Signal Processing and Renewable Energy
March 2019, (pp. 35-42)
ISSN: 2588-7327
eISSN: 2588-7335

Hardware Implementation of LIF and HH Spiking Neuronal Models

Pooya Soleimani Abhari[1], Farhad Razaghian[1*]

[1] Electrical Engineering Department, South Tehran Branch, Islamic Azad University, Tehran, Iran.

Received: 12-Oct-2018, Revised: 5-Dec-2018, Accepted: 9-Dec-2018.

Abstract

This paper presents a hardware implementation of both Hodgkin-Huxley (HH) and Leaky Integrate and Fire (LIF) spiking neuronal models. FPGA is used as digital platform due to flexibility and reconfigureability. The proposed neural models are simulated by MatLab and the results are compared with the HDL software's output in order to evaluate the design. Simple architecture uses two counters and a comparator used as the main part of leaky Integrate and Fire model. For the Hodgkin and Huxley model a Look Up Table based structure is utilized. Although it consumes large amount of area, it results more reasonable propagation delay time hence higher operating frequency. The proposed architectures are evaluated on Stratix III device using Quartus II simulator. Maximum operating frequency of 583 MHz (limited to 500 MHz due to the device port rate) and 76 MHz are achieved for the LIF and HH architectures respectively.

Keywords: Spiking neural network, LIF Model, HH Model, FPGA implementation.

1. INTRODUCTION

Biological important researches have illustrated the neurons are connected to each other. Neural networks of the brain comprise of three main parts which are functionally separated. Dendrite acts as the input device, collects signals from other neurons and transfers them to the soma. Soma asthe main processing unit has nonlinear functionality. When voltage of the soma exceeds threshold value, the output signal is generated and delivered to other neurons by axons resemble output device. Connection between two consecutive neurons is called a synapse. A neuron which is sending signal refers to the pre-synaptic cell and a neuron which is receiving refers to the postsynaptic cell. The neuronal signals consist of short duration electrical pulses that could be observed by placing a fine measuring system close to the soma or the axon of a neuron. The pulses have 1-2 ms duration and 100 mV amplitude and also called action potentials or spikes. A series of consecutive action potentials emerges from a single neuron is called a spike train which may occur at regular or irregular intervals. Since there are similar shape spike trains, the form of the action

*Corresponding Author's Email: razaghian@azad.ac.ir

potential has no information, instead the number and the time intervals of spikes contains significant information. The postsynaptic neuron can be affected by receiving spikes which cause a potential change that can be recorded with an intracellular electrode which measures the potential difference $u(t)$ between the inside of the cell and its surroundings. This potential difference is called the membrane potential. When there is not any input spike, the neuron is at rest means a constant membrane potential. When a spike arrives, the potential changes and finally return back to the resting potential [1]. There are several methods of simulation and implementation of spiking neural networks (SNN). Some of them are based on software like MatLab or C programming language. As the processing speed is an essential issue and should be considered, various methods of hardware implementation of SNNs have proposed like ASIC, DSP and FPGA. Higher performance designs could be reached utilizing FPGAs because they have design flexibility and can be reprogrammed. Also, they are well suited for parallel processing applications versus fixed ASIC designs and sequential DSP's structure. Reviewing literates demonstrates that FPGAs are widely used as a platform for SNNs implementations. A hippocampus inspired spiking neural network was implemented on an FPGA by Mokhtar, et al [2]. Izhikevich spiking neurons in a pipelined manner on FPGA was presented by Rice et al [3]. Grass et al considered another FPGA based approach for high speed simulation of conductance [4]. Pourhaji et al [5] introduced a hardware based approach to simulate action potential of large numbers of somas within a biological neural network, this paper showed multiple processors can work in parallel to increase processing power as required. Bonabi et al used CORDIC algorithm as an area reduction technique for implementing HH spiking neural model [6]. A comparison of analog and digital circuit implementations of SNN as a function of area and speed done by Joubert et al depicts digital designs consumes more area [7]. Zaman Farsa et al [8] utilized FPGA based SNN model for function approximation.

2. SPIKING NEURAL NETWORKS – NEURAL MODELS

2.1. Leaky Integrate and Fire Model

Each neuron contains an intracellular which is separated from surrounding liquid with a membrane. This is modeled simply by a capacitor. Also, there is a resistive path from inner region of the cell to the surrounding that is called leak path and could be modeled with a resistance. All spikes arrive to the neuron increase the membrane potential (potential difference between intracellular and surrounding due to the variation of ion's density). When the membrane potential reaches the specified value called threshold, spike shall be generated. Fig 1 illustrates the simplest spiking neural model called Leaky Integrate and Fire (LIF). Obviously, Eq. 1 is related to IF model where V is the membrane potential, R is leak path resistance and C is capacitive path. I(t) is used to model the input spikes. In Fig. 1, $\alpha(t-t_j^{(f)})$ represents spike arrives at time $t^{(f)}$ from neuron j and $\delta(t-t_i^{(f)})$ exhibits i'th neuron's spike generation at time $t^{(f)}$.

$$C\frac{dV}{dt} = -\frac{V}{R} + I(t) \qquad (1)$$

Assume two presynaptic neurons j = 1, 2, which both send spikes toward the postsynaptic neuron i. Neuron j = 1 fires spikes at $t_1^{(1)}$, $t_1^{(2)}$, similarly neuron j =2 fires at $t_2^{(1)}$, $t_2^{(2)}$. Each spike evokes a postsynaptic potential ε_{i1} or ε_{i2}, respectively.

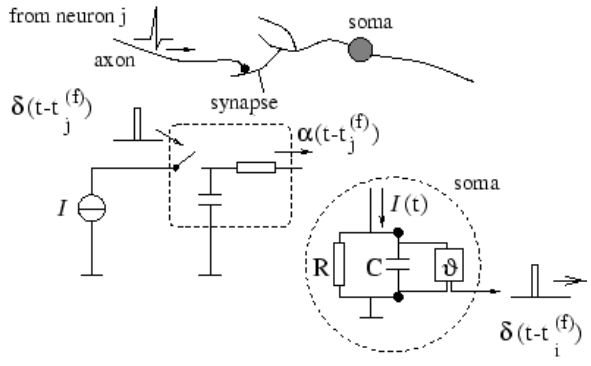

Fig. 1. Integrate and Fire model of the neuron.

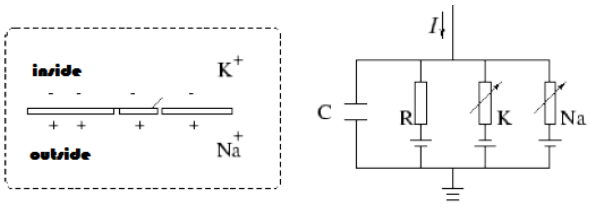

Fig. 2. HH model of SNN [1].

Total change of the potential can be calculated by the sum of the individual post synaptic potentials (see Eq. 2).

$$vi(t) = \sum_j \sum_f \varepsilon_{ij}(t - t_j^{(f)}) + v_{rest} \quad (2)$$

2.2. Hodgkin-Huxley Model of SNN

An experiment on the giant axon of the squid has been done by Hodgkin and Huxley hence three different types of ion current, sodium, potassium and leak current (due to the cl^- ions) has been introduced [1]. Ion current is caused by ions concentration's difference between interior and surrounding of the neuron. Fig. 2 shows the schematic of HH model.

Translating of the above mentioned to mathematical equation results are written in Eqs. (3)-(6). Eq. 3 is the conservation of electric charge on a piece of membrane implies that the applied current $I(t)$ may be split in a capacitive current I_C which charges the capacitor C and further components I_k which pass through the ion channels. In the standard Hodgkin-Huxley model there are only three types of channel: a sodium channel with index Na, a potassium channel with index K and an unspecific leakage channel with resistance R. $\sum_k I_K$ in Eq. 4 shows total leakage current, all channels have their specific channel conductance. The leakage channel is voltage independent whereas both of the remainder (g_{Na} ,g_K) have voltage and time dependent channel conductance. m, n and h are the parameters explain the probability that a channel is open. m and n are used as controlling parameters of Na channel while h controls the K channel. E_{Na} ,E_L and E_L is the reversal potential (due to the ions concentration differences) that are modeled with the batteries on each corresponding branches. All conductance and resting potentials are empirical where Table 1 demonstrates the original value assuming zero for resting potential. To get the values which are valid today, all should be shifted by E_{rest}=-65mV. Empirical variables α and β ar- e voltage dependent that are used for m, n, h calculation (Table 2). Differential Eqs. (6)-(8) are used to calculate m, n and h which are called gating variables [1].

$$I(t) = I + \sum_k I_k(t) \quad (3)$$

$$\frac{Cdv}{dt} = I(t) - \sum_k I_k(t) \quad (4)$$

$$\sum_k I_k(t) = g_{Na} m^3 h(v - E_{Na}) + g_K n^4 (v - E_K) + g_L (v - E_L) \quad (5)$$

$$\frac{dm}{dt} = \alpha_m(v)(1 - m) \quad (6)$$

$$\frac{dn}{dt} = \alpha_n(v)(1 - n) - \beta_n(v)n \quad (7)$$

$$\frac{dh}{dt} = \alpha_h(v)(1 - h) - \beta_h(v)h \quad (8)$$

Table 1. ***Conductance and reversal potential [1].***

X	E_x (mv)	g_x (mS/cm^2)
Na	115	120
K	-12	36
L	10.6	0.3

Table 2. α, β ***as empirical parameters [1].***

X	$\alpha_x(\frac{v}{mV})$	$\beta_x(\frac{v}{mV})$
N	$\frac{0.1-0.01v}{\exp(1-0.1v)-1}$	$0.125\exp(-\frac{v}{80})$
M	$\frac{2.5-0.1v}{\exp(2.5-0.1u)-1}$	$4\exp(-\frac{v}{18})$
H	$0.07\exp(-\frac{v}{20})$	$\frac{1}{\exp(3-0.1v)+1}$

3. DIGITAL IMPLEMENTATION OF THE LIF MODEL

Simple digital implementation of LIF model can be performed according to Euler's equation (Eq.9). Obviously, this can be modeled with two separated modules. One of them is the synapse block and another is the neuron's core block. The synapse block is a pulse generator whose output pulse duration refers to the synaptic weight and is triggered by the input spike. Larger synaptic weight causes wider pulse that is fed to the core block and has more effect on neuron's core. To implement the synapse block, a 8-bits counter is utilized that starts counting from initial value and force the block's output to be at high when it receives the input spike. If the counter's value reaches the synaptic weight, the counter reset its value and pulls down the output (logic 0).

The neuron's core block which is used to model behavior of the neuron in presence of the leak age and the input spike effect also contains 8bits counter. Another input

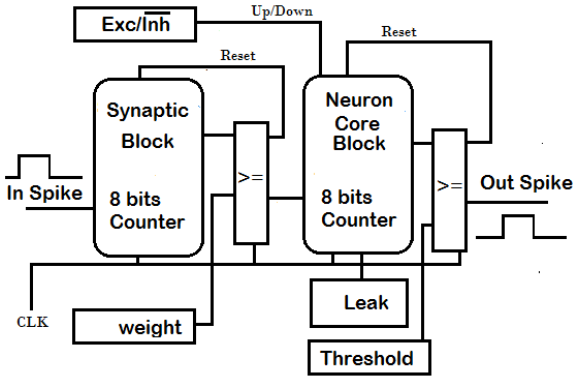

Fig. 3. LIF Model digital implementation block.

parameter called $Exc/\overline{Inh}$ determines the input spike is excitatory or inhibitory. Core's counter starts counting as the output spike of the synapse module is logic 1, when $Exc/\overline{Inh}$ is also logic 1 counter's value is increased otherwise down counting is performed. The pulse width which is received from the synapse module specifies the amount of the value should be added to or subtracted from counter's value. To model leaky current when there is no input pulse to the membrane counter (i.e. no counting due to the input spikes) the membrane counter's value is decremented with an adjustable step called leak period. Finally, in order to implement the membrane function, a comparator is used. As the membrane potential (core counter) reaches the firing threshold, the output spike will be generated and the counter is reset to its initial condition. Fig. 3 clearly demonstrates the functionality of the implemented modules [7].

$$v(i+1) = v(i) + \frac{\Delta t}{C}\left(-\frac{1}{R}v(i) + I_{Leak}\right) \quad (9)$$

4. DIGITAL IMPLEMENTATION OF THE HH MODEL (PROPOSED MODEL)

As discussed before, the membrane potential for HH model can be computed using Eq .5.

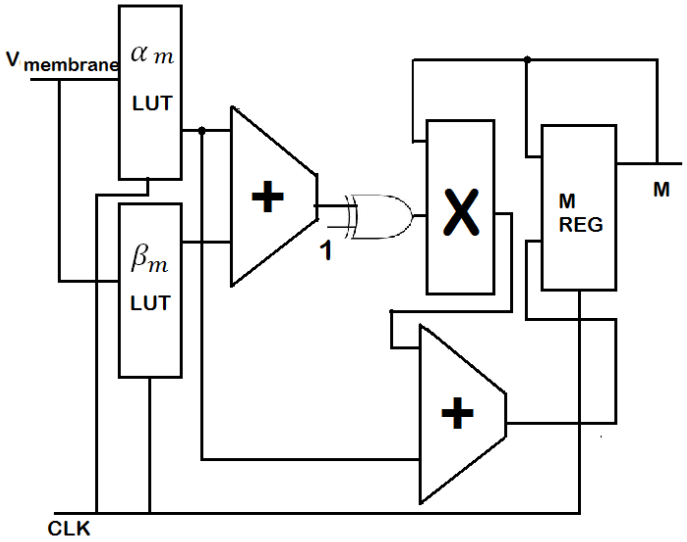

Fig. 4. m parameter calculation block.

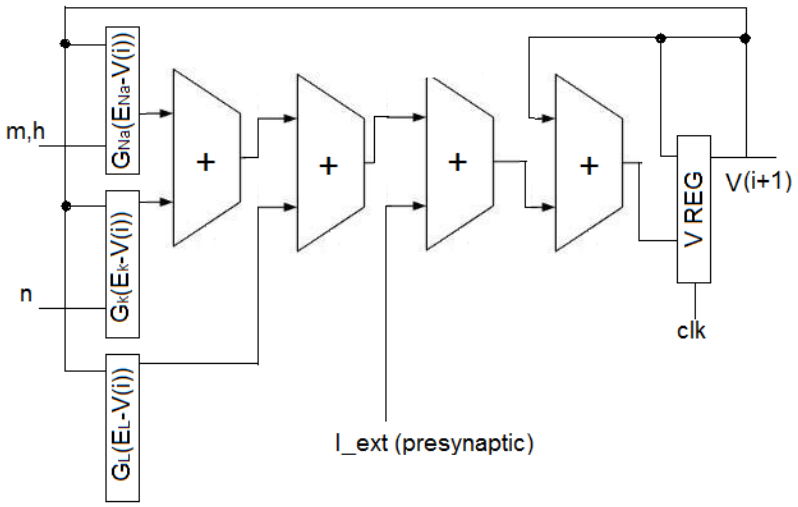

Fig. 5. Membrane potential calculation block.

There are several control parameters which should be pre-computed to solve the equation. In order to calculate m, n and h parameters, at first α *and* β must be estimated using Table 2. These variables only depend on the membrane potential value. A ROM is used to store α *and* β with the address line of the membrane potential (each membrane potential has its own specific α and β values).

According to Eqs. (6)-(8), clearly Euler's method is used to determine m, n and h parameters by recursive operations and remarking on initial conditions. Eq. 10 shows how Euler's method is used to calculate m, n and h. Fig. 4 illustrates the block diagram of logical circuit that is used to calculate m, the same circuits may be used for n and h parameters. Having these parameters, the variables G_{Na}, G_K and G_L (Eq. 11) are used to calculate the membrane potential recurrently applying Euler's method (see Eq. 12).

$$m(i+1) = m(i) + \Delta t(-(\alpha+\beta)m(i) + \alpha) \quad (10)$$

$$n(i+1) = n(i) + \Delta t(-(\alpha+\beta)n(i) + \alpha)$$

$$h(i+1) = h(i) + \Delta t(-(\alpha+\beta)h(i) + \alpha)$$

$$G_{Na} = g_{Na} * m^3 * h \ , \quad G_K = g_K * n^4,$$

$$G_L = g_L \quad (11)$$

$$v(i+1) = v(i) + \frac{\Delta t}{C}(I_{ext}(i) - G_{Na}(v(i) - E_{Na}) - G_K(v(i) - E_K) - G_L(v(i) - E_L)) \quad (12)$$

5. SIMULATION RESULTS

Simulation of the above-mentioned architectures for both LIF and HH model of spiking neural networks are performed on Stratix III Altera FPGA on Quartus II platform. To see the output signal flow modelsim simulator is used. Maximum operating frequency of 584 MHz (500 MHz due to the port rate) for LIF model and 76 MHz for HH model are achieved. LIF model has no memory requirement in turn HH uses 6 k byte of memory. LIF consumes total logic element of 59 and HH includes 324 combinational ALUs, 32 memory ALUTs and 179 dedicated registers, HH proposed architecture also contains 44 18-bit DSP blocks. Fig. 7 and Fig. 8 show the output of the simulation for the LIF model where resting potential and after spike potential are assumed zero. These models are also evaluated using MatLab (Fig. 6). Proposed architecture for HH single neuron model is simulated by setting the resting potential zero and an external signal is used to model the input spikes. For comparison with previous works, a CORDIC and LUT based architecture by Bonabi et al [9] is proposed on Spartan III, Xilinx device with maximum frequency of 37 MHz and 99 DSP blocks and 23512 number of 4 input LUTs. Clearly, this work has improved the performance in order to higher operating frequency and lower area consumption.

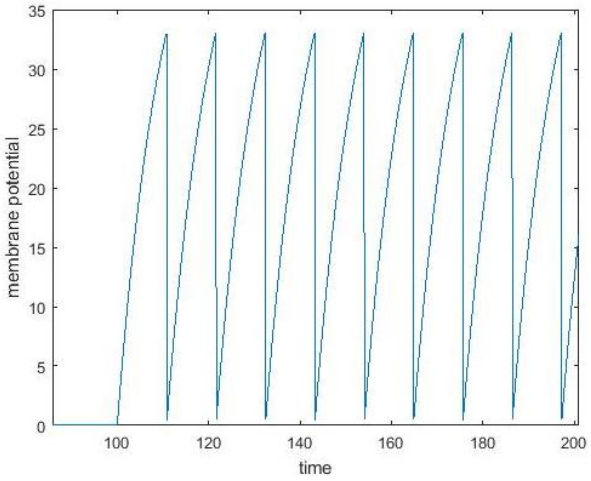

Fig. 6. LIF _MatLab simulation digital model.

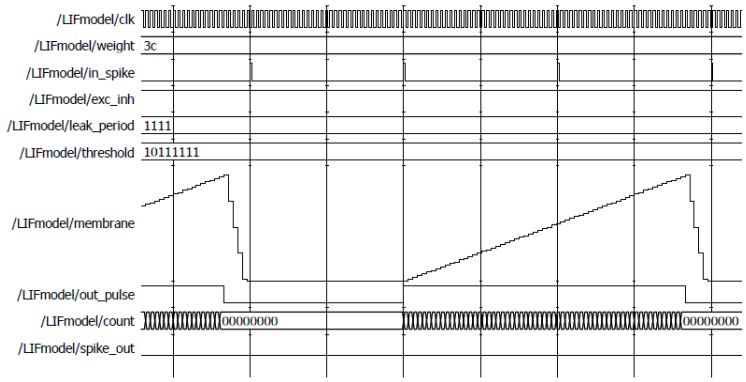

Fig.7. LIF model_ modelsim gate level simulation.

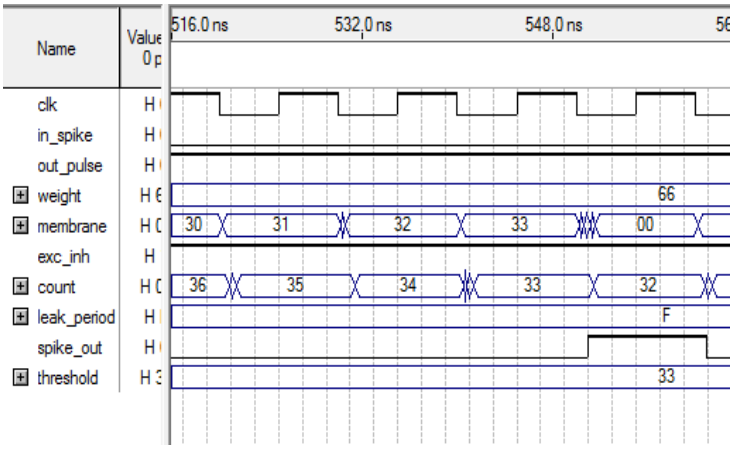

Fig. 8. LIF model Quartus II silmulation.

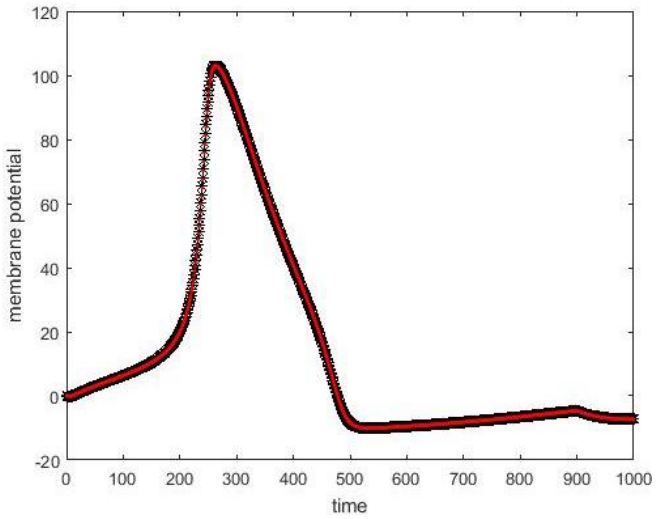

Fig. 9. HH model simulation (proposed digital algorithm(black), typical HH model (red)).

Table 3. ***Device utilization summary.***

Spiking Model	LIF	HH
f_{max} (MHz)	500	76
#ALUT	42	324
#Memory ALUT	0	32
#Registers	20	179
# Memory bits	0	49152
# DSP blocks	0	44

6. CONCLUSION

This paper presents two various logic models for implementation of the LIF and the HH model of the spiking neural networks. Single neuron model is considered to estimate the logic validity. These models are also inspected using MatLab and the result depicts the presented logical model could meet the goal. Maximum operating frequency of the device is achieved for the LIF design, whereas for the HH, f_{max} is restricted to 76 MHz. Optimization of the logical architectures may be a good idea for further works. There are various methods of reducing design area especially for HH model, like utilization of CORDIC algorithm instead of ROM, but this restricts the operating frequency. A hybrid architecture that uses both ROM and COORDIC to reach higher performance may be desired for future works.

REFERENCES

[1] W. Gerstner and W. M. KistlerSpiking Neuron Models, Cambridge University Press August 2002.

[2] M. Mokhtar, D.M. Halliday, A.M.Tyrrell. Hippocampus-Inspired Spiking Neural Network on FPGA, Proceedings of the 8th International Conference on Evolvable Systems, ICES '08. Springer-Verlag, Berlin, Heidelberg, pp. 362–371. 2008.

[3] K.L.Rice, M.A. Bhuiyan, T.M. Taha, C.N. Vutsinas, M.C. Smith, FPGA implementation of Izhikevich spiking neural networks for character recognition, Reconfigurable Computing and FPGAs. International Conference On. pp. 451–456. 2009

[4] E.L. Graas, E.A. Brown, R.H. Lee, An FPGA-based approach to high-speed simulation of conductance-based neuron models. Neuroinformatics 2, 417–435. 2004.

[5] P. Pourhaj, D.H. Teng, FPGA based pipelined architecture for action potential simulation in biological neural systems, Electrical and Computer Engineering (CCECE), 2010 23rd Canadian Conference On. pp. 1–4. 2010.

[6] S.Y .Bonabi, H. Asgharian, R. Bakhtiari, S. Safari and M. N. Ahmadabadi, FPGA Implementation of Hodgkin-Huxley Neuron Model, In Proceedings of the 4th International Joint Conference on Computational Intelligence, pp.522-528. 2012.

[7] A.Joubert, B. Belhadj, O. Temam, R. H´eliot. Hardware spiking neurons design: analog or digital?, The 2012 International Joint Conference on Neural Networks (IJCNN), IEEE, PP.1-5. 2012.

[8] E. Zaman Farsa · S. Nazari · M. Gholami, Function approximation by hardware spiking neural network, J Comput Electron, springer, pp. 707-716. 2015.