**ECE 298A Deliverable**

**Vivi Huang, Evan Zhang**

**Design Specifications**

Our design specifications are shown below.

Design Objective:

Implement a 4-neuron Binary Neural Network (BNN), on a Tiny TapeOut ASIC to classify a 6-bit binary input (line inputs) to perform classification into 4 different classes: turn left, turn right, stop, and U-turn.

Input/Output:

Use 6-bit parallel input and 4 binary outputs + 2 debug outputs, out of the 8 total I/Os. 6-bit parallel inputs are assumed to be “sensor-gathered data”. 4-bit output provides 4 different classifications for the input. 2-bit debug bits can be used to observe intermediate arithmetic processes for neuron 0, both for our building purposes as well as learning purposes.

4 of the 8 bidirectional I/O pins will be used to implement dynamic weight loading. This is loaded during runtime, and weights are shifted serially based on the cycle number. In two clock cycles, the 4 bidirectional I/O pins will finish writing into one neuron, and the complete process can be completed within 8 cycles. We use an additional I/O pin to use as the *load\_enable* signal to enable loading. An I/O table is shown below:

| Description | Pinout | Description |
| --- | --- | --- |
| Input | ui[7:2] | 6-bit parallel line data input |
| ui[1:0] | These pins are unused |
| Output | uo[7:4] | 4-bit classification output, uo[7] identifies left turn, 6 identifies right turn, 5 identifies turn around, 4 identifies stop. |
| uo[3:2] | 2-bit debug pins, uo[3] for neuron 0 after popcount, uo[2] for after threshold op. |
| uo[1:0] | These pins are unused |
| Bi-directional I/Os | uio[7:4] | 4-bit weight data, loaded serially |
| uio[3] | *Load\_enable* signal to enable loading |
| uio[2:0] | These bidirectional pins are unused; could be assigned to debugging purposes. |

Table 1. I/O table diagram

A block diagram is shown below:

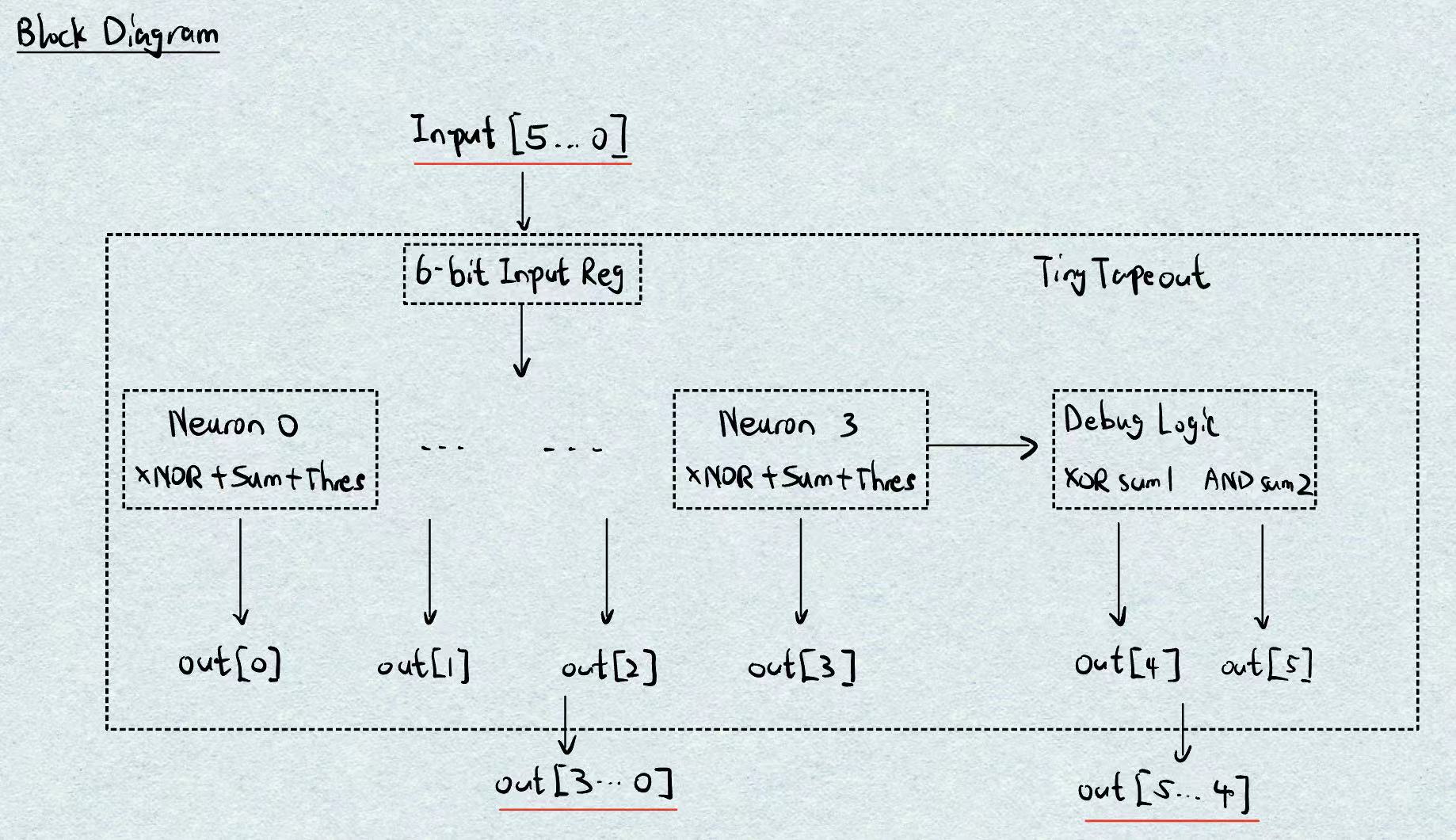
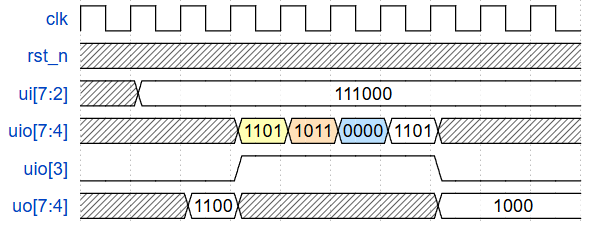


Figure 1. Block Diagram

A timing diagram is shown below:

Figure 2. Timing Diagram

Hardware Specs:

Each neuron will contain 6 weights and registers accordingly to perform **XOR + popcount + threshold** operations to produce a single output bit. By estimation and per neuron, we will need 11 flip-flops per neuron as well as 10 wires per neuron. FFs are used for weight storage, threshold register plus sum register. Wires are used in these 3 processes as well. In the total design, this accumulates to **53 FFs and 46 wires**. This means approximately 800 gates total, which is well within the limit of Tiny Tapeout Specs.

Testing:

Testing will be done just using CocoTB. We’ll be able to use the 8-bit I/O, clock, reset, power and ground like written above. The weight-loading can also be tested using 8 bi-directional I/Os.

Note that weight-training will be done outside of the Tiny-Tapeout chip, and will be performed separately using the Larq framework in Python to train the weights for BNN. A library of training data will be given to the BNN; this is what underlies the fundamental weights for our neural network.

Taking that into account, we will have 75 samples of training data sets and 15 sets of testing data in practice. These numbers are chosen for the following reasons. 1, BNN overfits the data quickly and a smaller dataset is beneficial to prevent overfitting. 2. 15 individual tests suffice to include all 4 feature detections, including some variations, to see if the classification process is satisfactory.