Mini Project II

1 Introduction

Multi-Layer Perceptron (MLP) is a category of feed-forward artificial neural networks. The biological neuron can be modelled mathematically as a perceptron. As its name suggests, an MLP is composed of perceptrons arranged in layers; usually an input layer which serves as an interface between the MLP and the input data (or signal), an output layer which is responsible for making the final prediction (or classification) and a number of layers between the output and the input layers called the hidden layer. The number of hidden layer is arbitrary and it denotes how "deep" the neural network is.

Parameters such as the number of neurons (perceptron) in a layer, number of layers in an MLP, choice of activation function and bit-width of the data within the network are some of the important parameters that affects the entire network implementation in hardware as the power, frequency and area of the final hardware design is influenced by these parameters. Additionally, the organization of neurons and layers within the design, serialization or parallelization of the network greatly affects the mentioned metrics.

The goal of this project is to explore the design space for a hardware implementation of an MLP, investigating the effect of the mentioned parameters in the overall area, timing and power of the network's synthesized design. This report contains the design, simulation and synthesis process of an MLP neural network using hardware description languages (VHDL and Verilog). The goal is to implement an MLP with:

- a one-neuron design for $M * N$ MLP (N layers with M neurons each) under 32-bit, 16-bit and 8-bit data precision.
- an N-neuron design for $M * N$ MLP under 32-bit, 16-bit and 8-bit data precision.
- a one-neuron design for $M * N$ MLP with 3 different activation functions (Sigmoid, Step and ReLU)
- an N-neuron design for $M * N$ MLP with 3 different activation functions (Sigmoid, Step and ReLU)

2 Method

This section explains the design and implementation process of the project. A mixed VHDL and Verilog design was implemented, simulated and synthesized using Vivado and Synopsys Design Compiler. The design of the neurons itself is explained, as well as the design of the MLP using one and N neurons. Then other implementation details such as the activation functions and data precision are explained.

2.1 The Neuron Design

The building block of an MLP are neurons. In this project, the neurons have been designed in a parallel scheme. The top module IP block of the parallel mac neuron is shown in figure 2.1.

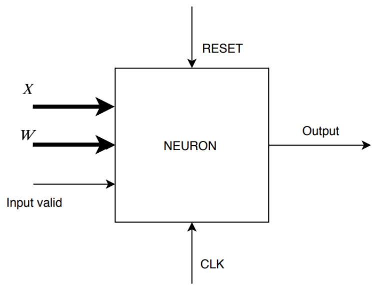


Figure 2.1: IP block for an M-input neuron with parallel MAC

The architecture has three main parts. The parallel multipliers, a multi-operand adder and an activation function. These can be seen in figure 2.4.

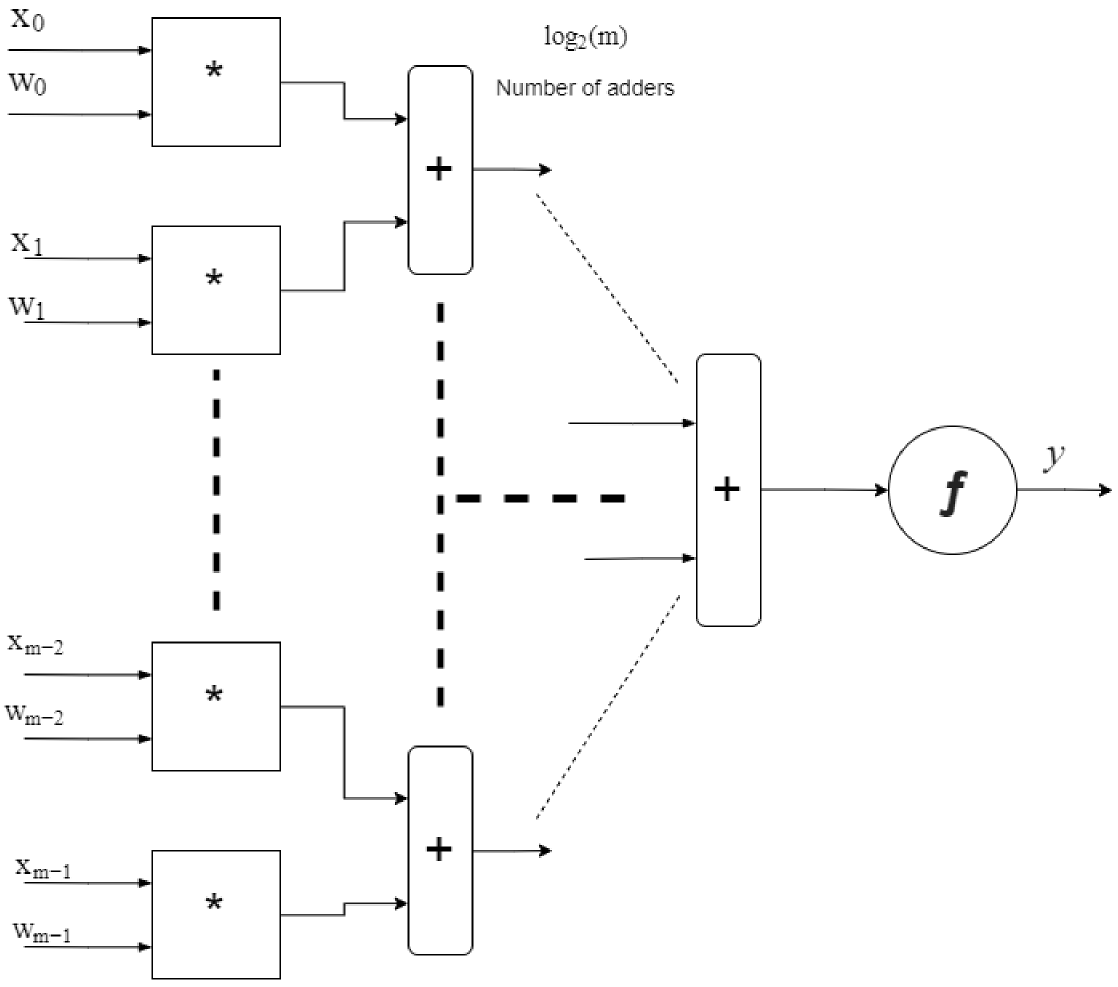


Figure 2.2: Block diagram for an M-input neuron with parallel MAC

When there are valid inputs at the multiplier, the signal referred to as *input_valid* in the block diagram goes high. The output of the multipliers are then feed into a combinatorial adder tree where each adder has a two inputs. The depth of the adder tree will scale with the amount of inputs to the neuron as $log_2(\#inputs)$. The input to the first layer of the adder tree will have 2 times as many bits as the input to the neuron. Every layer in the adder tree will add one bit to the width of the output. The last layer will therefor have a output width of

$$2 * neuron\_input\_width + log_2(\#neuron\_inputs)$$

. This output is truncated before it's passed to the activation function. The truncation is explained in section 2.5.

2.2 The One-Neuron MLP

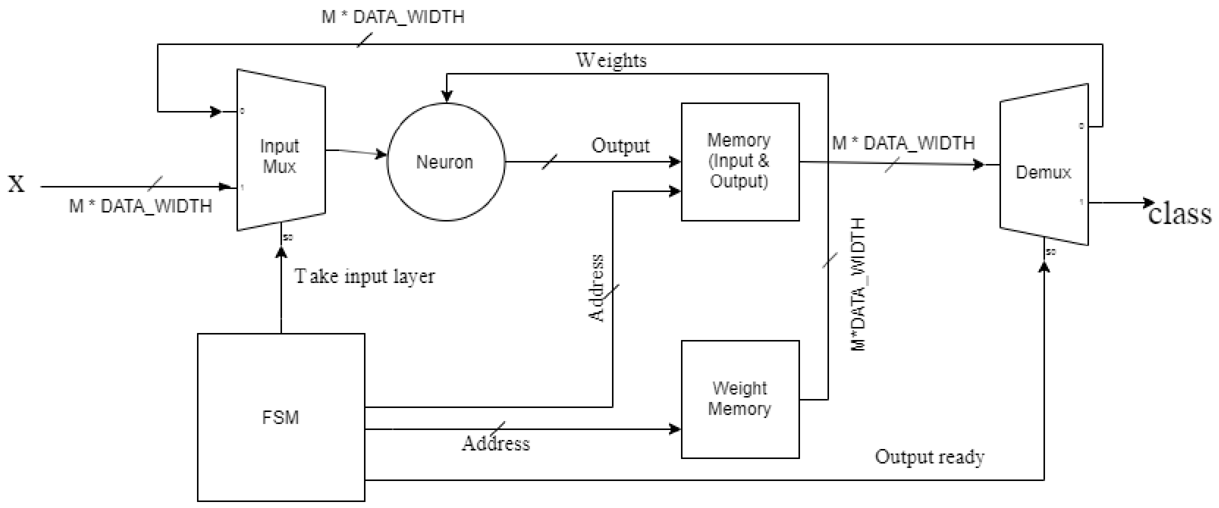


Figure 2.3: Block diagram for a One-Neuron MLP

The one-neuron architecture is shown in Figure 2.3. The model works by alternating between two sets of memory located within Memory (Input & Output). These 2 sets of memory will be called as the memory banks Set A and Set B. The Input and Output from the Neuron are controlled by the FSM. For one iteration Set A acts as the Input and provides the input to the neuron unit through Input Mux which in-turn writes to the Set B, and in the next iteration Set B serves as the input and neuron unit writes to Set A. This alternates as for the Number of Layers. This optimization was taken to reduce the memory usage of the accelerator thereby improving the characteristics of the design.

2.3 The N-Neuron MLP

The N-neuron architecture is depicted in Figure 2.4. The design contains N layers with M neurons in each layer. In contrast to the one-neuron design where the architecture is realized by using only one neuron, this design has $M * N$ neurons instantiated in it, all working in a parallel scheme. The data path of the network is fully combinations and the full computation is done in one clock cycle. I.e the throughput is always 1. The weights are stored in a memory and are loaded into the corresponding neuron when input is ready at the first layer.

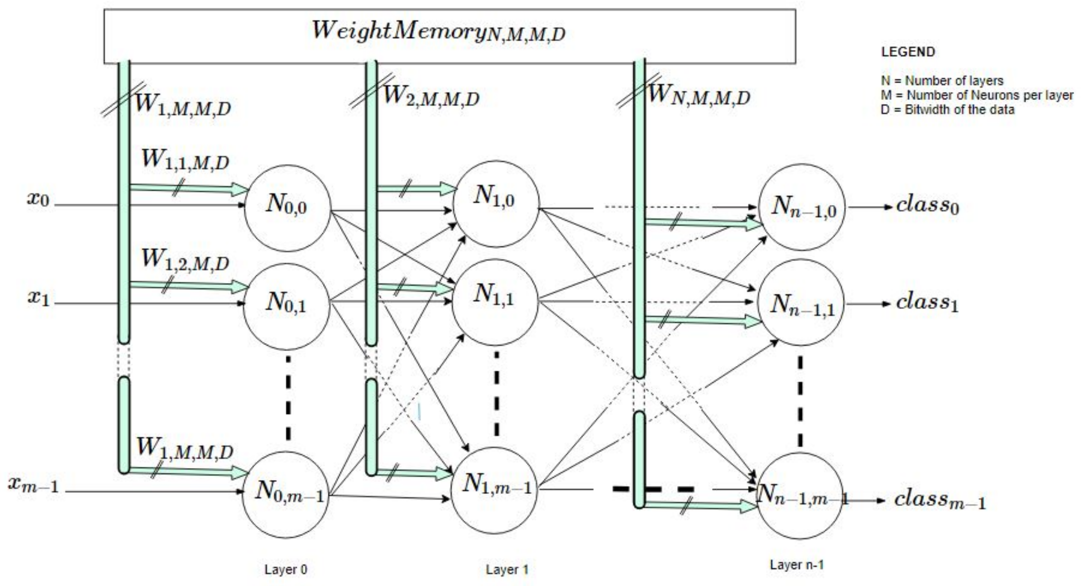


Figure 2.4: Network diagram for an N-Neuron MLP

2.3.1 Fixed Point Numbering System Design

The fixed-point formats of $Q_{4.4}$, $Q_{8.8}$ and $Q_{16.16}$ for data widths of 8, 16 and 32 bits respectively. The resolutions were 2^{-4}, 2^{-8} and 2^{-16} for the corresponding data widths.

2.4 Activation Functions

Three different activation functions were implemented in Verilog for this project: The Sigmoid function, the ReLU function and the Step function. Since an unsigned numbering system was used throughout the project, a shifted variant of the mentioned functions were implemented, to be able to see the whole output range of the functions in the simulation.

The step and the ReLU functions were implemented in Verilog using simple if and else statements, taking the bit width of their input and output as a parameter. Regarding the Sigmoid function, an approximation was implemented by taking 7 samples from the function for inputs in the range of [-6,+6].

2.5 Truncation

The bit growth while multiplication and addition is taken care of within the boundary of the MAC units and the activation function. The bit growth caused by the MAC unit at the worst case will cause its output to require $2 * input\ bit\ width + log_2(number\ of\ inputs)$ bits. Since this can grow rapidly using higher bit widths and larger input numbers, the bit growth cannot be sustained. Therefore, it was decided to truncate the outputs of both the serial and parallel MAC units before passing it to the activation function.

The activation functions were designed with the same bit width as the neuron inputs, so the output from the MAC was truncated down to either 8, 16 or 32 number of bits depending on the data width used. The same fixed-point Q format was sustained as well.

3 Results

The results shown in the tables in the following section were produced with Synopsys design compiler. The designs where compiled with the following design parameters

- Wire load mode : TOP
- Wire load model : WireAreaLowkCon
- Cell library : tcbn90gtc NCCOM

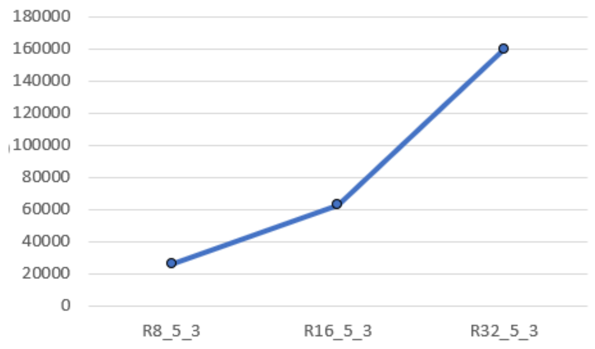
Following results are derived from the cells, area, power, and timing reports. The legend for identifying models in the figures are as follows:

Each model is denoted with a symbol Ax_y_z. A is a letter that indicates the activation function:
R → ReLU
S → Step
G → Sigmoid
The first number (x) indicates the bit-width, the second number (y) indicates M, or number of inputs or the number of neurons in each layer. The last number (z) indicates the N, or the number of layers. Example : R8_5_3 means the data was obtained from a model that used ReLU function, with the bit-width of 8, had 5 inputs and 3 layers.

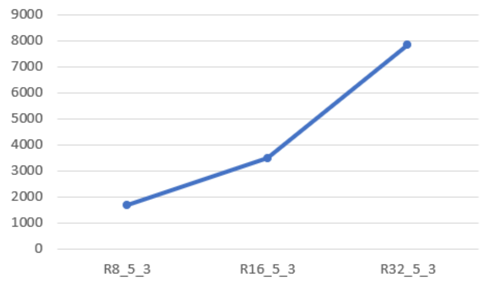
3.1 One-Neuron MLP

Number of bits	Power	Number of Cells	Total cell area:	Max clock frequency
8	2.3541 mW	1680	25951.56 nm	342.46 MHz
16	4.8704 mW	3486	62711.30 nm	207.90 MHz
32	10.2376 mW	7832	159682.52 nm	203.25 MHz

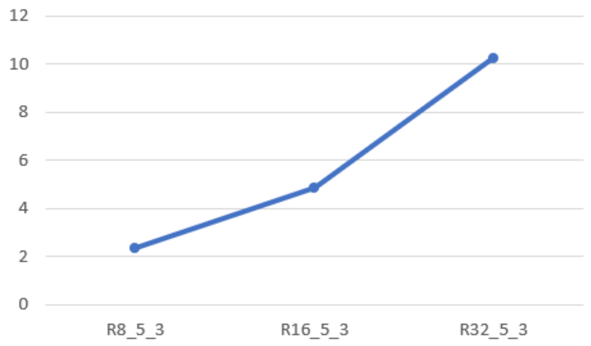
Table 3.1: The reported values for a 5-input one-neuron MLP with 3 layers using ReLu as activation function while varying the bit-width



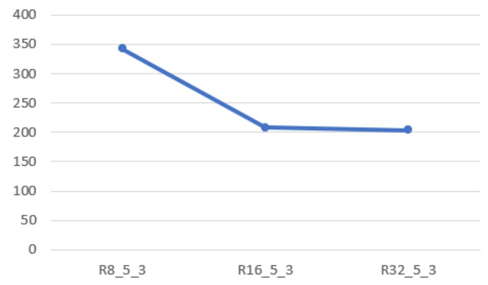
(a) Area (nm) vs. Bit Width.



(b) Cells vs. Bit Width.



(c) Power (mW) vs. Bit Width.

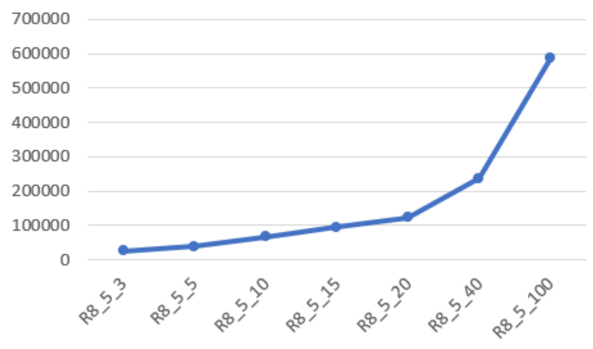


(d) Max Frequency (MHz) vs. Bit Width.

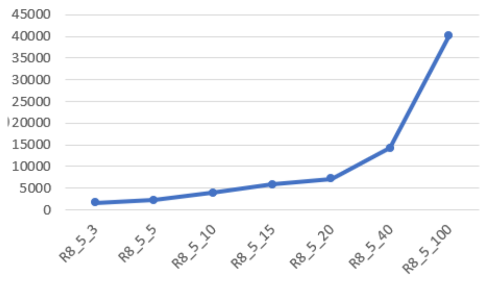
Figure 3.1: Effect of changing the data precision (8, 16 and 32bits) for a MLP with one-neuron model with constant number of inputs, constant number of layers and ReLU as the activation function.

Number of Layers	Power	Number of Cells	Total cell area:	Max clock frequency
3	2.3541 mW	1680	25951.56 nm	342.46 MHz
5	3.5749 mW	2237	38555.59 nm	337.83 MHz
10	6.5563 mW	3918	66771.53 nm	354.60 MHz
15	9.5368 mW	5781	94760.06 nm	354.60 MHz
20	12.4921 mW	7139	121951.87 nm	353.35 MHz
40	24.4239 mW	14278	237234.82 nm	343.64 MHz
100	60.9126 mW	40166	588247.84 nm	298.50 MHz

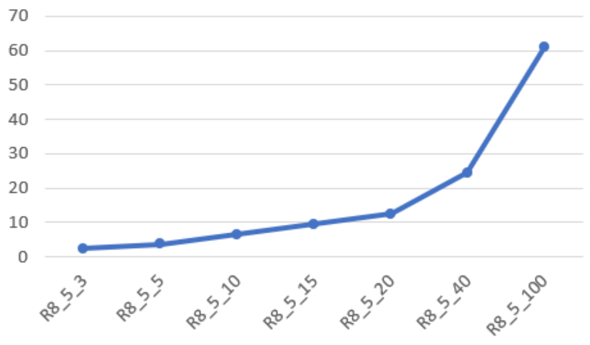
Table 3.2: The reported values for an 8-bit one-neuron MLP with 5 inputs using ReLu as activation function while varying the number of layers



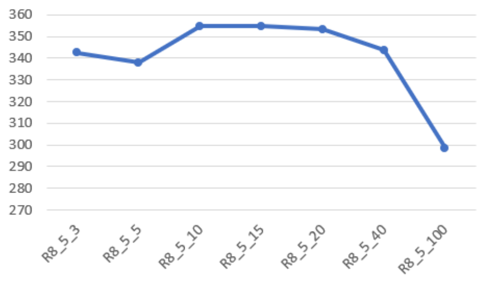
(a) Area (nm) vs. Layer.



(b) Cells vs. Layer.



(c) Power (mW) vs. Layer.

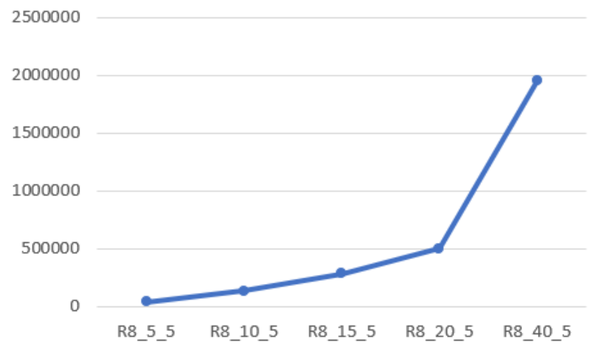


(d) Max Frequency (MHz) vs. Layer.

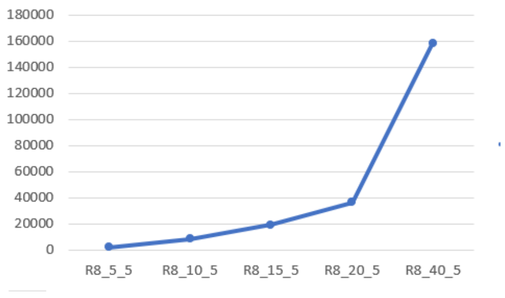
Figure 3.2: Effect of changing the number of layers for a one-neuron MLP with constant number of input, constant bit-width and ReLU as the activation function.

Number of Inputs	Power	Number of Cells	Total cell area:	Max clock frequency
5	3.5749 mW	2237	38555.59 nm	337.83 MHz
10	12.9891 mW	8648	134322.85 nm	303.95 MHz
15	28.5342 mW	19392	287137.27 nm	295.85 MHz
20	51.7849 mW	36665	502067.06 nm	272.47 MHz
40	198.7172 mW	158592	1953175.02 nm	204.91 MHz

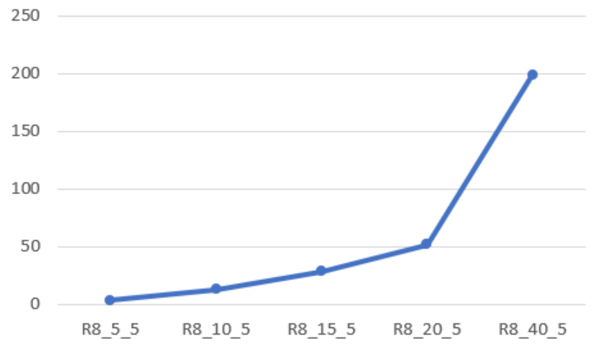
Table 3.3: The reported values for an 8-bit one-neuron MLP with 5 Layers using ReLu as activation function while varying the number of inputs



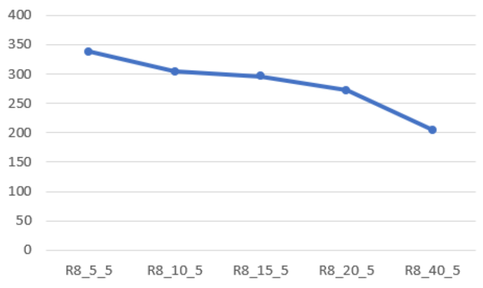
(a) Area (nm) vs. Input.



(b) Cells vs. Input.



(c) Power (mW) vs. Input.

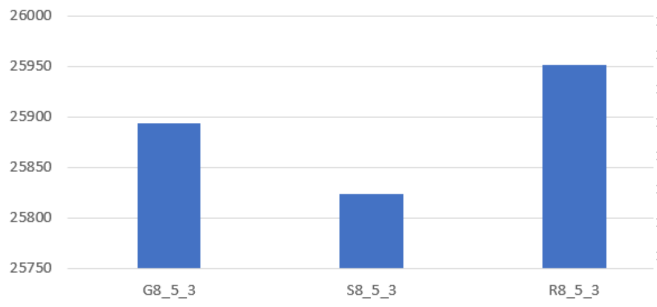


(d) Max Frequency (MHz) vs. Input.

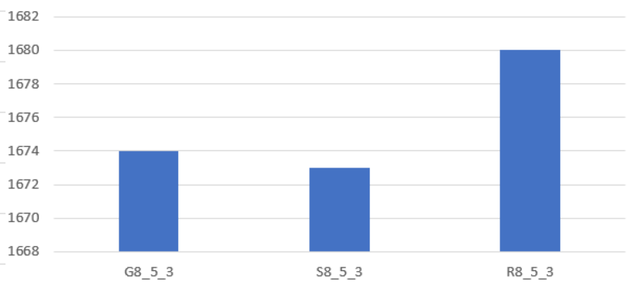
Figure 3.3: Effect of changing the number of inputs for a one-neuron MLP with constant number of layers, constant bit-width and ReLU as the activation function.

Table 3.4: The reported values for a 3 input serial 8-bit neuron while varying the activation function

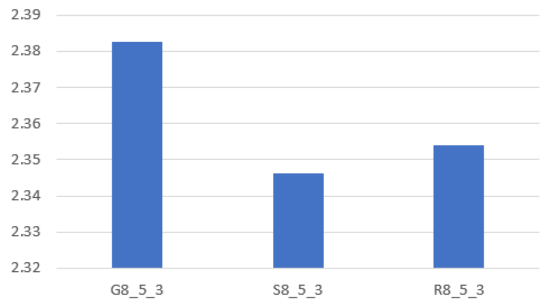
Activation Function	Power	Number of Cells	Total cell area:	Max clock frequency frequency
Step	2.3462 mW	1673	25824.15 nm	370.37 MHz
ReLu	2.3541 mW	1680	25951.56 nm	342.46 MHz
Sigmoid	2.3825 mW	1674	25893.5 nm	359.71 MHz



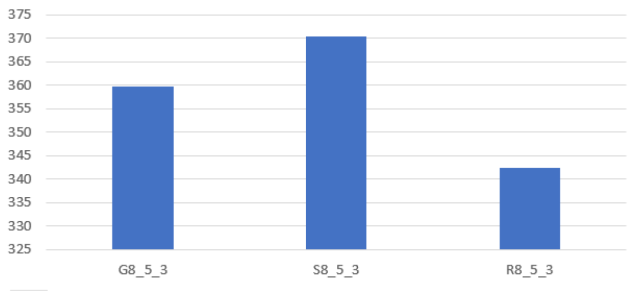
(a) Area (nm) vs. Activation Functions.



(b) Cells vs. Activation Functions.



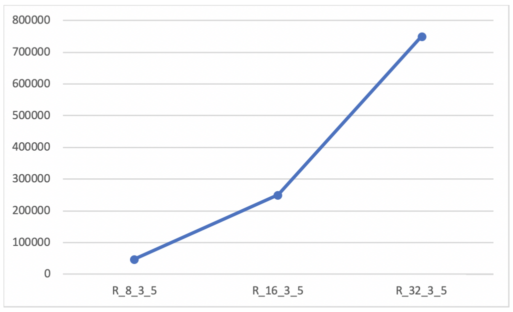
(c) Power (mW) vs. Activation Functions.



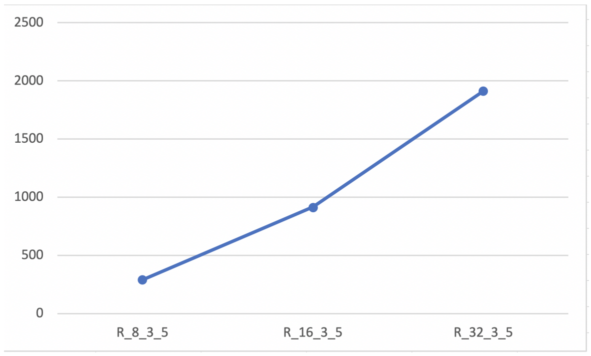
(d) Max Frequency (MHz) vs. Activation Functions.

Figure 3.4: Effect of changing the Activation Functions for a MLP with single Neuron Model with constant number of Layers, Constant Bit Width and constant number of Inputs.

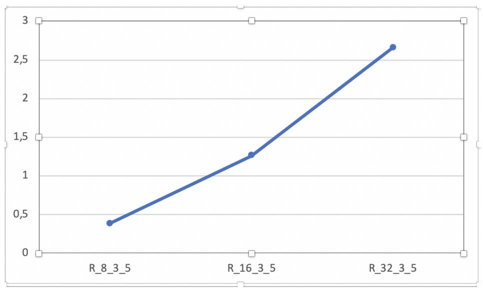
3.2 The N-Neuron MLP



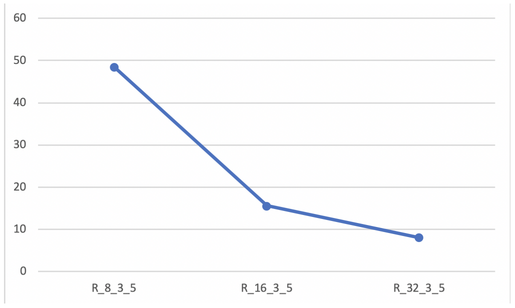
(a) Area (nm) vs. Bit Width.



(b) Cells vs. Bit Width.



(c) Power (mW) vs. Bit Width.



(d) Max Frequency (MHz) vs. Bit Width.

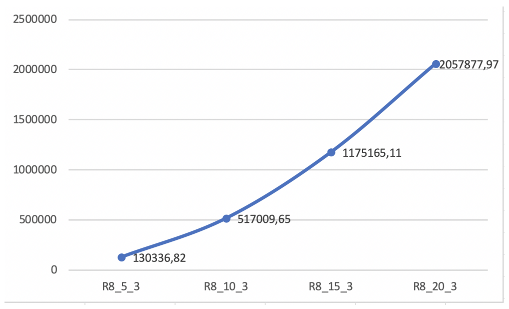
Figure 3.5: Effect of changing the data precision (8, 16 and 32bits) for a MLP with one-neuron model with constant number of inputs, constant number of layers and ReLU as the activation function.

Number of bits	Power	Number of Cells	Total cell area:	Max clock frequency
8	0.3808 mW	289	47565.50 nm	48.43 MHz
16	1.2584 mW	914	249963.03nm	15.6MHz
32	2.65211 mW	1913	750260.05nm	8.1MHz

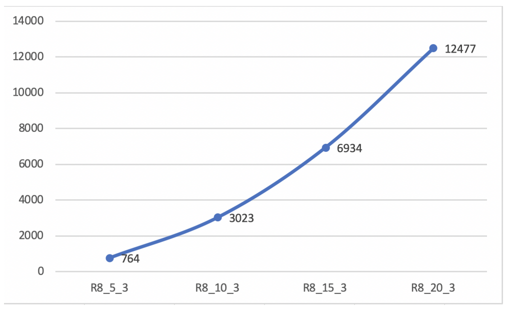
Table 3.5: The reported values for 5-layer N-neuron MLP with 3 neurons per layer with varying bit width and ReLU as activation function

Activation Function	Power	Number of Cells	Total cell area:	Max clock frequency
Step	0.5948 mW	455	78323.21nm	30.0MHz
ReLu	0.8134 mW	453	78711.90nm	28.6MHz
Sigmoid	0.7622 mW	455	78558.68nm	29.8MHz

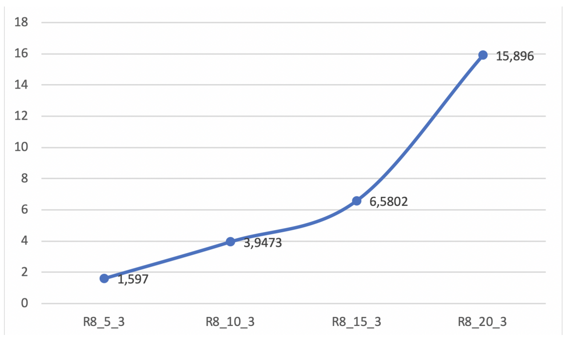
Table 3.6: The reported values for 5-layered N-neuron MLP with 3 8-bit neurons per layer while varying the activation function



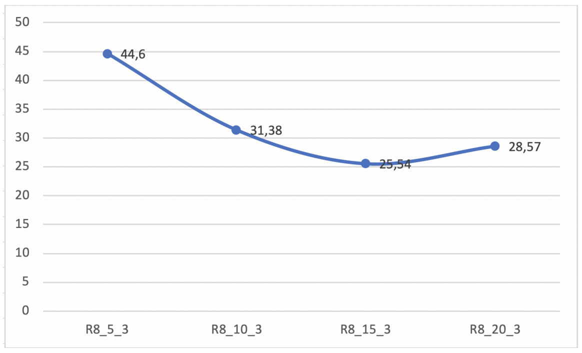
(a) Area (nm) vs Inputs.



(b) Cells vs Inputs.



(c) Power (mW) vs Inputs.

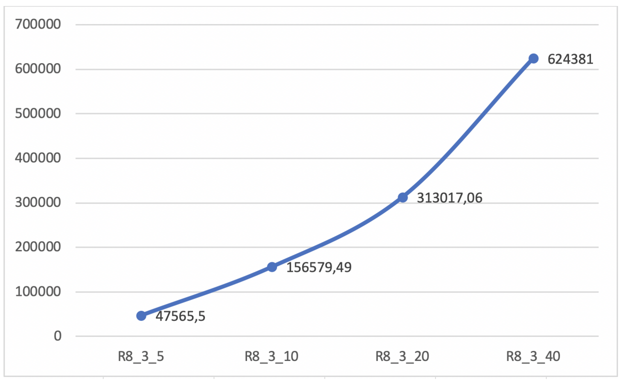


(d) Max Frequency (MHz) vs Inputs.

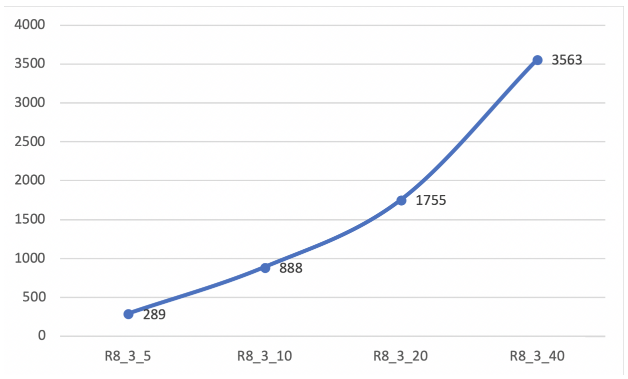
Figure 3.6: Effect of changing the number of neurons per layer for a N-neuron MLP with constant number of layers, constant bit-width and ReLU as the activation function.

Number of Inputs	Power	Number of Cells	Total cell area:	Max clock frequency
5	1.5970 mW	764	130336.82 nm	44.60 MHz
10	3.9473 mW	3023	517009.65 nm	31.38 MHz
15	6.5802 mW	6934	1175165.11 nm	25.54 MHz
20	15.8965 mW	12477	2057877.97 nm	28.57 MHz

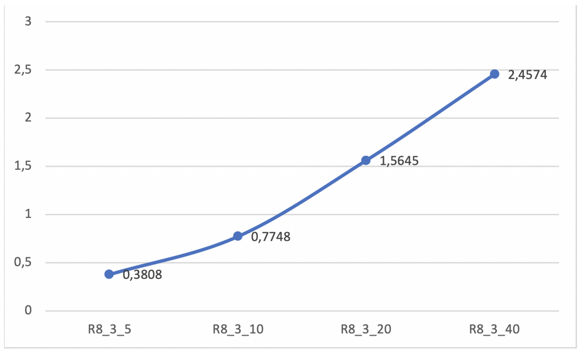
Table 3.7: The reported values for an 8-bit N-neuron MLP with 3 layers using ReLu as activation function while varying the number of neurons per layer



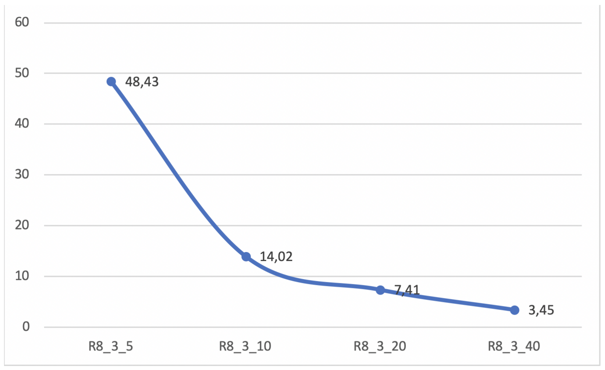
(a) Area (nm) vs. Layers.



(b) Cells vs. Layers.



(c) Power (mW) vs. Layers.



(d) Max Frequency (MHz) vs. Layers.

Figure 3.7: Effect of changing the number of layers for a N-neuron MLP with constant number of neurons per layer, constant bit-width and ReLU as the activation function.

Number of Layers	Power	Number of Cells	Total cell area:	Max clock frequency
5	0.3808 mW	289	47565.50 nm	48.43 MHz
10	0.7748 mW	888	156579.49 nm	14.02 MHz
20	1.5645 mW	1755	313017.06 nm	7.41 MHz
40	2.4574 mW	3563	624381.00 nm	3.45 MHz

Table 3.8: The reported values for an 8-bit N-neuron MLP with varying number of layers using ReLu as activation function and 3 neurons per layer

4 Analysis and discussion

In the following section a analysis of the data presented in the result section is carried out. First a discussion of the two different designs with a following subsection where the two design are compared.

4.1 Trends in One-Neuron MLP

4.1.1 Area & Cells

From the cell report in the Weight Memory in Appendix we can see the weights of the MLP contributed to the maximum Area also since it is implemented as a 4D array of signals, it raises the complexity of the design interconnects exponentially at the higher number of inputs and layers.

4.1.2 Power

We see the Power follows closely to the area and cells values and this is expected. Moreover, looking closer to the Power report in Appendix we see the internal power of the registers had been the major contributor to the overall power and we can conclude maximum power of the circuit was consumed in the Weight Memory.

4.1.3 Frequency

Frequency decreases almost linearly with number of inputs and exponentially with number of layers. Looking at the report we see the components used by Synopsys were different and the model with higher number of inputs had been buffered which may have contributed to improved frequency of those models, this variation can be contributed to Synopsys's optimization algorithm.

4.1.4 Throughput

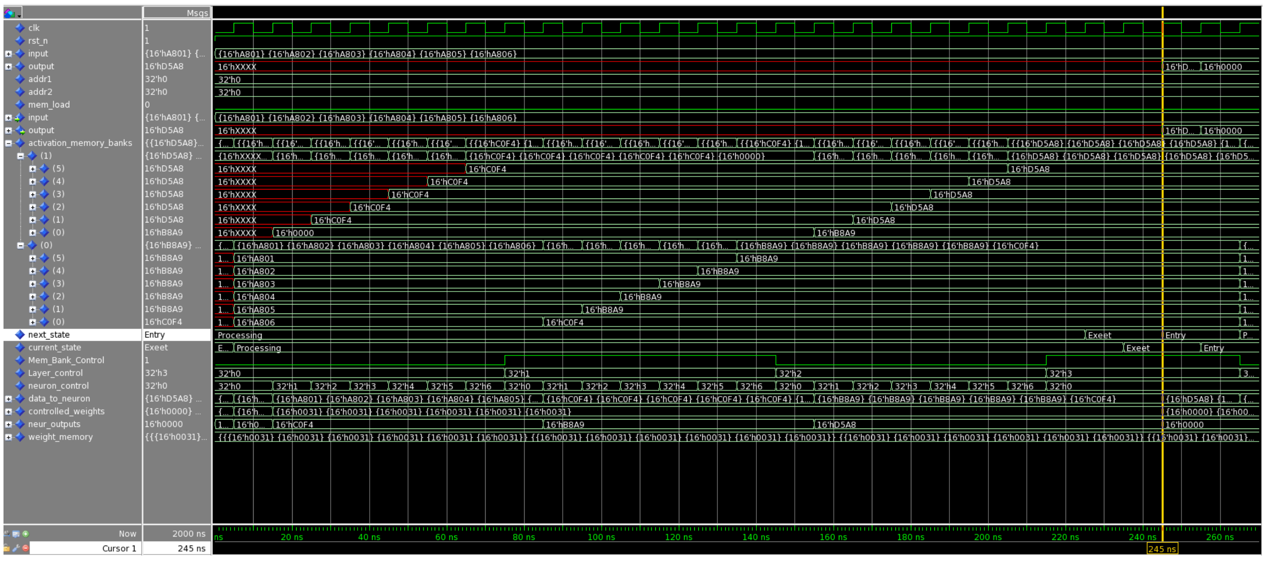


Figure 4.1: Block diagram for a One-Neuron MLP

The model shown in Figure 4.1 is simulated with 6 inputs and 3 layers and bit-width of 16. The model takes 6 clock cycles to process each value of the memory banks as described in Section 2.2 and since it has 3 layers it does the process 3 times. Ignoring overheads of state-changes we can say the model will provide the result after

$$M * N \quad Cycles \tag{1}$$

Where M is the number of neurons per layer and N is the number of layers in the network.

4.2 Trends in N-Neuron MLP

4.2.1 Area & Cells

The number of cells and area scales with the number of layers and number of neurons per layer. The majority of the area is used for the memory holding the weights.

4.2.2 Power

The power of the parallel MLP scales quite linear with the number of neurons. From the power reports one can see that most of the power is due to combinational logic. The power that is related to registers are mainly due to the memory holding the weights.

4.2.3 Frequency

The maximum clock frequency was calculated using the inverse of the data arrival time, i.e, from a valid input to a valid output. The maximum clock frequency that the parallel MLP can operate with becomes smaller when the amount of neurons per layer increases as well when the number of layers increases. This effect is expected and can be understood by examining the architecture of the parallel MLP. The architecture does not contain any registers between the different layers, i.e the data path is fully combinatorial. This will make the critical path longer for each added layer. From table 3.7 one can see that increasing the number of neurons per layer does not effect the maximum clock frequency as much as adding layers.

4.2.4 Throughput

The data path trough the parallel MLP is fully combinational, and does not have any registers between the input and the output. This means that the output will be produced in one clock cycle. Scaling up the network will not change the throughput since no sequential logic is added.

4.2.5 Memory System

Each neuron is loaded with its corresponding weights at the rising edge of the clock cycle that start the computation at the first layer. If the design would have a constraint where this would not be possible and instead one would have to load the weights in different clock cycles one would have to make some changes to the structure. One solution is to first load the weights and then start the actual computation. This would require two clock domains. One that is used during the setup of the weights and one for starting the computation and storing the output. One could also infer registers between the layers and create a pipelined architecture. Then the weights for the next layer could be loaded while the current layer where performing computations.

4.3 One-Neuron MLP vs. N-Neuron MLP

4.3.1 Maximum clock frequency

There is a substantial difference between the maximum clock frequency that the two design can use and how it scales when adding neurons. This was expected since the parallel MLP does the calculations in a layer fully in parallel while the one-neuron neuron does it in a serial fashion. The parallel MLP has a fully combinational data path so the critical path is considerably shorter. The critical path of the one-neuron MLP will not change as much as the parallel when increasing the number of neurons in the design. The parallel MLP might will produce a result faster than the one-neuron despite the fact that the clock period is longer. So the latency of the parallel MLP is shorter than the one-neuron. One extension of the parallel MLP is to pipeline the architecture so that each layer can perform calculations in parallel. This can be done by inserting registers between the layers. This would decrease the clock period and increase the throughput. This would also decrease the latency.

4.3.2 Area

The N-neuron MLP have a greater area than the One-neuron implementation. This was expected since the one-neuron MLP reuses the hardware resources for computation while the N-neuron MLP have exlusive hardware resources for each neuron. One can see that the difference between the area of the two implementations is smaller when the amount of neurons in the design is smaller. Most of the area used by both designs are due to the memory containing the weights. Since the N-neuron architecture will add hardware resources for computations for each neuron and more memory for weights while the one-neuron implementation will mostly just add more memory one can see that the area of the N neuron design grows more than the one-neuron when increasing the amount of neurons

4.3.3 Power

The N-neuron MLP consumes significantly less power than the one-neuron MLP. It is not really clear why this is the case. The reports generated by synopsys shows that the internal power of registers is what makes up the major difference between the two architectures. The internal power is due to changes in the input of a cell that does not effect the output of the cell and the dynamic power dissipated within the boundary of a cell. It might be due to the state machine logic. The power of the N-neuron MLP is mostly dissipated by combinational logic

4.3.4 Scalability

From the analysis above one can see that the one neuron MLP scales better than the parallel in terms of area. The N-neuron has a lower lataency and scales better in terms of power. The one neuron is therefor recommended for larger networks when the area of the chip is limited. The N-neuron design is more useful for networks where the latency and power dispation is the most important factor.