CAD - Final Project

MNIST Classification

Dr. Modarresi

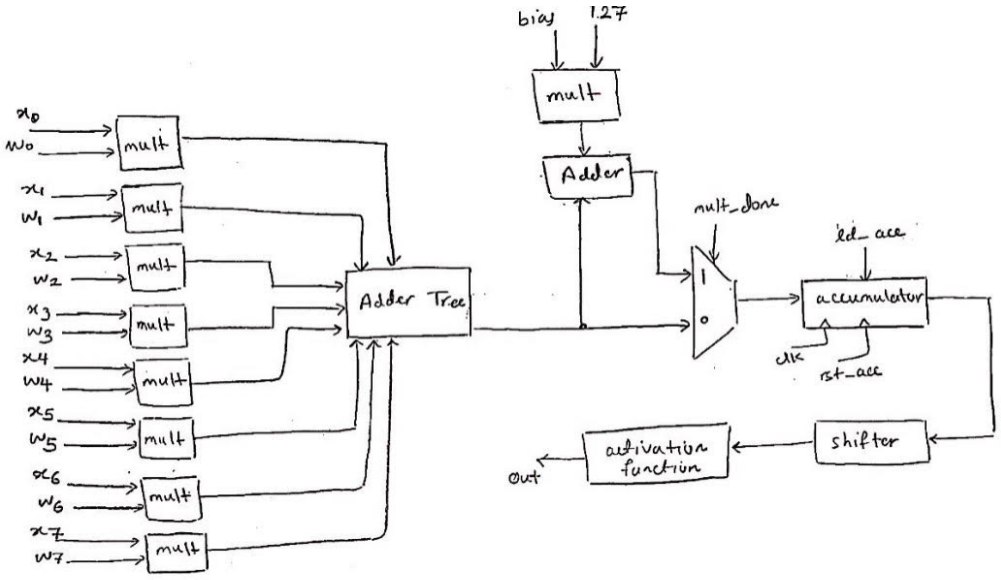
Fall 2021

Parnian Fazel 810198516

Paria Khoshtab 810198387

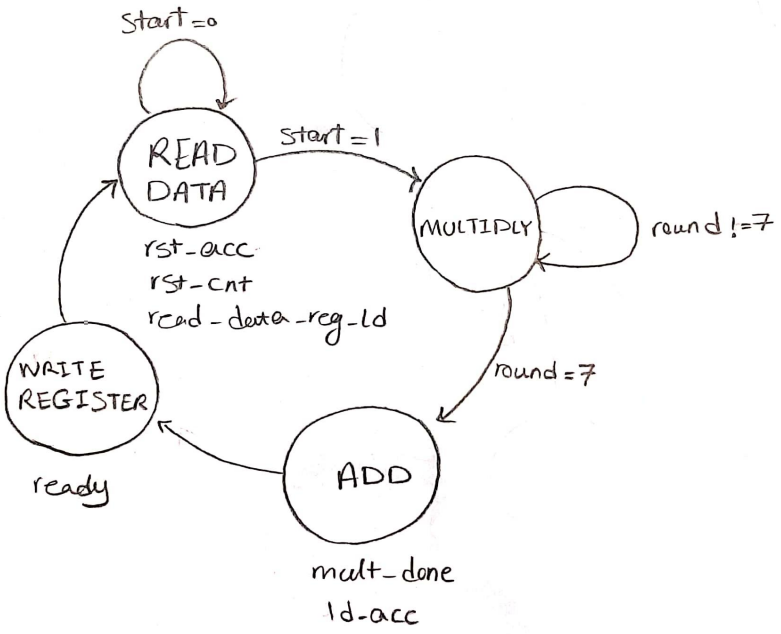
❖ PU

➢ Datapath



As can be seen above, each PUs is a multiply-and-accumulate (MAC) unit that has eight inputs and eight weights. It has eight multipliers, an adder tree, a multiplier for multiplying 8-bit bias value by +127, an adder for summing up partial sums and multiplied bias, an accumulator, a shifter for scaling the output layer inputs, and an activation function.

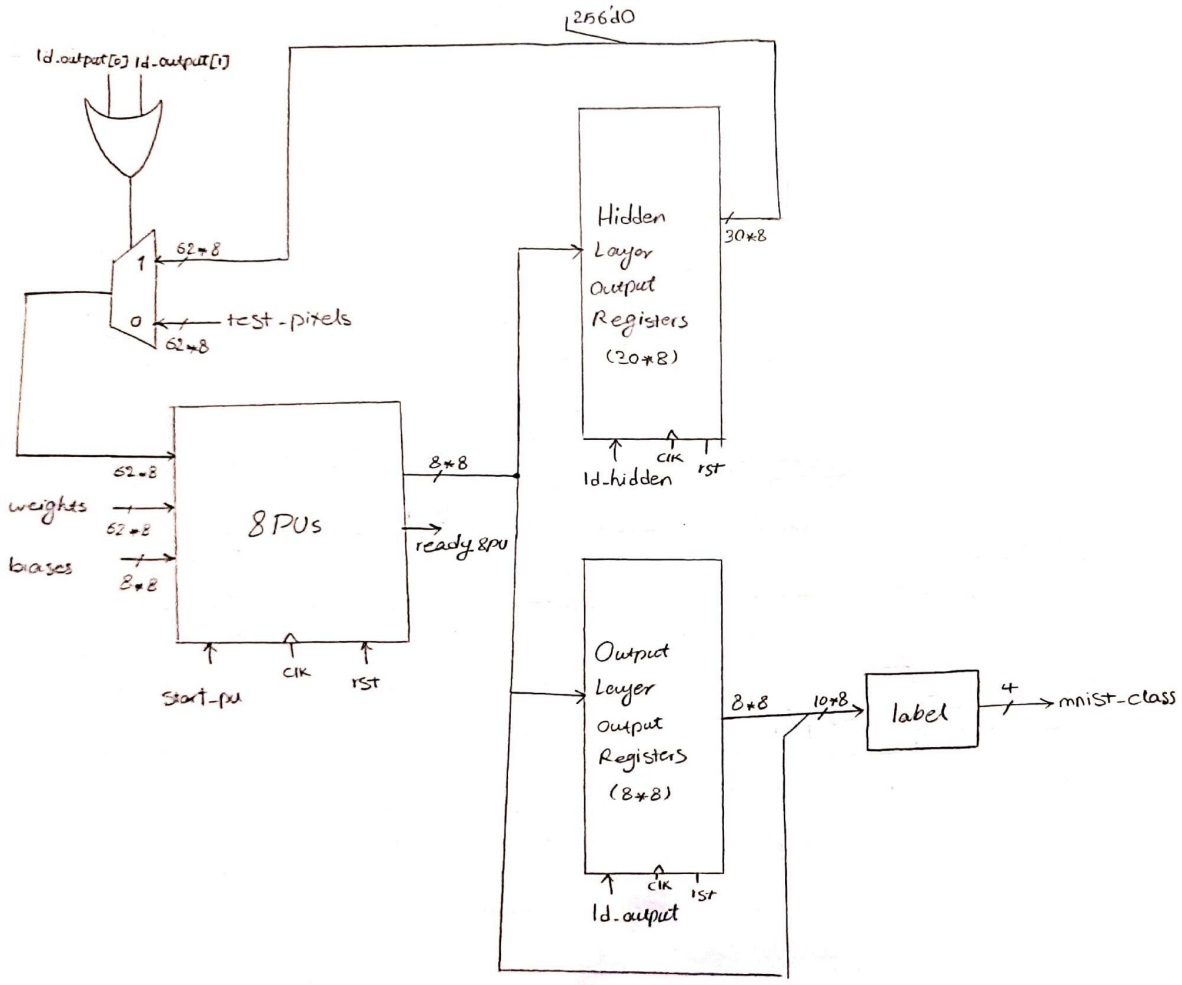
➢ Controller



As mentioned in the description, a PU requires four steps (reading data from memory, multiplier, the adder tree + activation, and writing the results to the registers. Each step takes one clock cycle) to calculate output of a neuron by summing 8 partial sums then adding with bias and finally applying activation function (here ReLU) to the output. Note that since there are 64 inputs, a PU requires at most 8 rounds to calculate the output result of a neuron.

❖ Neural Network

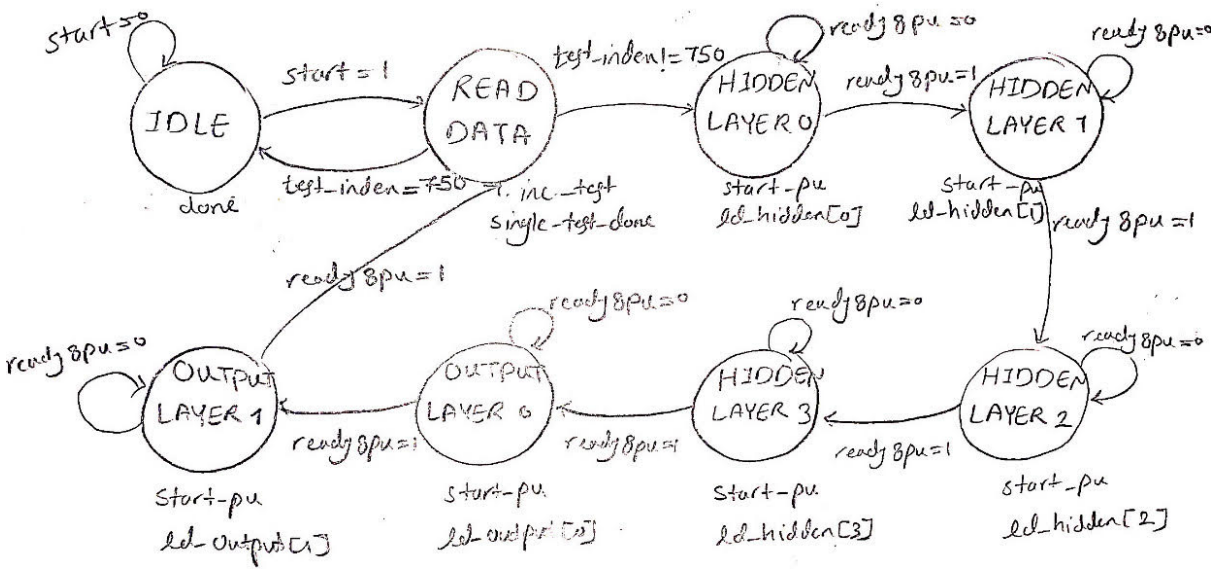
➢ Datapath



In this design the 8 PUs box calculates 8 neurons' outputs then the results are stored in a set of registers. First we calculate the hidden layer outputs by employing the 8 PU box 8 times with the inputs of the test inputs data and storing the results in the Hidden Layer Output Registers box. Then, by

using the outputs of hidden layer neurons (stored in the registers), we calculate the output layer outputs by employing the 8 PU box 2 times and store the results in the Output Layer Output Registers box. Finally, after having the results of all output neurons (10 neurons), we choose the maximum output in the Label box to find the predicted class.

➢ Controller



As can be seen above, our FSM consists of eight states:

- IDLE: All the inputs are labelled by feeding into the neural network (test_index = 750), so we issue the done signal and wait for another start pulse to start the flow again.

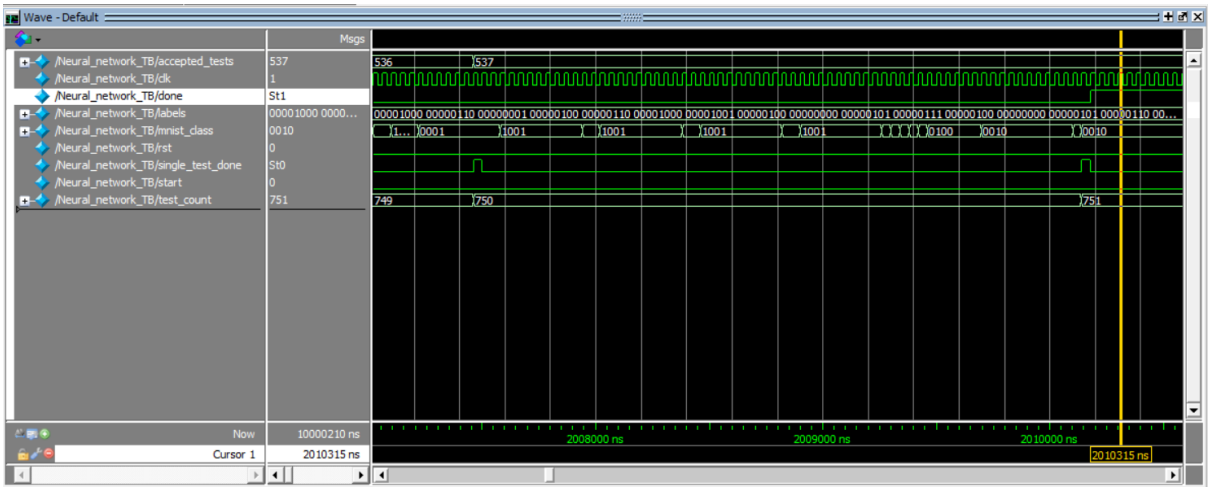
- READ_DATA: A single test is labelled by feeding into the neural network, so we issue the single_test_done signal and increment the number of labelled test inputs by setting inc_test signal to one.
- HIDDEN_LAYER_0: Inputs, weights, and biases related to neurons 0-7 of the hidden layer are selected to flow in the 8 PUs by setting start_pu to 1, and then the corresponding outputs are stored in a set of registers in the network datapath by setting ld_hidden[0] to 1. When all 8 PUs' outputs are ready(ready[i] = 1; for i = 0,..,7) the "ready8pu" signal is issued by the datapath and we go the next state.
- HIDDEN_LAYER_1: Inputs, weights, and biases related to neurons 8-15 of the hidden layer are selected to flow in the 8 PUs by setting start_pu to 1, and then the corresponding outputs are stored in a set of registers in the network datapath by setting ld_hidden[1] to 1. When all 8 PUs' outputs are ready(ready[i] = 1; for i = 0,..,7) the "ready8pu" signal is issued by the datapath and we go the next state.
- HIDDEN_LAYER_2: Inputs, weights, and biases related to neurons 16-23 of the hidden layer are selected to flow in the 8 PUs by setting start_pu to 1, and then the corresponding outputs are stored in a set of registers in the network datapath by setting ld_hidden[2] to 1. When

all 8 PUs' outputs are ready(ready[i] = 1; for i = 0,..,7) the "ready8pu" signal is issued by the datapath and we go the next state.

- HIDDEN_LAYER_3: Inputs, weights, and biases related to neurons 24-30 of the hidden layer are selected to flow in the 8 PUs by setting start_pu to 1, and then the corresponding outputs are stored in a set of registers in the network datapath by setting ld_hidden[3] to 1. When all 8 PUs' outputs are ready(ready[i] = 1; for i = 0,..,7) the "ready8pu" signal is issued by the datapath and we go the next state.
- OUTPUT_LAYER_0: In this state all outputs of 30 neurons in hidden layer are calculated. Hence, inputs, weights, and biases related to output neurons 0-7 are selected to flow in the 8 PUs by setting start_pu to 1, and then the corresponding outputs are stored in a set of registers in the network datapath by setting ld_output[0] to 1. When all 8 PUs' outputs are ready(ready[i] = 1; for i = 0,..,7) the "ready8pu" signal is issued by the datapath and we go the next state.
- OUTPUT_LAYER_1: In this state 2 neurons of output layer are still not defined so inputs, weights, and biases related to output neurons 0-7 are selected to flow in the 8 PUs by setting start_pu to 1, and then the corresponding outputs are stored in a set of registers in the

network datapath by setting ld_output[1] to 1. When all 8 PUs' outputs are ready(ready[i] = 1; for i = 0,..,7) the "ready8pu" signal is issued by the datapath and we go the next state. Now, all outputs of a test are ready and the prediction is done, so we go to the READ_DATA state to read the next test data inputs.

❖ Simulation Result:




```
VSIM 35> run -all
# --> Accuracy:            71%
```

$$Accuracy = \frac{\#correct\ predicted}{\#test} = \frac{537}{750} = 0.71$$

* Note that we have used given weights and inputs, related to the trained MNIST perceptron neural network.