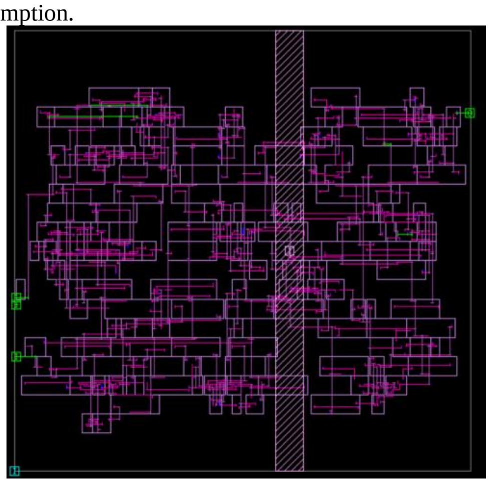
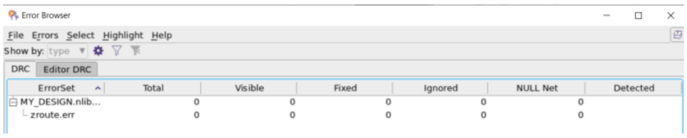


Figure 10- routing



ErrorSet	Total	Visible	Fixed	Ignored	NULL Net	Detected
MY_DESIGN.nlib...	0	0	0	0	0	
zroute.err	0	0	0	0	0	

Figure 11- routing report

After the routing, the physical design was finished, but still need to be verified. EDA playground tool was used for the verification of the design.

The focus in verification of our design was on the first two layers.

The two neurons in the first layer have a threshold equals 3, so each neuron would wait until its membrane potential reached the threshold and would then send a spike to the neurons in the next layer.

The clock would be turned of for any neuron that has a membrane potential equals zero, in order to reduce the power consumption of the design.

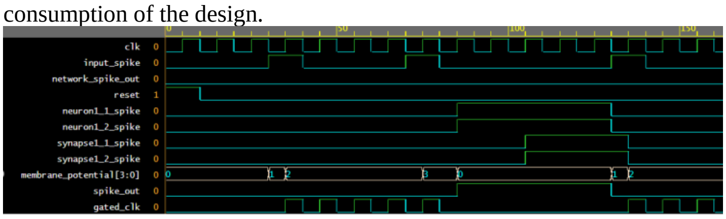


Figure 12- layer 1 testing

For the second layer, the same testing technique was followed, a neuron in the second layer has a threshold equals 2, while the other has a threshold equals 3, both neurons would receive spiked from the previous layer, and would wait until their membrane potential reaches their threshold to send a spike for their following layer.

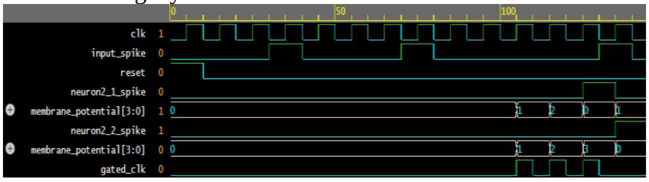


Figure 13- layer 2 testing

10. Conclusion

In this paper, we presented the implementation of a Spiking Neural Network (SNN) architecture using a 32nm process, tailored for Internet of Things (IoT) devices. The unique characteristics of SNNs, such as spike-based communication and parallel signal processing, make them highly suitable for energy-efficient and low-power applications. Our design demonstrates significant improvements in power and area optimization, leveraging the advanced capabilities of the 32nm process technology.

We have outlined the detailed architecture of the SNN, including neuron and synapse modules, and provided a comprehensive design methodology focusing on scalability and efficiency. The integration of the SNN with IoT devices highlights the potential of this technology to enhance the intelligence and responsiveness of IoT systems, making them more capable of real-time processing and decision-making.

Through rigorous verification involving both simulation-based and hardware testing, we ensured that our SNN implementation meets the required specifications and performs reliably in various scenarios. Performance evaluation metrics, such as processing latency, spike throughput, accuracy, and resource utilization, were used to validate the effectiveness of our design. The successful integration and optimization of the SNN for IoT applications demonstrate the feasibility and advantages of this approach. Future work could focus on further reducing power consumption, increasing the scalability of the network, and exploring additional real-world IoT applications to fully realize the potential of SNNs in enhancing IoT technology.

11. References

[1] [Online]. Available: https://www.mdpi.com/1424-8220/23/14/6275#B1-sensors-23-06275.

[2] [Online]. Available: https://hal.science/hal-03324299/document.

[3] [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667015404562/pdf?.

[4] [Online]. Available: https://medium.com/@theagipodcast/spiking-neural-network-architectures-e6983ff481c2.

[5] [Online]. Available: https://cnvrg.io/spiking-neural-networks/.

[6] [Online]. Available: https://www.frontiersin.org/journals/computational-neuroscience/articles/10.3389/fncom.2023.1215824/full.