CAD Design of Digital VLSI Systems I Final Project: Digital Implementations of Biologically Plausible Neurons

EN.520.491
Ralph Etienne-Cummings, Ph.D
retienne@jhu.edu
Dept. of Electrical & Computer Engineering
The Johns Hopkins University

Final Presentation on Wednesday, December 15th from 2PM - 5PM EST

For your final project, you will form groups of two and demonstrate one of five different neural models in software and hardware. Your task is to comprehensively understand your chosen neuron model using both software & hardware simulations. Please follow the instructions carefully:

1 Backgrounds of Different Neural Models

There are a variety of neuron models that vary based on their similarity to biology and implementation complexity. Below is a figure illustrating the relationship between biological plausibility & computational efficiency of several neural models.

For this course, you will choose one of the following neuron models to execute in software and digitally in hardware simulations:

- Leaky Integrate-and-Fire with Spike-rate Adapation
- Izhikevich Spiking Model
- FitzHugh-Nagumo
- Mihalaş-Niebur

Most of these models are summarized in Izhikavech's paper (found **here**) published in 2004. Please thorougly observe the neuro-computational properties of biological spiking neurons such that you can replicate similar behaviors in hardware and software simulations.

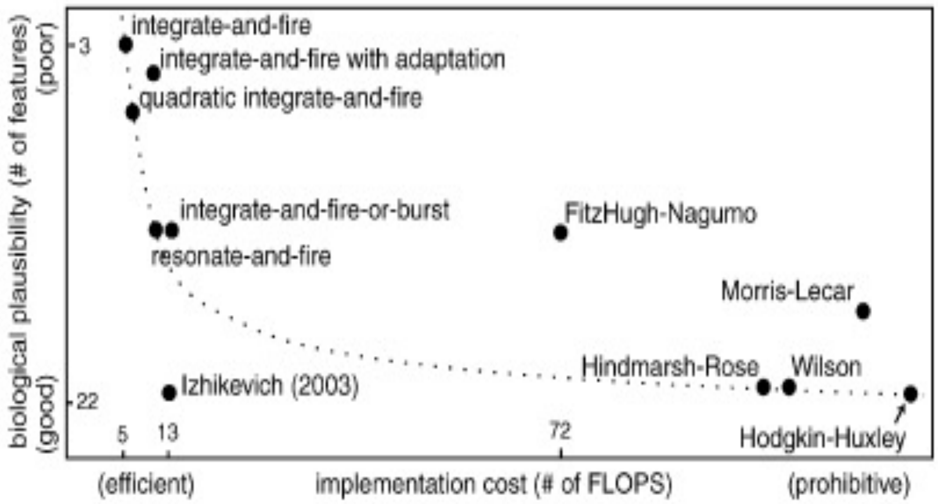

Figure 1: Trade-off between biological plausibility (with respect to a number of features known to exhibited by neurons) and implementation cost in FLOPS per 1 ms simulation of the model (from: Izhikevich 2004).

NOTE: Please refer to Euler's method to linearize your differential equation such that you can assess each model numerically.

1.1 LIF with Spike-rate Adaptation (SRA)

Probably the most popular neuron model because of its simplicity, the leaky integrate-and-fire neuron can be modeled using a membrane potential, an input current, a leakage term, and linear parameters used to achieve spiking behavior. Here, we ask you to implement an amended version that adjusts update rules based on spike-rate adaptation:

$$v' = I + a - bv + g(d - v)$$

$$g' = \frac{-g}{\tau}$$

$$\text{if } v \geq v_{th}, \qquad \text{then } \begin{cases} v \longleftarrow c \\ g \leftarrow d \end{cases}$$

where v is the membrane potential, I is in the input current, and a, b, d, and v_{th} are the parameters. g is the update function and τ is the time reference (constant). These dynamical equations model the adaptive nature of a neuron to a static stimulii.

1.2 Izhikevich Spiking Model

$$v' = 0.04v^2 + 5v + 140 - u + I$$
$$u' = a(bv - u)$$
$$\text{if } v \geq +30\text{mV}, \qquad \text{then} \begin{cases} v \longleftarrow c \\ u \longleftarrow u + d \end{cases}$$

where v represents the membrane potential of the neuron and u represents the membrane recovery variable. v and u are dimensionless variables. I is the injected DC currents and a, b, c, and d are dimensionless parameters.

1.3 FitzHugh-Nagumo (FN)

$$v' = v - v^3/3 - u + I$$
$$u' = \varepsilon(v + a - bu)$$

where I corresponds to excitation and a & b are the controlling parameters. u is the linear membrane recovery variable and ε is the normalizing modulation variable. This model is known as a simpler version of Hodgkin-Huxley and for emulating the current signal observed in a living organism's excitable cells.

1.4 Mihalaş-Niebur (MN)

The MN Neuron Model is based on the dynamics between the membrane potential (v), internal interactions of currents (I_j), and an instantaneous threshold (Θ) which has a maximized dependence (θ_∞) on the membrane potential. C is the membrane capacitance. The currents are exponentially decaying such that the evolution of such threshold relies on the constant external current, I_e. As such, here is the evolution of the state variables:

$$I_j' = -k_j I_j; \quad j = 1, ..., N$$
$$v' = \frac{1}{C}\left(I_e + \sum\nolimits_j I_j - G(v - E_L)\right)$$
$$\Theta' = a\left(v - E_L\right) - b\left(\Theta - \Theta_\infty\right)$$

The set of update rules is as follows:

$$I_j \longleftarrow R_j \times I_j + A_j$$
$$v \longleftarrow v_r$$
$$\Theta \longleftarrow max(\Theta_r, \Theta),$$

where the parameters k_j, R_j, G, A_j, v_r (reset), E_L, and Θ_r (reset) can be chosen freely, subject only to the constraint $\Theta_r > v_r$. A_j and R_j can be set to 1, leading to an additive update. If they're set to 0, there is a constant update response (spike-induced). In this particular model, the adaptive threshold is updated continuously using and not only when a spike occurs.

2 Software Simulations - presentation on 12.02.21

Once you have chosen your neuron model, you will implement a numerical solution using MATLAB or Python. Using an analytical solution will not allow you to understand the timely behavior of the spiking modalities, so please reduce your equations to numerical representations. Please try to replicate as many spiking behaviors as you can.

3 Update on Hardware Simulations - presentation on 12.09.21

Once you have confirmed the behavior in a software simulation, please use Verilog or VHDL to do the same to demonstrate the hardware representation. Once you have simulated your model in an HDL, continue with synthesis and place & route to realize your model. This is imperative in analyzing your model, especially when comparing to other models.

When presenting your functional post-layout model, you will need to show a Figure of Merit (FOM) to distinguish you among your peers. The FOM is generally as follows:

$$FOM = \frac{FLOP/s}{Area \cdot Power} \tag{1}$$

4 Hardware vs Software Comparison - presentation on 12.15.21

4.1 BONUS: Application of Hardware Representation

Congratulations! At this point, you have implemented your neuron model in software and hardware. Using your work so far, you can now demonstrate an application of spiking neurons. Here are some suggestions:

- simple spiking classification task
- biomimetic central pattern generator
- illustrate long-term potentiation (LTP) & long-term depression (LTD) effects

5 Grading

5.1 Presentations

The grading will be based on the difficulty in project choice as well as a FOM to distinguish your group among your peers.

For each power-point presentation / update on your progress, you need to consistently include the following:

- figure & table numbers with clear descriptions
- FOM calculation (when necessary)
- relevant citations when necessary
- slide numbers for easy reference

Each group of 2 will give a 10 minute powerpoint presentation on their project detailing design decisions, challenges, implementation, and performance. A PDF version of the presentation must be turned in on Blackboard prior to class on the presentation date to be graded; thus, the slides should be detailed enough to provide a full picture of the work completed. The following is the timeline:

1. Software Simulation: Thursday, December 2nd from 4:30PM - 5:45PM EST
2. Hardware Implementation: Thursday, December 9th from 4:30PM - 5:45PM EST
3. Final Presentation: Wednesday, December 15th from 2PM - 5PM EST

N.B. All transistors in this project should be minimum sized transistors using the 45 nm process

5.2

Final Report Due Friday, Dec 17 at 11:59PM on Blackboard

A final report, written in the form of a journal paper **(found here)** is required for your project. The paper length is limited to 4 pages. The paper should look EXACTLY like an article from the IEEE Journal on Solid-State Circuits. Summarize your design, approach and results in a succinct, scientific manner. In you final report, please also include:

- A paragraph, titled Global Impacts of IC Design, that discusses modern microprocessor design, IC fabrication technologies and environmental impact of IC design, fabrication and electronics waste disposal.
- A paragraph, titled Engineering Ethics, on how you applied the IEEE Code of Ethics to your project planning and execution.

Background reading on Global Impacts and Engineering Ethics may be found on Blackboard.