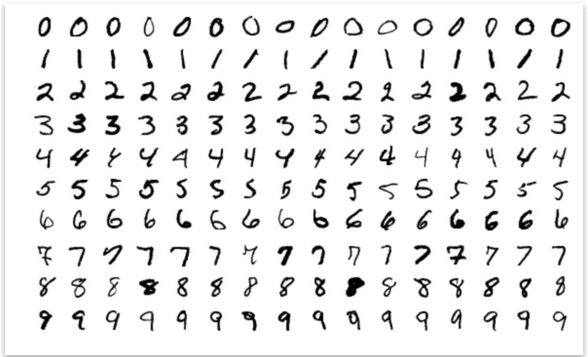
Hardware Approach of Forward Propagation in Neural Network on FPGA

Mid-Project Progress

Team 9
Randy, Wales, Yu & Zhuojun

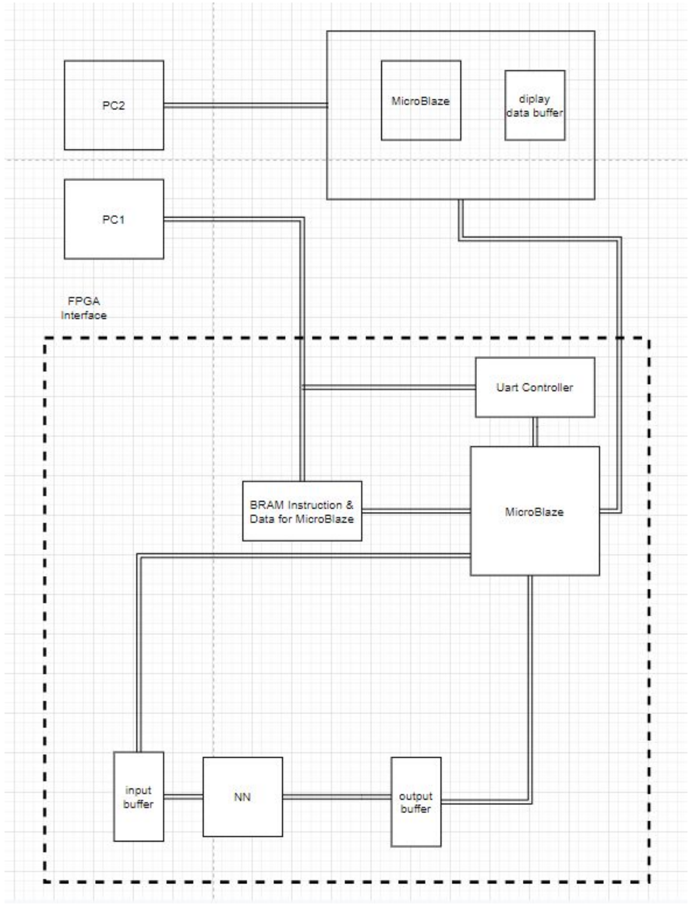
Project Overview

- Neural Network with forward propagation implemented on hardware and backward propagation on MicroBlaze
- This Neural Network will be able to recognize handwritten digits using the MNIST dataset to train it
- The Neural Network will be conducting forward propagation throughout 2 layers



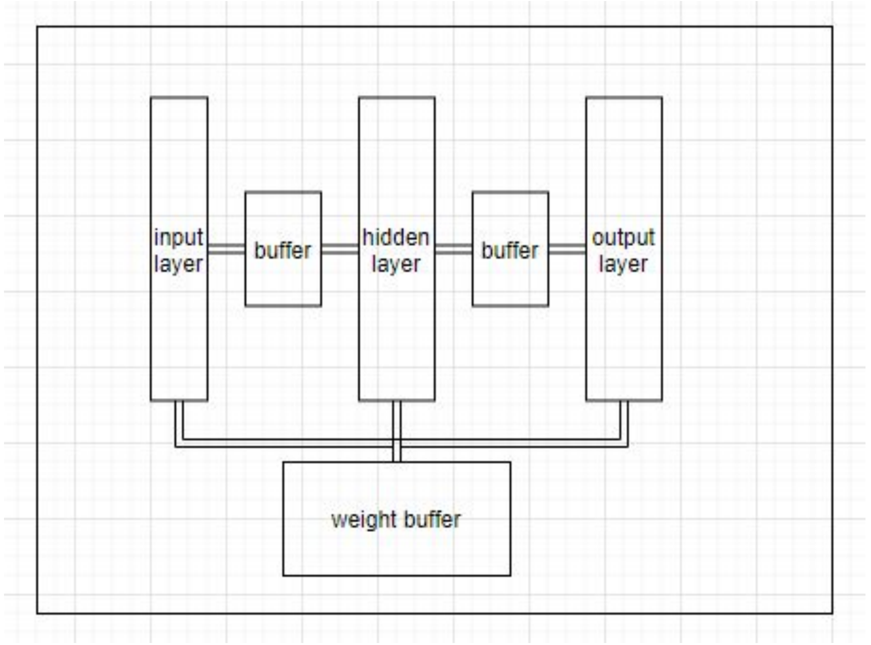
Proposed System Diagram (Hardware)

- The data will be stored in a BRAM buffer
- Neural Network block will retrieve the input data from the buffer
- Output data, as well as the target output (expected output), will be stored in a BRAM buffer
- The MicroBlaze will retrieve the data from the BRAM buffer and perform backpropagation, updated weights are stored in another BRAM buffer



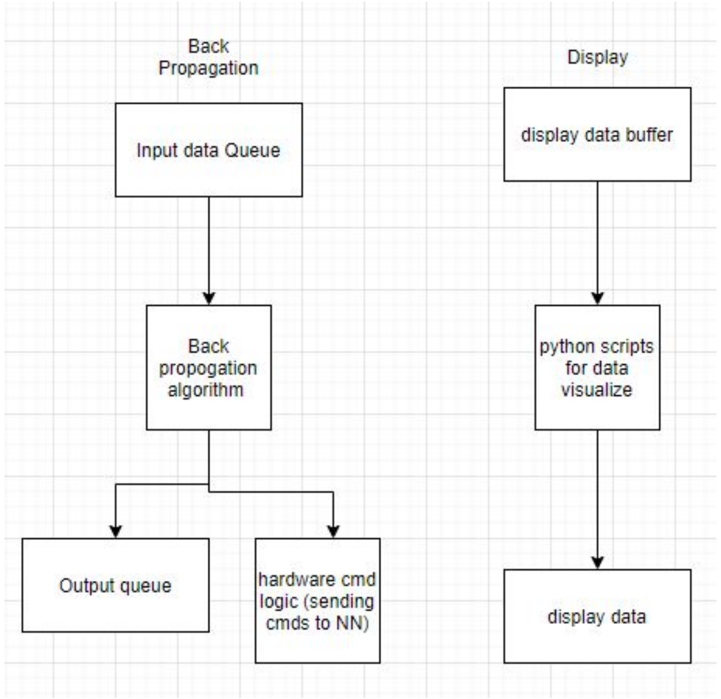
Proposed System Diagram (Neural Network)

- Multi-Stage 2-Layer Pipelined Architecture
- Forward Propagation will be performed from the input layer of the Neural Network to the output layer, through fully connected layers
- Each layer output to a temporary buffer
- Each layer connected through AXI4-stream
- Weights stored as a matrix



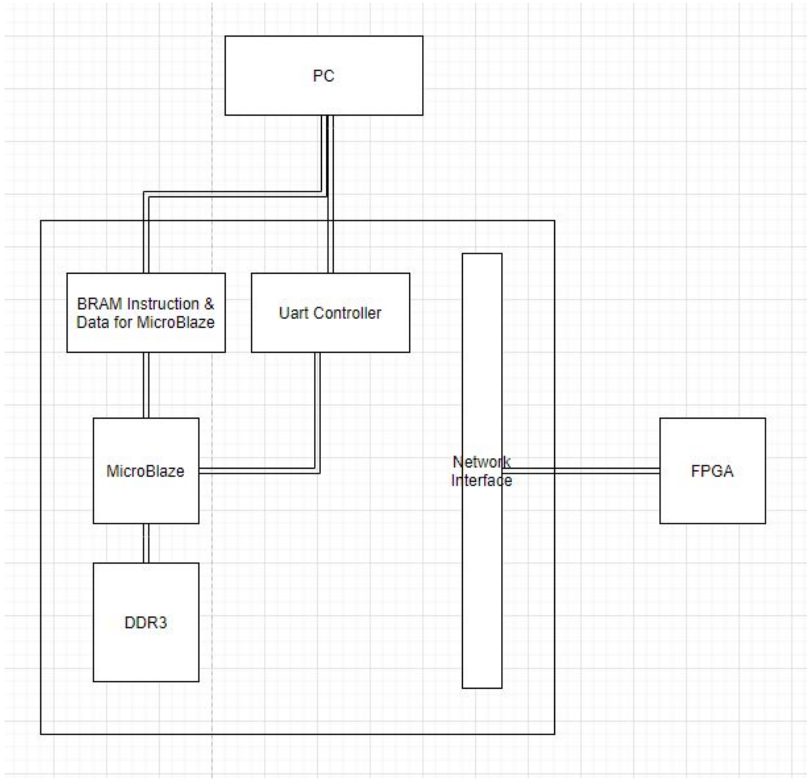
Proposed System Diagram (Software)

- Back propagation is conducted using MicroBlaze through software
- Neural Network block receives a signal indicating weights are to be updated, retrieve the weights from the buffer
- Cycle through the previous process, then after certain numbers of runs, send the data to a BRAM buffer for monitoring display
- The results will be collected by the monitoring unit (in our case, it will be on another FPGA board)


Back Propagation
Input data Queue
Back propogation algorithm
Output queue
hardware cmd logic (sending cmds to NN)
Display
display data buffer
python scripts for data visualize
display data

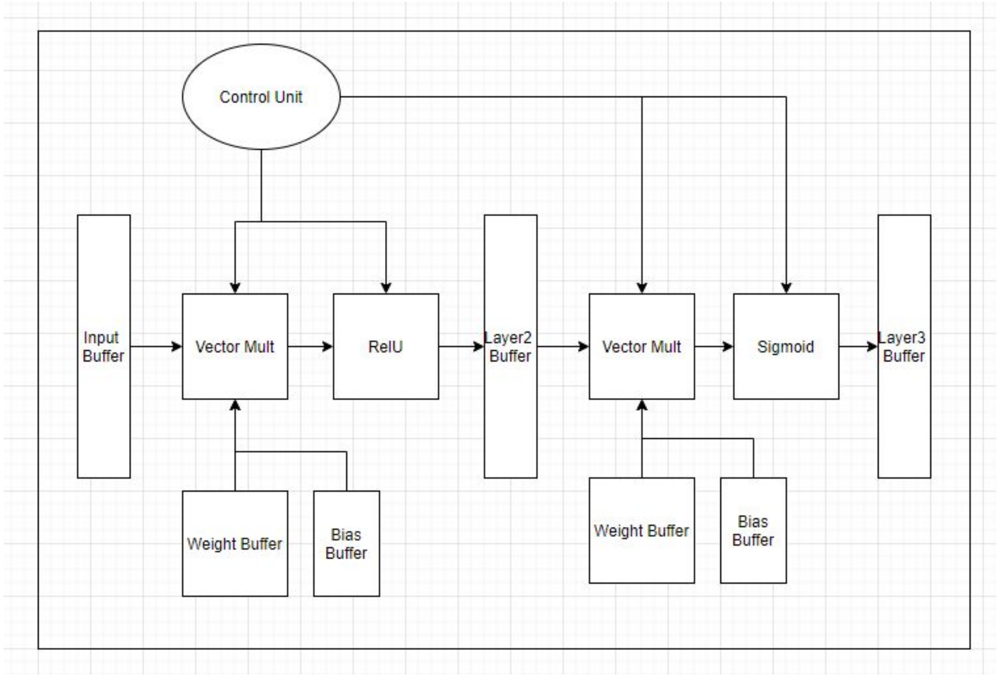
Current System Diagram (Hardware)

- Testing data and pre-trained weights and biases (with Python Keras) are sent through UART as Byte and floating-point values with a baud rate of 230400
- The testing data will be stored in a DDR3 memory alongside pre-trained Neural Network weights and biases
- MicroBlaze uses software implementation of 2-layer Neural Network (784-800-10) to conduct classification on training data



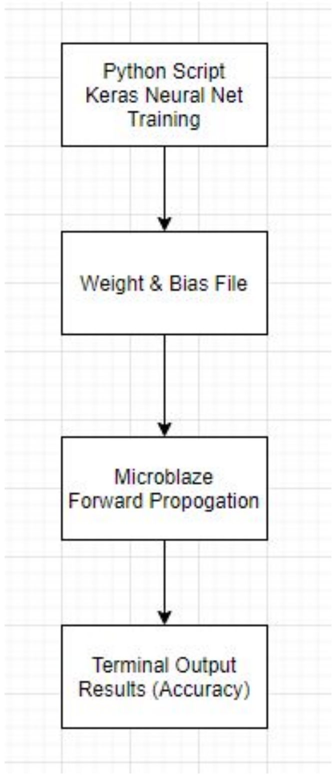
Current System Diagram (Neural Network)

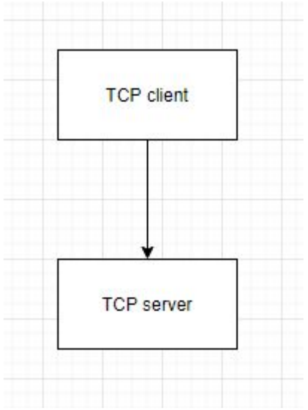
- Multi-Stage 2-Layer Sequential Architecture
- Forward Propagation will be performed from the input layer of the Neural Network to the output layer
- Weights and Biases stored in a buffer
- Activations and weights are multiplied in Vector Mult Unit (saves more space compared to Matrix Mult)
- Uses 16-bit fixed-point precision (Q4.12)
- ReLU and Sigmoid approximation used as activation functions for layers



Current System Diagram (Software)

- Training and back propagation is conducted on PC using Python (pre-trained weights and biases)
- Data is sent using the TeraTerm "Send File" function to send testing images and Neural Network weights and biases
- On MicroBlaze, the forward propagation is performed with pre-trained weights & bias
- Implementation of TCP network communication between two Nexys Video Board FPGAs



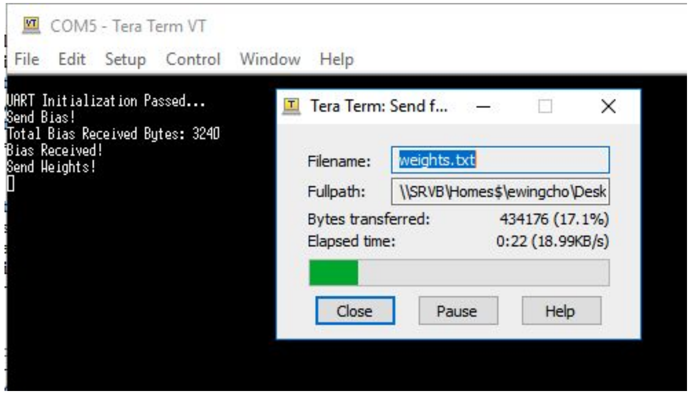


Challenges

- Maintaining Data Precision in Hardware (Floating-Point to Fixed-Point Conversion)
  - Loss in precision with quantized bits and potential data overflow for MAC (Multiply-Accumulate) computation
- FPGA Resource Utilization
  - Still unclear on designing 2-layer (784-800-10) Neural Network will utilize more resources than we have
- Timing constraints with data communication and MicroBlaze Performance
  - Using UART connection at the highest BAUD rate still requires ~5 minutes to send all 10,000 MNIST testing images
    1. Longer if we are training with 60,000 MNIST training images
  - MicroBlaze will take ~8 hours to pre-process all 10,000 testing images; had to reduce down to 100
  - MicroBlaze will take days to train our Neural Network if we integrated backpropagation

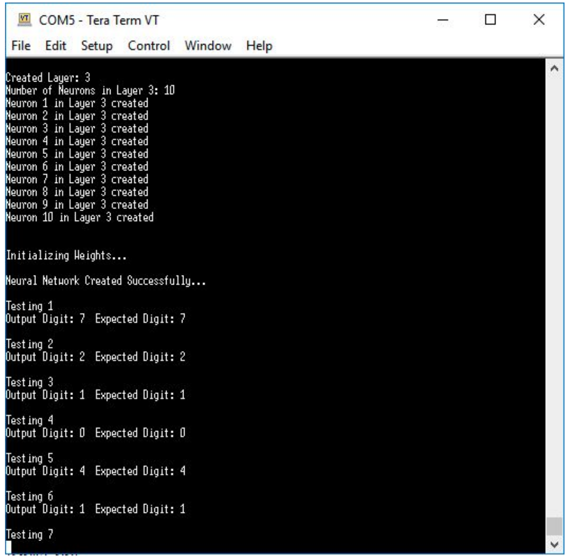
Mid-Progress Demo

- Forward Propagation of MNIST dataset on MicroBlaze with pre-trained weights and biases for a 2-layer (784-800-10) Neural Network
- Neural Network for logic implementation on MicroBlaze with real-time creation, training and testing
- Nexys Video board TCP/IP communication between two FPGA boards
- Sigmoid approximation hardware block for the activation function



```
UART Initialization Passed...
Send Bias!
Total Bias Received Bytes: 3240
Bias Received!
Send Weights!
```

Filename: weights.txt
Fullpath: \\SRVB\Homes$\ewingcho\Desk
Bytes transferred: 434176 (17.1%)
Elapsed time: 0:22 (18.99KB/s)



```
Created Layer: 3
Number of Neurons in Layer 3: 10
Neuron 1 in Layer 3 created
Neuron 2 in Layer 3 created
Neuron 3 in Layer 3 created
Neuron 4 in Layer 3 created
Neuron 5 in Layer 3 created
Neuron 6 in Layer 3 created
Neuron 7 in Layer 3 created
Neuron 8 in Layer 3 created
Neuron 9 in Layer 3 created
Neuron 10 in Layer 3 created


Initializing Weights...

Neural Network Created Successfully...

Testing 1
Output Digit: 7  Expected Digit: 7

Testing 2
Output Digit: 2  Expected Digit: 2

Testing 3
Output Digit: 1  Expected Digit: 1

Testing 4
Output Digit: 0  Expected Digit: 0

Testing 5
Output Digit: 4  Expected Digit: 4

Testing 6
Output Digit: 1  Expected Digit: 1

Testing 7
```

Remaining Work

- Neural Network Hardware implementation integrated into the current software implementation on the MicroBlaze
- Integrate the TCP/IP FPGA communication into our current MicroBlaze MNIST implementation
  - Neural Network performance is sent to the second FPGA and stored on the second PC
  - Interpret data in CSV or Python Application
- Possibly, integrate backpropagation on hardware to speed up training
  - If time and resources are available

Final Demo Plan

- Show full-system of forward propagation with testing images
- Neural Network classifies within reasonable accuracy >85%
- The second PC receives live performance data from the first FPGA through ethernet and second FPGA and interprets it in a unique and well-organized way
- Backpropagation and training of Neural Network through hardware or software?