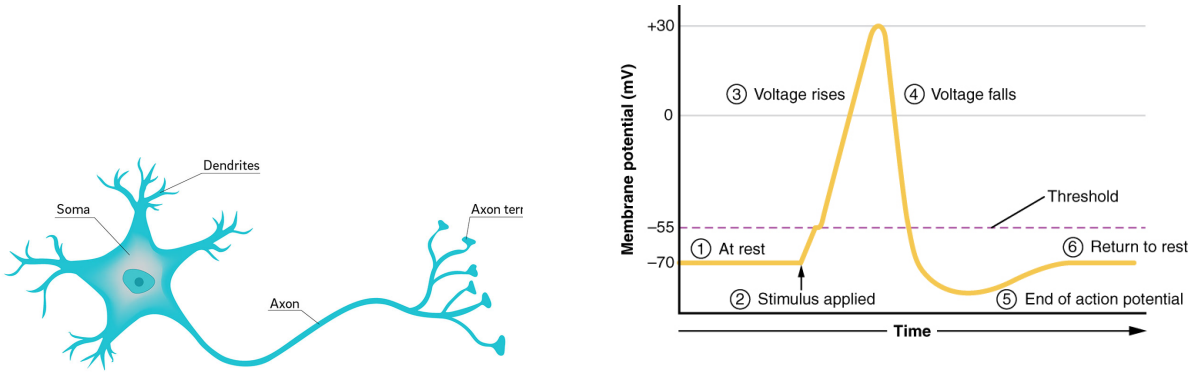
ECS4003 Advanced Computer Engineering
Laboratory & Coursework

The laboratory for this module addresses the design, analysis and optimisation of an example accelerator on FPGA; this is intended as a representation of the kinds of design issues faced across accelerator devices. It relies heavily on the use of HDL Coder, a toolbox in MATLAB which allows the behaviour of a function to be described using a dataflow model, from which Hardware Description Language (HDL) code can be generated to allow the function to be realised on FPGA.

Modelling Brains

Neuromorphic computer architectures attempt to model the behaviour of the human brain by constructing networks of neurons. Unlike the neurons used in artifical neural network (such as those used in deep learning), these neurons are *spiking*, and their networks known as Spiking Neural Networks (SNNs). The general structure of a neuron and its is shown in Fig. 1.



(a) Neuron Structure

(b) Behaviour

Figure 1: Models of Neurons

The *dendrites* act as inputs to the main cell body or *soma*. They transport streams of electrical current, in the form of potentials, from connected neurons. These accumulate an action potential in the soma. When the voltage reaches a critical value, the soma discharges, its internal voltage and a potential spike is output of the *axon* to neighbouring neurons.
An example model of this behaviour is the Izhikevich neuron, where the internal potential v is updated to its next value v' by a stream of incoming voltages on the dendrites, according to:

$$\tilde{v} = v + \left[0.04v^2 + 5v - u + J + 140\right] \tag{1}$$
$$\tilde{u} = u + \left[a\left(bv - u\right)\right] \tag{2}$$

$$v' = \begin{cases} -65 & \text{if } \tilde{v} > 30 \\ \tilde{v} & \text{otherwise} \end{cases} \tag{3}$$

$$u' = \begin{cases} \tilde{u} + d & \text{if } \tilde{v} > 30 \\ \tilde{u} & \text{otherwise} \end{cases} \tag{4}$$

Part I - Thinking, Slowly [**100 marks**]

a) Using Simulink, realise an Izhikevich neuron as per (1) - (4). Take the following coefficient values:

- $a = 0.02$
- $b = 0.2$
- $d = 8$
- initial value of $v = -65$
- initial value of $u = 0$

[**20 marks**]

b) Adapt your Izhikevich neuron model from a) to use *only* two-input add, multiply, subtract and switch component. Verify the spiking behaviour of the neuron. Save and submit the simulation as *izhikevichTest.slx* [**10 marks**]

c) Develop timed two-input multiply, add, subtract and switch components. Set their latencies (computation times) to the values defined by your student number.

d) Using the computation times from c), determine the iteration interval (II), latency and throughput of your neuron [**25 marks**]

e) Modify your neuron model so that its behaviour when using the timed components from c) matches that from b). Save and submit your simulation as *timedIzhikevichTest.slx*. [**20 marks**]

Submission
The three items above will be assessed via the following:

- For each of b) & e) a short record of the models, tests and results used (as appropriate).
- A derivation of II from d) and a description of the implications on the II on the behaviour of the neuron and the manner in which it should be stimulated.
- The *izhikevichTest.slx* and *timedIzhikevichTest.slx* Simulink models.

Part II - Thinking, Roughly

This task uses Xilinx System Generator to recast your model into one which can be realised on FPGA and which takes accounts of practical implications of the type of arithmetic used and the performance and cost of the resulting FPGA accelerator. System Generator harnesses Simulink, extending with Xilinx blocksets which allow modelling of physical components used to realise a design on FPGA. The following may be a useful guide as you get used to System Generator:

`https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_1/ug897-vivado-sysgen-user.pdf`

a) Translate your *timed* Izhikevich model using System Generator to one using only Xilinx Blockset components (assume double-precision floating-point arithmetic). Verify proper operation via simulation using System Generator. You can assume the initial value for $v = 0$ [**10 marks**]

b) Characterise the variation in accuracy of your neuron with floating-point wordsize. [**10 marks**]

c) Translate your System Generator model to one which uses two's-complement fixed-point arithmetic. Verify correct behaviour via simulation in System Generator. Repeat accuracy analysis for fixed-point arithmetic. [**20 marks**]

d) Using System Generator, derive HDL for your floating and fixed point designs. Implement these on Xilinx FPGA and record LUT, DSP48E, FF and BRAM costs. Compare and contrast the relative FPGA resource costs of your design. [**20 marks**]

e) Compare the performance and cost of your FPGA designs. Recommend optimal floating and fixed-point word sizes to employ, and a 'best' overall solution. [**40 marks**]

Submission
Your submission should consist of a report alongside two System Generator models. Please note that the primary method of assessment is your report, and you should be careful to demonstrate that you have satisfied the criteria for each of a; - e, including documenting models, behaviours, cost, accuracies and any other important detail.

- *Report*: This is the assessed element; you should be careful to demonstrate that you have achieved the outcomes for each assessed element in your report; specifically, this should include
  - Floating-point model, behaviour and quantitative analysis of variation in accuracy.
  - Fixed-point model, behaviour and quantitative analysis of variation in accuracy.
  - Quantified implementation cost variation on Xilinx FPGA.
  - Quantitative, justified analysis of accuracy/cost trade-off and choice of variant.
- *izhikevichTestFloat.slx*: double-precision floating-point model.
- *izhikevichTestFixed.slx*: two's-complement fixed-point model.

Please note that the models submitted are for verification purposes only and will not accrue any marks.