UCSB ECE 274 – Winter 2025

Assignment 4: Training and Testing a Small Neural Network with IF Neurons and STDP

Total Points: 120

Due at 5pm, Thursday, March. 13 (no late submission)

1. Background

In this assignment, you will build a small neural network using Verilog with IF (Integrated-and-Fire) neurons for image recognition. You will train this small network using STDP (Spike Timing Dependent Plasticity) with provided training samples. After training the network, you will use the testing dataset to check whether the network is well functioning.

1.1. Neural and Synaptic Models

You will use the discretized zero-th order synaptic model just like in the previous assignment (but there is no leaky parameter at this time):

$$V_i[t_k] = V_i[t_{k-1}] + K_{syn} \sum_{j=1}^{M} w_{ji} \cdot S_j[t_{k-1}]$$

where V_i is the membrane potential of neuron *i*, *M* is the number of pre-synaptic neurons connected to neuron *i*, K_{syn} is the synaptic weight parameter, w_{ji} is the synaptic weight between neurons *j* and *i* (neuron j is the pre-synaptic neuron connected to neuron *i*), and S_j is the indicator function that indicates whether neuron j fired (S_j will be 1 if neuron j fired and otherwise 0).

In this assignment, use the parameters given below:

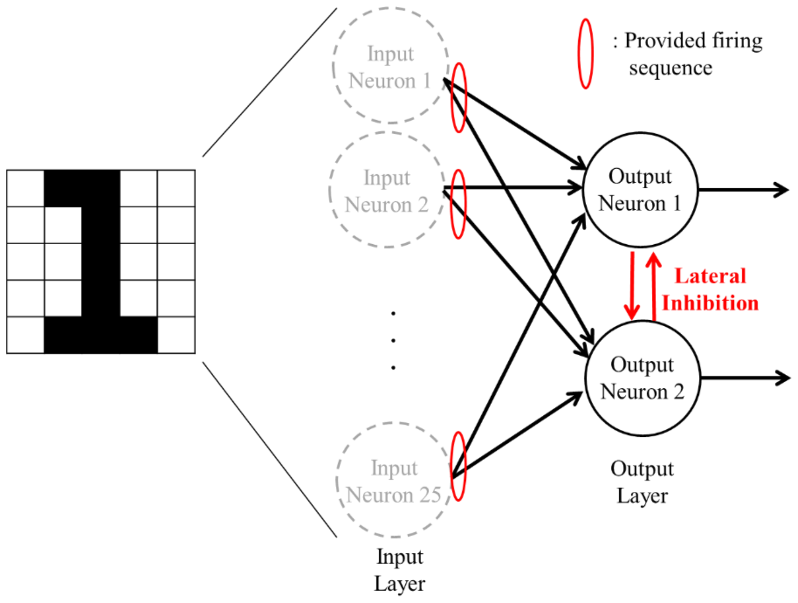
$V_i[t_0] = V_{rest}$(*resting membrane potential*) = 6, K_{syn} = 1, V_θ(*threshold voltage*) = 65.

(All parameters are normalized and ready-to-use)

** Note that the lowest possible membrane potential is constrained to V_{rest} at all times. Also, there is no leaky parameter in this assignment.

**If the membrane potential exceeds the threshold voltage at a rising clock edge, it will generate an output spike immediately. Please make sure that the membrane potential is maintained for that clock cycle and only reset the membrane potential to the resting potential at the next rising clock edge. (Wait for the next rising clock edge, i.e. membrane voltage reset will happen at the next clock cycle. The same shall be assumed for a reset by lateral inhibition discussed later.)

Fig. 1 Overall architecture of the design



1.2. Two-Layer Spiking Neural Network

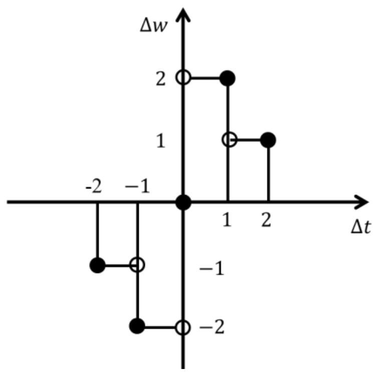
The network under consideration consists of two layers (Input, Output) as shown in Fig.1. The 'input layer' consists of 25 neurons and each of the neuron represents one pixel of the input image. The 'output layer' consists of two neurons, which shall fire at a higher level of activity under input digits '0' and '1', respectively. The output spikes of the neurons in the 'input layer' are given, as specified later.

In the output layer, '**lateral inhibition**' between the two neurons works as follows: 1) if only one output neuron fires, it will reset the other output neuron by setting the membrane potential of the other output neuron to the resting membrane potential V_{rest}; 2) If both output neurons fire at the same time, they do not laterally inhibit each other, i.e., the effect of lateral inhibition is non-existent. You must properly implement the above lateral inhibition behavior for the output neurons.

1.3. STDP Learning Rule

In this assignment, you will use the simplified version of STDP curve which is shown in Fig. 2. There are only 4 possible values (i.e. 0, 1, 2, 3) for the weight, which are be $00_{(2)}$, $01_{(2)}$, $10_{(2)}$, $11_{(2)}$ for hardware implementation. Please make sure to properly limit weight values for correct simulation, i.e. weight values shall not go below zero or go above 3.

Fig. 2 Simplified STDP curve



For the initial weight values, you will use the weight map as shown above in Fig. 3. "<Weight map to 'Output neuron 1>" shows the initial weight values for the synapses between the 25 input neurons and "output neuron 1". Similarly, "<Weight map to 'Output neuron 2>" shows the initial weight values for the synapses between 25 input neurons and "output neuron 2". In Fig. 3, the values are represented in the binary format. (Ex. 11 = $11_{(2)} = 3_{(10)}$)

Fig. 3 Initial weight map

<Weight map to 'Output neuron 1'>

01	00	00	01	10
00	11	10	11	00
10	10	01	11	11
01	11	00	00	10
11	01	01	00	01

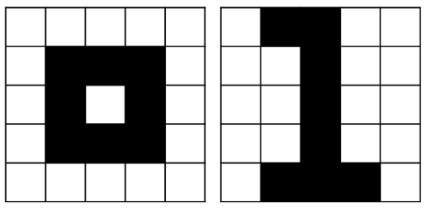
<Weight map to 'Output neuron 2'>

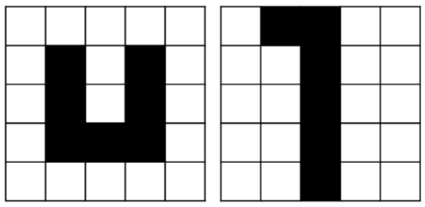
00	10	11	00	01
10	00	10	00	00
01	10	11	01	10
00	01	01	00	01
10	11	01	11	00

1.4. The datasets

In this assignment, you will only use a tiny training dataset which have two classes (digits '0' and '1') and one training example for each class (a total of 2 training examples), as shown in Fig. 4. Please use the training example for digit '0' first (the one on the top left), and then use the training example for digit '1' (the one on the top right). You should make sure that the weights are adapted with respect to the STDP curve in Fig. 2.

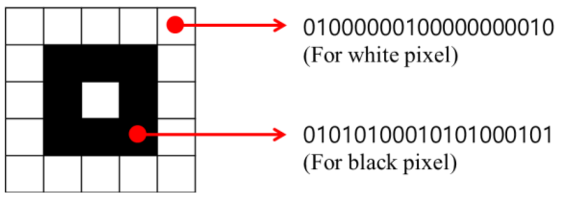
Fig. 4 Dataset


<Training Dataset>


<Testing Dataset>

After training (assume that the number of training iteration is 1), you will use the testing dataset which is also shown in Fig.4. There is one testing example for each class (a total of 2 testing examples). Make sure that the weights will be trained only during the training, and will not change during the testing.

Fig. 5 Firing sequence



In this assignment, you shall use the following coding scheme for the spike trains of the input neurons. As shown in Fig. 5, white pixels are represented as 01000000100000000010 and black pixels are represented as 01010100010101000101 (20-bits per input sequence, starting from the left). As mentioned above, since each neuron in the input layer represents a pixel. Therefore, there are twenty-five firing sequences going into the output layer neurons.

2. The Problem and Credit Breakdown

First, you shall train the network with the provided 'Training dataset'. After training the network using each example, provide a weight map with updated weights. In other words, you shall first train the network using the input firing sequences corresponding to the training input '0', and then report the updated weight map; You shall then repeat for the training input '1'. The reported weight maps shall be displayed in grayscale as shown in Fig. 6. Assume that the number of training iterations is one.

2.1. Results on training

For each training example, more specifically you shall report the followings:

1) The waveform $'V_i[t]'$ of the output neurons (output neuron 1, and 2)

2) The waveforms of the spike trains of the output neurons.

3) Numbers of firings of the output neurons.

4) Weight maps before and after training over each sample.

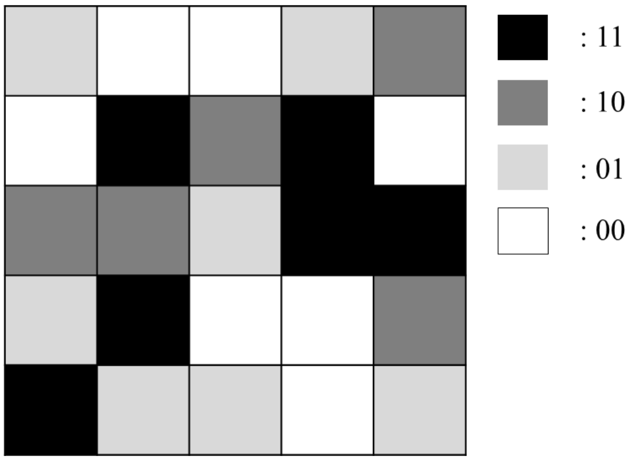
(Report the weight maps for both output neuron 1 and output neuron 2)

5) What does each weight map look like? Describe why it looks like what you may observe and compare the differences before and after the training.

Include the above results in your report.

Fig. 6

01	00	00	01	10
00	11	10	11	00
10	10	01	11	11
01	11	00	00	10
11	01	01	00	01



2.2. Results on testing

After training, test the network with the ‘testing dataset’ which is provided above. Use the trained weight values and see how the output neurons work.

For each testing example, **you shall include the following in the report**:

1) The waveform $'V_i[t]'$ of the output neurons (output neuron 1, and 2)

2) The waveforms of spike trains of the output neurons.

3) Numbers of firings of the output neurons

4) Check whether the network is well functioning and analyze the reasons. Which output neuron represents input digit ‘1’? Provide a detailed description of your design, particularly, describe how you implement STDP.

2.3. What to submit

A zipped file containing the report with the required results mentioned in Section 2.1 and 2.2. and your Verilog code and testbench.

Credit Breakdown

- Quality of the report & Verilog code, test bench: 20 points
- Correctness of each simulation:
  - Results on the training examples: 60 points
  - Results on the testing examples – 40 points

Submission Note

To complete this assignment, prepare three files: 1. report in PDF, 2. Verilog file, 3. Testbench file, and then zip them into a single file for uploading to Canvas. Please make sure that all the required results are included.