An Analog CMOS Implementation for Multi-layer Perceptron With ReLU Activation

Chao Geng, Qingji Sun, and Shigetoshi Nakatake
Information and Media Engineering Department
The University of Kitakyushu
Fukuoka, Japan
E-mail: naka-lab@kitakyu-u.ac.jp

***Abstract*—This paper presents an analog circuit comprising a multi-layer perceptron (MLP) applicable to the neural network(NN)-based machine learning. The MLP circuit with rectified linear unit (ReLU) activation consists of 2 input neurons, 3 hidden neurons, and 4 output neurons. Our MLP circuit is implemented in a 0.6μm CMOS technology process with a supply voltage of ±2.5V. An experimental case is conducted to demonstrate the feasibility and effectiveness of the MLP circuit. The simulation results show that our MLP circuit has a power dissipation of 200mW, a wide range of working frequency from 0 to 1MHz, and a moderate performance in terms of the error ratio.**

***Index Terms*—multi-layer perceptron, neural network, ReLU activation, neuron.**

I. Introduction

Artificial neural network(ANN)-based machine learning, known as a promising technology, has been researched widely to enable the electronic devices more intelligent and efficient [1]. ANN is inspired by the brain of living creatures which contains components such as neurons, connections, weights, and propagation function. ANN has a huge set of neurons that are highly interconnected and arranged in layers. Its structure mimics the function of dendrites, soma, and axon. Dendrites serve as receiving inputs and are equivalent to the weighted connections between neurons in ANN. Soma collects input signals and generates an output, the output will result in a response when the neuron cell has crossed the threshold, the process is equivalent to the neuron and activation function in ANN. Axon transmits an output signal to the dendrites of other neurons in the subsequent layer, and is equivalent to the connection between the hidden and output layers. Perceptron is an essential element of ANN, which has multiple inputs and a single output. It's commonly represented by a mathematical model as follows: each input is multiplied by a weight and all weighted inputs are summed at the output, the resulting sum is then passed through an activation function. ANN-based machine learning fits the transfer function of the system by a training process where input-output pairs are iteratively presented, while the variable parameters/weights are adjusted. The MLP is constituted by perceptron which is a fundamental structure for the feedforward NN, in the VLSI (very-large-scale integration) implementations incorporating various learning algorithms, MLP is a common choice as it has been continuously researched many years [2].

On the other hand, Ishiguchi et al. proposed an analog perceptron circuit with DAC-based multiplier in the work [3] aiming at the advanced sensor nodes, such as the biological sensor systems introduced in [4] and [5]. In comparison with the traditional sensor system where the central processing unit and the signal processing unit are necessary. Analog VLSI implementations are preferable for a sensor node as they have lower power and smaller device area. The analog MLP has been becoming popular in recent years for the NN-based machine learning. Several Analog MLPs have been implemented to solve classification problems in the past decade, which show promising results [6]–[8]. However, the MLP realized by analog circuits is vulnerable to many factors, such as offset voltage, noise coupling, impedance, limited scalability and process-introduced variation [9]. The accumulated effect will degrade the feasibility of the MLP circuit. Moreover, most of the implementations adopt complicated circuits for the activation function which are difficult to train, the weights are also not easy to control.

Based on an analog perceptron with DAC-based multiplier, we implement an MLP circuit with ReLU activation in a 0.6μm CMOS technology process with a supply voltage of ±2.5V. The circuit utilizes an improved source follower with a simple structure to approximate the ReLU function greatly. To improve the reliability of the whole circuit so that the DC biasing of the ReLU circuit is adjustable, one adder and an inverter are inserted after each perceptron. In addition, we present the impedance issue in the cascading of neurons in MLP, which is critical to the feasibility of the whole circuit. Finally, the simulation results in an experimental case of our analog MLP circuit, shows that the circuit has a power dissipation of 200mW, a wide range of working frequency from 0 to 1MHz, and a moderate performance in terms of the error ratio.

The rest of the paper is organized as follows. Section II briefly describes analog perceptron circuit with DAC-based multiplier. Section III presents an MLP circuit for the machining learning and introduces some aspects critical to the success of the implementation. Section IV explains the simulation of the whole circuit and summarizes the simulation results. Finally, Section V concludes this work.

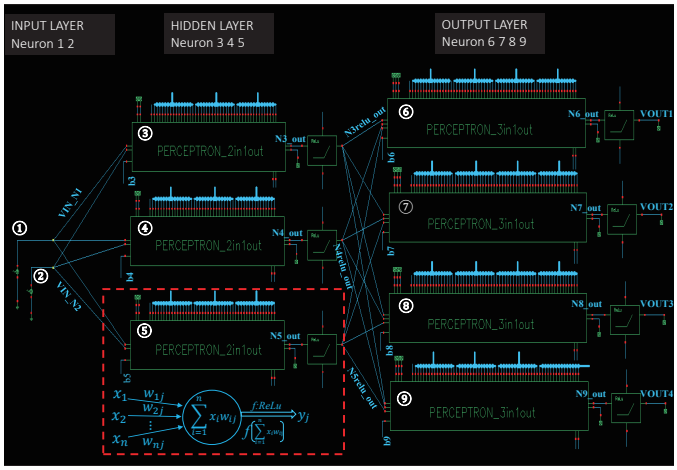


Fig. 1. Top-level schematic of our MLP circuit

II. Analog Perceptron Circuit with DAC-based Multiplier

Aiming at an analog front-end of a sensor node, Ishiguchi et al. propose an analog perceptron circuit with DAC-based multiplier in the work [3] to improve the sensing information. The analog perceptron circuit employs the mechanism of neural network-based machine learning, the model of which can be represented as: $f_{out}(t) = w_1 \cdot f_1(t) + w_2 \cdot f_2(t) + \cdots + w_n \cdot f_n(t)$. Where $f_1(t), \cdots, f_n(t)$ are the inputs, each of which is multiplied by a weight $w_1, \cdots, w_n$, all weighted inputs are summed at the output $f_{out}(t)$. All the weights are controlled only by digital codes. Two DACs are used in a DAC-based multiplier, a DAC is inserted at the input of the negative feedback circuit, while the other is at the loop of the feedback. DACs serve as two variable resistors and changing of the DAC output current looks as if the resistance value were to change. The input-output/$V_{in} - V_{out}$ relationship of the multiplier can be represented as: $V_{out} = -\frac{X1}{X2}V_{in}$, where X1 and X2 are the decimal input codes of DAC1 and DAC2, respectively. For a three-input perceptron circuit, the input-output/$V_{in} - V_{out}$ relationship of the circuit can be represented as: $V_{out} = -\frac{X1}{X4}V_{in1} - \frac{X2}{X4}V_{in2} - \frac{X3}{X4}V_{in3}$, where X1, X2, X3 are the decimal codes of DACs for the input V_{in1}, V_{in2}, V_{in3}, respectively. While X4 is the decimal code of DAC in the negative feedback of OPAMP. Note that in our implementation of an MLP circuit, a non-inverting adder is used to collect the outputs of multipliers rather than an OPAMP in order to reduce the interaction. An OPAMP-based inverter is also used to invert the output of the perceptron circuit, so that each output of the node is positive and easy to verify the correctness.

III. Multi-layer Perceptron Based Circuit

In this section, a multi-layer perceptron circuit with a non-linear activation function is introduced. Once a neural network model is trained by the back-propagation algorithm in the software, the weight value of each connection can be determined. As long as ensuring the configuration and weights are consistent with the software, our circuit can be used to recover a learning model on hardware. In this paper, we primarily focus on the implementation of the circuit, the software part is beyond the scope of this paper.

A block diagram of the MLP circuit is shown in the Fig. 1. This network is constituted by three layers: an input layer with two neurons, a hidden layer with three neurons, and an output layer with four neurons. The neuron number is annotated in the corresponding module, the net name is annotated as well and the connection relationship is indicated by its name directly. Each neuron in hidden and output layers is followed by a ReLU module, which is used to approximate the ReLU function in the neural network. The function of a perceptron and ReLU circuit can be represented by a mathematical model, as shown in the dotted box of the figure. Where w_{ij} denotes the weight for the connection of two neurons. The subscript i represents the sequence number of the starting neuron, and j represents that of the target neuron. For example, the weights of the sixth neuron are w_{36}, w_{46} and w_{56}, respectively. x_i denotes the input of the j-th neuron, and y_j denotes the output of the j-th neuron. The perceptron module is comprised of three or two DAC-based multipliers and other function circuits. The ReLU module is comprised of an improved source follower and an OPAMP-based buffer. The weight of each perceptron module is controlled by digital codes.

A. An Improved Source Follower

ReLU is the most commonly used activation function in neural networks, especially in convolutional neural networks (CNNs). Mathematically, it is defined as: $y = max(0, x), x \in R$. Inspired by the input-output characteristic curve of a source follower, we attempt to implement a ReLU circuit by it since the curve is approximately similar to ReLU function. However, the output voltage of a typical source follower(see Fig. 2(a)) always has a significant difference with the input voltage, which reduces the accuracy of the neural network model and prediction correctness. Therefore, we modify the

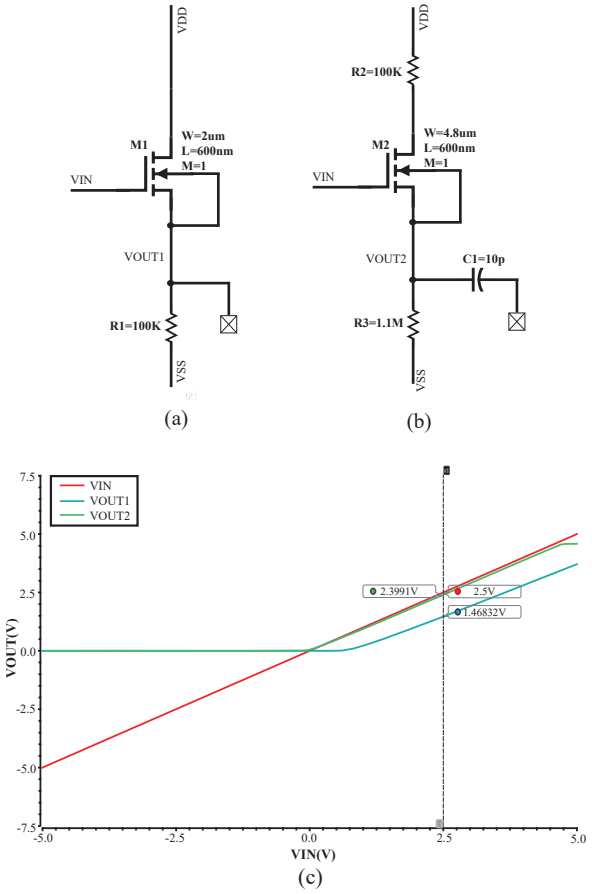


Fig. 2. Circuits and simulation for the ReLU activation function

circuit and its configuration is shown in Fig. 2(b). The value of each component is annotated as well. When a proper DC operating voltage is applied at the input, the NMOS transistor works in a saturation condition. The DC operating voltage applied influences not only the drain current of M2, but also the linearity of input-output characteristic. Hence, a proper DC biasing is important to approximate the ReLU function. In the preceding stage of ReLU circuit, we have a non-inverting adder that incorporates a DC biasing, which can adjust the DC operating point of ReLU circuit. The input-output characteristic of the above two circuits is shown in the Fig. 2 (c). An improved source follower (b) can better approximate the ReLU function, as the difference between the input and output is smaller.

Note that both the gain of circuits in figure is not larger than 1, even if the R1 and R3 are close to infinite. When R3 reaches a certain value, increasing the value of R3 only contributes a little to the slope of the curve. Capacitor C1 is used to filter out DC voltage so as not to influence the DC operating point of the next stage. Besides, the voltage loss on ReLU circuit is inevitable as output cannot perfectly follow the input. We also note that when the input voltage for (b) is beyond 4.7V, the output voltage does not increase, that's

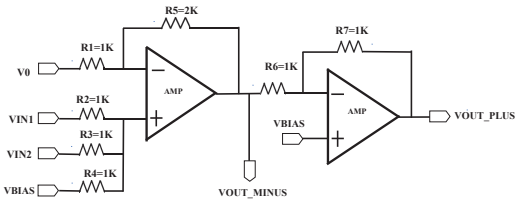


Fig. 3. Adders inside our perceptron module

because as the gate-source voltage increases, the number of carriers in the transistor channel no longer increases, and the drain current tends to be constant.

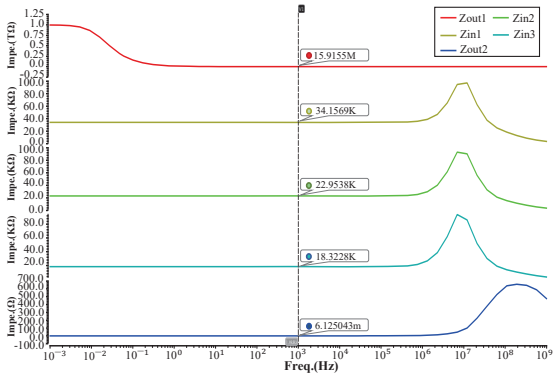
B. Adders for Improving Reliability

Inside the perceptron module of the top-level schematic, there are DAC-based multipliers and adders. The circuit shown in the Fig. 3 is a sub-circuit of a 2-input perceptron, which is constituted by a non-inverting adder and an OPAMP-based inverter. The port name and the value of all resistors are annotated in the figure. In our implementation of the MLP circuit, V0 connects to ground, VIN1, and VIN2 connect to the outputs of multipliers, respectively. VBIAS connects to a biasing generator circuit which provides various biases we need. It can serve as a bias in the mathematical model of the neural network. By utilizing the bias, the centering voltage of the summed signals can be adjusted. Besides, the DC biasing can be also used to bias the ReLU circuit which follows after the perceptron, so as to adjust its operating point. Note that the bias of each ReLU can be different and independent, therefore the reliability and flexibility of MLP is improved.

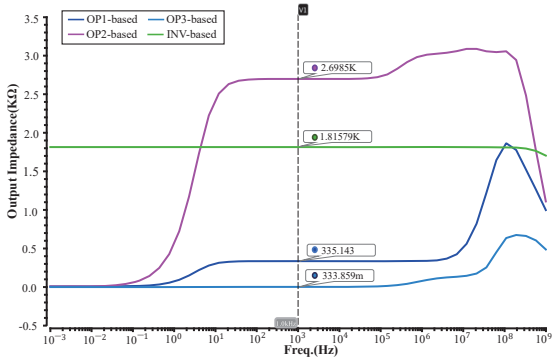
The OPAMP-based inverter is used to invert the phase of the summed signals. As a perceptron circuit with DAC-based multiplier always generates an inverting signal, such as $Vout = (-X1/X2)Vin1 + (-X3/X4)Vin2$, here, X represents the decimal value of the digital codes of DAC. It's hard to verify the output of each neuron in this form, especially in a complicated neural network with multiple hidden layers. Therefore, we use an OPAMP-based inverter to achieve that the output of neurons always has the same phase with the input, so that the calculation at each neuron becomes easy. Note that the positive port of the OPAMP-based inverter connects to VBIAS rather than the ground. That's because the non-inverting adder has a DC component at its output.

C. Impedance Issue of Cascading Neurons

Since the output impedance of the ReLU circuit is extremely high, therefore it has a poor ability to drive its succeeding stage. The high output impedance of the ReLU circuit results in not working when combining two neurons. Consequently, it's critical to deal with the impedance issue when cascading neurons. In our implementation of an MLP circuit, we measure the output impedance of a ReLU circuit following a 2-input



(a)



(b)

Fig. 4. Impedance issue of cascading. (a) Comparasion between the output impedance and the input impedance. (b) The output impedance of the circuit after inserting buffers.

perceptron, and the input impedance of a 3-input perceptron which follows the ReLU circuit. As shown in the graphs of Fig. 4(a), Zout1 represents the output impedance of a ReLU circuit; Zin1, Zin2, and Zin3 represent the input impedance of three inputs, respectively; Zout2 represents the output impedance of ReLU circuit inserting an OPAMP-based buffer after it. When the circuit works at 1KHz, before inserting a buffer, the output impedance is very high that reaches 15.9MΩ, whereas the input impedance of three inputs is 34.1KΩ, 22.9KΩ, and 18.3KΩ, respectively. It's far smaller than the output impedance of the ReLU circuit, therefore ReLU cannot drive the next stage if without dealing with impedance issue. In dealing with the impedance issue of cascading circuits, inserting a buffer between two stages is an effective way. As seen in the graph Zout2, in our implementation of the circuit, the output impedance is significantly reduced after inserting a buffer, which has only 6.1mΩ and load-carrying capacity has been improved.

However, it's not any buffer can alleviate the issue of high output impedance. We need to design the output impedance of the buffer carefully. Fig. 4(b) shows the measuring results of the output impedance of four buffers. There are OP1-based, OP2-based, OP3-based and INV-based buffers. When working at 1KHz, their output impedance is 335.1Ω, 2.7KΩ, 333.8mΩ and 1.8KΩ, respectively. Finally, we choose OP3-based buffer as it has the lowest output impedance. Besides, it has a stable frequency characteristic ranging from 0-1MHz. However, the other three buffers cannot enable the MLP circuit to work even inserting them after a ReLU circuit. Note that ReLU circuit inserting with an OP3-based buffer is packed inside the ReLU module in the top-level schematic.

Furthermore, from both figures of the measuring results, it shows that impedance of the circuit is relevant to the working frequency, as the impedance value of the circuit is changed dramatically when the frequency is beyond 1MHz. It implies that our MLP circuit is constrained by the working frequency.

IV. Experimental Case

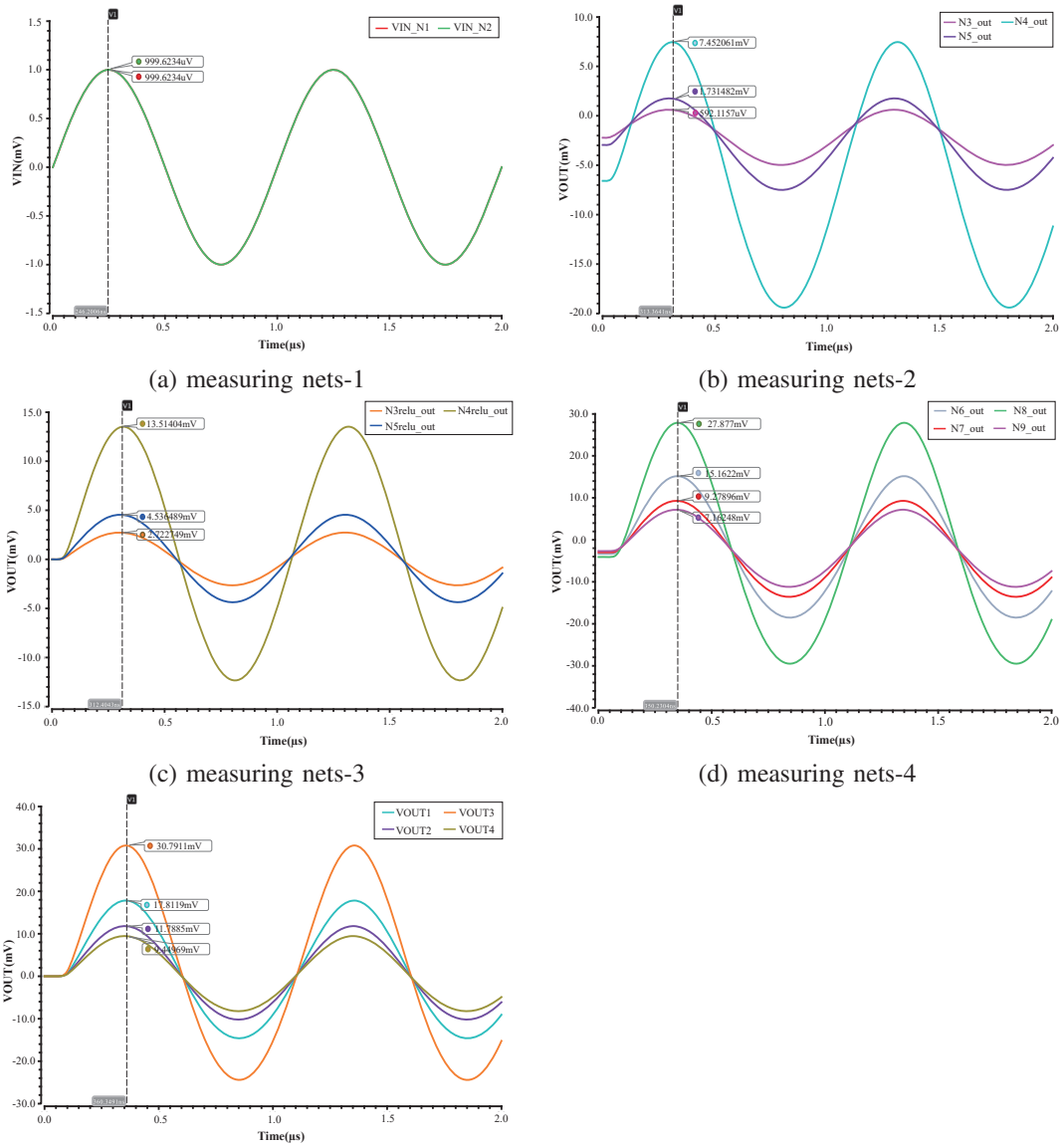
A. Simulation Explanation

After each module of the MLP circuit is tuned well, we conduct a system simulation for the top-level schematic. Once a learning model is obtained and each weight of connection is determined, the digital circuit sets the weight of the MLP circuit to a corresponding value consistent with the software. Subsequently, two test signals are applied to the inputs. Each measuring net of the MLP circuit is plotted and the amplitude of the waveform is recorded. Since the result of each measuring net is a simple weighted sum, the measuring result of simulation can be used to verify the behavior of a learning model on hardware easily, the correctness and the feasibility of our MLP circuit can be demonstrated as well.

In our system simulation, we apply two same signals with the amplitude of 1mV and the frequency of 1MHz at inputs, and measure the points of interest on the nets of the top-level schematic. As seen in the Fig. 5, we obtain five sets of graphs, there are corresponding to the measuring nets of inputs(see Fig. 5(a)), neuron outputs of the hidden layer(see Fig. 5(b)), ReLU outputs of the hidden layer(see Fig. 5(c)), neuron outputs of the output layer(see Fig. 5(d)), and ReLU outputs of the output layer(see Fig. 5(e)), respectively. The legend of each set of graphs is directly corresponding to the net name on the top-level schematic. The amplitude of each graph is annotated in the figure and recorded in a table in the after-mentioned subsection. As seen from the figure, each measuring net has sinusoid outputs, which implies that our MLP circuit can work. In the following subsection, we introduce the performance of our circuit. When comparing the fig. (c) to the fig. (b), and the fig. (e) to the fig. (d), we find that the offset of the outputs is eliminated, this is because the capacitor following after the improved source follower filters out the offset.

B. Summary and Discussion of MLP Circuit

The specifications and weights of our MLP circuit are summarized in Table I. Our entire circuit is designed in a 0.6μm CMOS technology process with a supply voltage of ±2.5V. The working frequency of our circuit ranges from 0 to 1MHz, which can satisfy the most of applications. In the implementations of the MLP circuit, the interaction of neurons is complicated, such as the impedance issue in



(a) measuring nets-1

(b) measuring nets-2

(c) measuring nets-3

(d) measuring nets-4

(e) measuring nets-5

Fig. 5. Simulation results for the top-level schematic

cascading, and the unit-gain frequency of OPAMP, both can influence the working frequency of the entire circuit. In terms of power dissipation, it shows a moderate performance with a power dissipation of 200mV, as we adopt not an advanced technology process considering the cost. However, it's still far smaller than most of FPGA(field-programmable gate arrays)- or GPU(graphics processing units)-based implementations. Based on our experience to implement layout for a perceptron in a 0.6µm PHENITEC technology, the expected area of the MLP circuit is approximately $1.69\mathrm{mm}^2$. With respect to the weight value, we list them in the form of two columns by referring to the top-level schematic, they are corresponding to the weight value of neurons in the hidden layer, and neurons in the output layer, respectively. The weight is set to the value consistent with the software by configuring the digital codes of each perceptron. For example, the 6th neuron in our MLP circuit has weights of w_{36}=3, w_{46}=0.6, and w_{56}=0.4, the subscript 3, 4, 5, represent the starting points, the third, the 4th, the 5th neuron, respectively. The subscript 6 represents the target point, the 6th neuron. Similarly, the other weights follow the same rule. In this experiment, all of biases for neurons are set to 0 for the simplicity of demonstration. However, we can insert an non-inverting adder at the final output of the MLP circuit, adding the amount generated by the weights. Since the bias in each neuron is a constant in the neural network model, the total amount generated by the network is also a constant.

Table II shows the amplitude of the measuring nets, their corresponding mathematical calculation, and the error ratio.

TABLE I
THE SPECIFICATIONS AND WEIGHTS OF OUR MLP CIRCUIT.

Technology process	0.6μm CMOS	
Supply voltage	±2.5V	
Working frequency	0-1MHz	
Power dissipation	200mW	
Expected area	1.69mm^2	
Weight value	w_{13}=2, w_{23}=1	w_{36}=3, w_{46}=0.6, w_{56}=0.4
	w_{14}=5, w_{24}=10	w_{37}=2, w_{47}=0.25, w_{57}=0.75
	w_{15}=3, w_{25}=2	w_{38}=3, w_{48}=1, w_{58}=2
	n/a	w_{39}=0.83, w_{49}=0.16, w_{59}=1.16

TABLE II
EVALUATION TO OUR MLP CIRCUIT.

Measuring net	Mathematical calculation (mV)	Simulation result (mV)	Error ratio (%)
N3ReLU_out	3	2.72	9.3
N4ReLU_out	15	13.51	9.9
N5ReLU_out	5	4.54	9.2
VOUT1	20	17.81	11.0
VOUT2	13.5	11.79	12.7
VOUT3	34	30.79	9.4
VOUT4	10.83	9.45	12.7

We omit some measuring nets and only list the values at the outputs of the ReLU circuits, as the omitted measuring nets are not final outputs of neurons and also have the offset. In the following, we give an example of the mathematical calculation for a measuring net. For the net N3ReLU_out at the output of the ReLU circuit after the third neuron, which has weights of w_{13}=2, w_{23}=1, thus, the output is as: $y_3 = x_1 \cdot w_{13} + x_2 \cdot w_{23}$. Since the amplitudes of x_1 and x_2 are both 1mV in our implementation, the amplitude of y_3 is 3mV. Ideally, the amplitude of y_3 is the final result for the net N3ReLU_out if the circuit can realize the ReLU function perfectly. In the same way, we can calculate that the output of the net N4ReLU_out is 15mV and that of the net N5ReLU_out is 5mV. After all of the outputs in the hidden layer is obtained, using them as inputs of next layer. The output of neuron in the output layer is equal to that its inputs are multiplied by corresponding weights and are then summed up, the output for the 6th neuron is as: $y_6 = x_1 \cdot w_{36} + x_2 \cdot w_{46} + x_3 \cdot w_{56}$. Here x_1, x_2, and x_3 denote the inputs of the 6th neuron, the amplitude of y_6 is 20mV, the final result for the net VOUT1 is also 20mV under the assumption of the ideal circuit. Similarly, the outputs for the nets VOUT1, VOUT2, and VOUT3, are 13.5mV, 34mV, and 10.83mV, respectively. Note that all the outputs of neurons are positive, as the OPAMP-based inverter inside perceptron module inverts the summed signal. The amplitude for each measuring net is listed in Table 2, and the error ratio is given accordingly. The error ratio is calculated by the difference between the mathematical calculation and simulation result divided by the mathematical calculation. Our MLP circuit shows a moderate performance in terms of the error ratio. On the one hand, the ideal ReLU function and the approximated function generated by circuit have an difference which is inevitable. Besides, the interaction between the analog circuits is complicated especially in a highly-interconnected neural network, designers need to trade off in the consideration of many aspects, such as noise, offset, impedance, the input and output range.

When compared to most of VLSI implementations based on FPGAs or GPUs with very high power dissipation, this work demonstrates the feasibility and effectiveness of our MLP circuit implemented on an analog CMOS circuit.

V. CONCLUSION

Based on an analog perceptron circuit with DAC-based multiplier, this paper presents an MLP circuit with ReLU activation for the neural network. Our MLP circuit is implemented in a 0.6μm CMOS technology process with a supply voltage of ±2.5V. We propose an improved source follower to greatly approximate the ReLU activation function. One adder and an inverter are adopted to improve the reliability of the whole circuit, so that the DC biasing of the ReLU circuit is adjustable. In addition, the impedance issue in the cascading of neurons in MLP is presented, which is critical to the feasibility of the whole circuit. Finally, an experimental case is conducted on our MLP circuit, the simulation results show that the circuit has a power dissipation of 200mW, a wide range of working frequency from 0 to 1MHz, and a moderate performance in terms of the error ratio. It demonstrates the feasibility of our MLP circuit, and shows a promising function of a learning model implemented on an analog CMOS circuit. Our future works are to improve the DAC of the multiplier to expand the range of the weight value, and reduce the interaction between neurons to improve correctness.

REFERENCES

[1] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, 2019.

[2] J. B. Lont and W. Guggenbühl, "Analog cmos implementation of a multilayer perceptron with nonlinear synapses," *IEEE Trans. Neural Networks*, vol. 3, no. 3, pp. 457–465, 1992.

[3] Y. Ishiguchi, D. Isogai, T. Osawa, and S. Nakatake, "Analog perceptron circuit with dac-based multiplier," *Integration*, vol. 63, pp. 240–247, 2018.

[4] J. Xu, S. Mitra, A. Matsumoto, S. Patki, C. Van Hoof, K. A. Makinwa, and R. F. Yazicioglu, "A wearable 8-channel active-electrode eeg/eti acquisition system for body area networks," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 9, pp. 2005–2016, 2014.

[5] C.-L. Goh and S. Nakatake, "A sensor-based data visualization system for training blood pressure measurement by auscultatory method," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 4, pp. 936–943, 2016.

[6] C.-H. Pan, H.-Y. Hsieh, and K.-T. Tang, "An analog multilayer perceptron neural network for a portable electronic nose," *Sensors*, vol. 13, no. 1, pp. 193–207, 2013.

[7] T. Zhang, Y. Cao, F. Ye, and J. Ren, "Use multilayer perceptron in calibrating multistage non-linearity of split pipelined-adc," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2018, pp. 1–5.

[8] F. M. Bayat, M. Prezioso, B. Chakrabarti, H. Nili, I. Kataeva, and D. Strukov, "Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits," *Nature communications*, vol. 9, no. 1, p. 2331, 2018.

[9] M. Heidari and H. Shamsi, "Analog programmable neuron and case study on vlsi implementation of multi-layer perceptron (mlp)," *Microelectronics Journal*, vol. 84, pp. 36–47, 2019.