LIF-IF-Tutorial: Modeling Leaky Integrate-and-Fire (LIF) and Integrate-and-Fire (IF) Neurons

- Written by: Khanh N. Dang (khanh [at] u-aizu [dot] ac [dot] jp)

- Last update: Dec 20, 2024

This tutorial demonstrates how to model LIF and IF neurons using Verilog HDL and Python. It also includes testbenches and visualizations for better understanding.

---

1. Introduction

Spiking Neural Networks (SNNs) are biologically-inspired models that mimic the behavior of neural networks using discrete spikes to transmit information. Central to SNNs are Leaky Integrate-and-Fire (LIF) and Integrate-and-Fire (IF) neurons, which simulate the membrane potential dynamics of biological neurons. These networks are at the forefront of innovations in neuromorphic computing and low-power AI systems, offering energy-efficient solutions for next-generation artificial intelligence

---

2. Mathematical Modeling

Note: the equations of IF and LIF can be varied. The following are two types of equation.

Integrate-and-Fire (IF)

- Membrane potential equation: $V(t+1) = V(t) + I(t)$
- **Key behaviors**:
  - Spikes when $V(t) \geq V_{\text{th}}$.
  - Resets after spiking: $V(t) = V_{reset}$.

Leaky Integrate-and-Fire (LIF)

- Membrane potential equation: $V(t+1) = \alpha V(t) + I(t)$
  - $\alpha < 1$: Leakage factor that models decay.

---

3. Python Implementation

Code for Simulation and Visualization

You can find the Python script at sw/LIF_neuron.py. To make IF model, please adjust the leak_factor to 1.0.

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters
time_steps = 100
threshold = 1.0
leak_factor = 0.9
input_current = np.random.uniform(0.1, 0.2, time_steps)
membrane_potential = np.zeros(time_steps)
prefire_membrane_potential = np.zeros(time_steps)
spikes = np.zeros(time_steps)

# Simulation
for t in range(1, time_steps):
    membrane_potential[t] = leak_factor * membrane_potential[t-1] + input_current[t]
    prefire_membrane_potential[t] = membrane_potential[t]
    if membrane_potential[t] >= threshold:
        spikes[t] = 1
        membrane_potential[t] = 0  # Reset after spike

# Plot
plt.figure(figsize=(10, 5))
plt.subplot(2, 1, 1)
plt.plot(membrane_potential, label="Membrane Potential")
plt.plot(prefire_membrane_potential, label="Membrane Potential (Before Firing)")
plt.axhline(y=threshold, color='r', linestyle='--', label="Threshold")
plt.legend()
plt.title("LIF Neuron Simulation")
plt.subplot(2, 1, 2)
plt.stem(spikes, label="Spikes", use_line_collection=True)
plt.legend()
plt.show()
```

Illustration

The plot of LIF neuron is shown below:

4. Verilog HDL Implementation

LIF Neuron Code

This source code is available at hw/LIF_Neuron.v

```verilog
module LIF_Neuron (
    input clk,
    input reset,
    input [7:0] input_current,
```

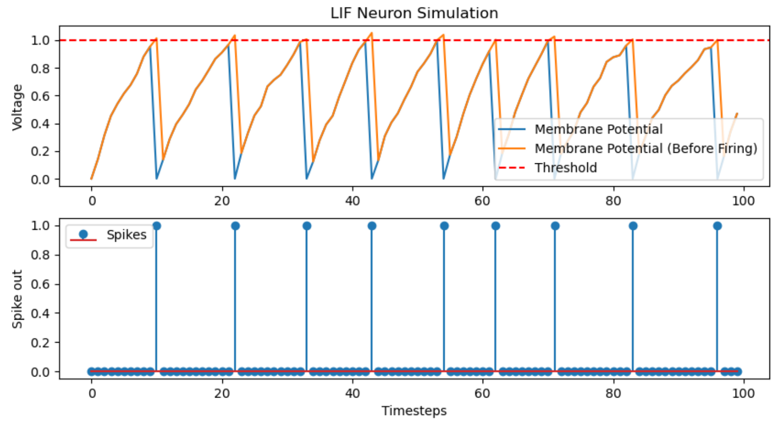


Figure 1: LIF neuron simulation result.

```
    output reg spike
);
    reg [7:0] membrane_potential;
    parameter THRESHOLD = 8'd128;
    parameter LEAK = 8'd1;

    always @(posedge clk or posedge reset) begin
        if (reset) begin
            membrane_potential <= 8'd0;
            spike <= 1'b0;
        end else begin
            if (membrane_potential >= THRESHOLD) begin
                spike <= 1'b1;
                membrane_potential <= 8'd0; // Reset
            end else begin
                spike <= 1'b0;
                membrane_potential <= (membrane_potential >> 1) + input_current; // Leaky i
            end
        end
    end
endmodule
```

LIF Neuron Testbench Code

This source code is available at hw/LIF_Neuron_tb.v

```
`timescale 1ns/1ps
```

```verilog
module LIF_Neuron_tb;

    // Testbench signals
    reg clk;
    reg reset;
    reg [7:0] input_current;
    wire spike;

    // Instantiate the DUT (Device Under Test)
    LIF_Neuron dut (
        .clk(clk),
        .reset(reset),
        .input_current(input_current),
        .spike(spike)
    );

    // Clock generation
    always #5 clk = ~clk; // 10 ns clock period

    // Testbench sequence
    initial begin
        // Enable waveform generation
        $dumpfile("waveform.vcd");
        $dumpvars(0, LIF_Neuron_tb);

        // Initialize signals
        clk = 0;
        reset = 1;
        input_current = 0;

        // Apply reset
        #10 reset = 0;

        // Test case 1: Low input, no spike
        input_current = 8'd10;
        #100;

        // Test case 2: High input, trigger spike
        input_current = 8'd50;
        #100;

        // Test case 3: Reset during operation
        reset = 1;
        #10 reset = 0;
        #100;
```

```
        // Test case 4: Alternating inputs
        input_current = 8'd30;
        #50;
        input_current = 8'd0;
        #50;

        // End simulation
        $finish;
    end

    // Monitor signals
    initial begin
        $monitor($time, " Reset=%b, Input=%d, Membrane Potential=%d, Spike=%b",
                 reset, input_current, dut.membrane_potential, spike);
    end

endmodule
```

Simulation and waveform

Before running the simulation, please check out some explanations about iverilog and gtkwave here.

To run the simulation, please follow this instructions.

Install iverilog and gtkwave If your machine does not have these tools, please install them from :

- iverlog: https://steveicarus.github.io/iverilog/
- gtkwave: https://gtkwave.sourceforge.net/

Note: if you are fimilar with hardware design, you can use different tools to simulate.

Run the simulation First, check the current work folder (`hw`) and the availability of the tools:

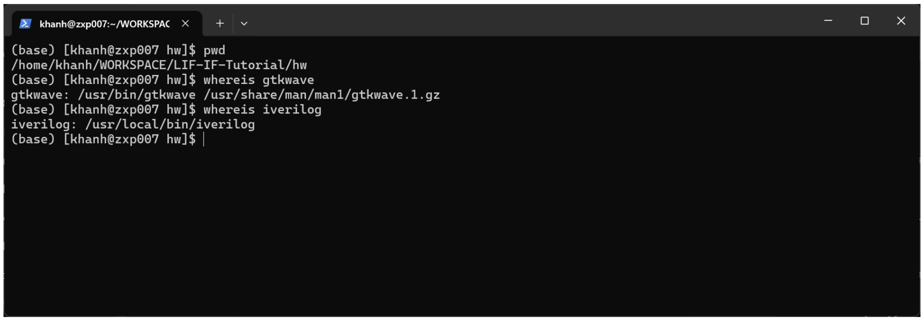
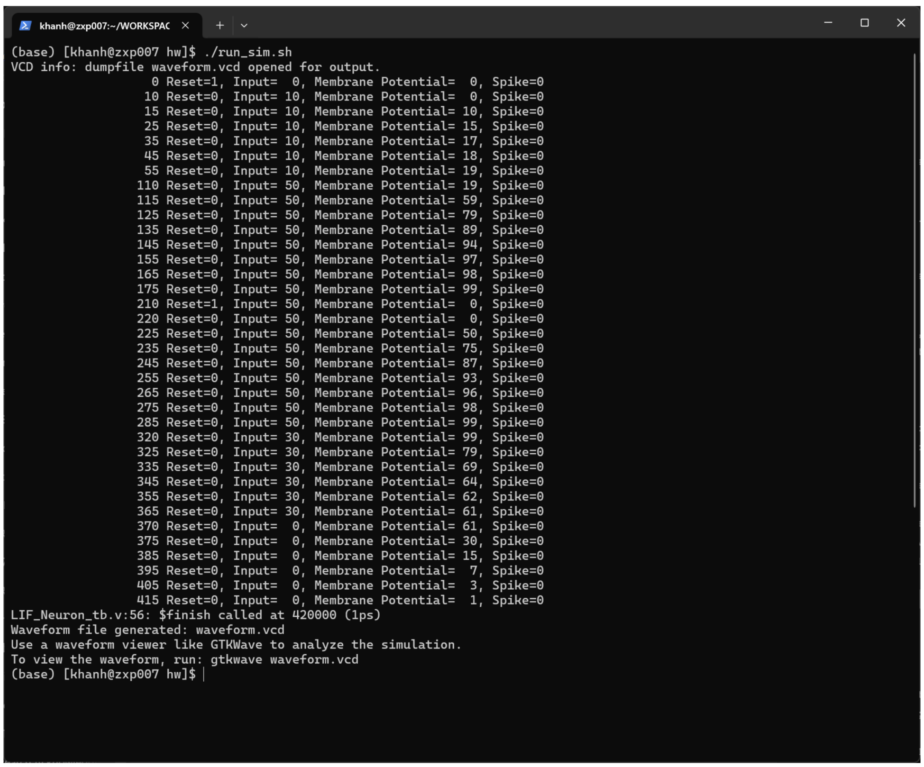


Figure 2: Check the availability of the tools.

Then, run the `run_sim.sh` script:



```
(base) [khanh@zxp007 hw]$ ./run_sim.sh
VCD info: dumpfile waveform.vcd opened for output.
                   0 Reset=1, Input=  0, Membrane Potential=  0, Spike=0
                  10 Reset=0, Input= 10, Membrane Potential=  0, Spike=0
                  15 Reset=0, Input= 10, Membrane Potential= 10, Spike=0
                  25 Reset=0, Input= 10, Membrane Potential= 15, Spike=0
                  35 Reset=0, Input= 10, Membrane Potential= 17, Spike=0
                  45 Reset=0, Input= 10, Membrane Potential= 18, Spike=0
                  55 Reset=0, Input= 10, Membrane Potential= 19, Spike=0
                 110 Reset=0, Input= 50, Membrane Potential= 19, Spike=0
                 115 Reset=0, Input= 50, Membrane Potential= 59, Spike=0
                 125 Reset=0, Input= 50, Membrane Potential= 79, Spike=0
                 135 Reset=0, Input= 50, Membrane Potential= 89, Spike=0
                 145 Reset=0, Input= 50, Membrane Potential= 94, Spike=0
                 155 Reset=0, Input= 50, Membrane Potential= 97, Spike=0
                 165 Reset=0, Input= 50, Membrane Potential= 98, Spike=0
                 175 Reset=0, Input= 50, Membrane Potential= 99, Spike=0
                 210 Reset=1, Input= 50, Membrane Potential=  0, Spike=0
                 220 Reset=0, Input= 50, Membrane Potential=  0, Spike=0
                 225 Reset=0, Input= 50, Membrane Potential= 50, Spike=0
                 235 Reset=0, Input= 50, Membrane Potential= 75, Spike=0
                 245 Reset=0, Input= 50, Membrane Potential= 87, Spike=0
                 255 Reset=0, Input= 50, Membrane Potential= 93, Spike=0
                 265 Reset=0, Input= 50, Membrane Potential= 96, Spike=0
                 275 Reset=0, Input= 50, Membrane Potential= 98, Spike=0
                 285 Reset=0, Input= 50, Membrane Potential= 99, Spike=0
                 320 Reset=0, Input= 30, Membrane Potential= 99, Spike=0
                 325 Reset=0, Input= 30, Membrane Potential= 79, Spike=0
                 335 Reset=0, Input= 30, Membrane Potential= 69, Spike=0
                 345 Reset=0, Input= 30, Membrane Potential= 64, Spike=0
                 355 Reset=0, Input= 30, Membrane Potential= 62, Spike=0
                 365 Reset=0, Input= 30, Membrane Potential= 61, Spike=0
                 370 Reset=0, Input=  0, Membrane Potential= 61, Spike=0
                 375 Reset=0, Input=  0, Membrane Potential= 30, Spike=0
                 385 Reset=0, Input=  0, Membrane Potential= 15, Spike=0
                 395 Reset=0, Input=  0, Membrane Potential=  7, Spike=0
                 405 Reset=0, Input=  0, Membrane Potential=  3, Spike=0
                 415 Reset=0, Input=  0, Membrane Potential=  1, Spike=0
LIF_Neuron_tb.v:56: $finish called at 420000 (1ps)
Waveform file generated: waveform.vcd
Use a waveform viewer like GTKWave to analyze the simulation.
To view the waveform, run: gtkwave waveform.vcd
(base) [khanh@zxp007 hw]$
```

Figure 3: Run the simulation script.

Finally, run `gtkwave waveform.vcd` to view the waveform.

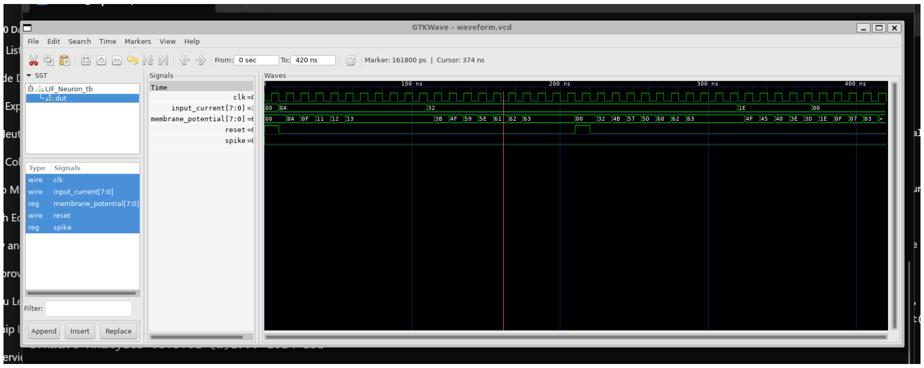


Figure 4: Open the waveform and view the signals.

5. Exercises

Modification of LIF model

Using the previous design, you need to modify the LIF model into this equation

- Membrane potential equation: $V(t+1) = V(t) + I(t) - \lambda$
- **Key behaviors**:
  - Spikes when $V(t) \geq V_{\text{th}}$.
  - Resets after spiking: : $V(t) = V(t) - V_{reset}$.

You need to modify both Verilog and Python codes and make a report to **show the correctness of your new model**.