Indonesian Journal of Electrical Engineering and Computer Science
Vol. 14, No. 2, May 2019, pp. 949~956
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v14.i2.pp949-956

FPGA-based architecture of hybrid multilayered perceptron neural network

Lee Yee Ann[1], P. Ehkan[2], M.Y. Mashor[3], S.M. Sharun[4]
[1,2]School of Computer and Communication Engineering, Universiti Malaysia Perlis, Pauh Putra Campus, Malaysia
[3]School of Mechatronic Engineering, Universiti Malaysia Perlis, Pauh Putra Campus, Malaysia
[4]Faculty of Innovative Design and Technology, Universiti Sultan Zainal Abidin, Gong Badak Campus, Malaysia

Article Info

Article history:

Received Sep 22, 2018
Revised Dec 25, 2018
Accepted Jan 14, 2019

Keywords:

Artificial Neural Network
FPGA-based architecture
Hybrid Multilayered Perceptron
Neural Network
VHDL

ABSTRACT

The HMLP is an ANN similar to the MLP, but with extra weighted connections that connect the input nodes directly to the output nodes. The architecture of the HMLP neural network for implementation on FPGA is proposed. The HMLP architecture is designed to be concurrent to demonstrate the parallel nature of the HMLP where each hidden or output node within the same hidden or output layer of the HMLP can calculate its output independently. The HMLP architecture is designed to be modular as well, such that if modification to a module is necessary, only the specific module need to be modified and all other modules can be retained. This modularity will be especially helpful when different activation function is to be swapped in to replace current activation function. All calculations in the HMLP are performed in floating-point arithmetic. The HMLP architecture is compiled, simulated and finally implemented on the Cyclone V FPGA of DE1-SoC board. The simulation outcome and FPGA outputs showed that the developed HMLP architecture is able to calculate correct output values for all test datasets.

Corresponding Author:

Lee Yee Ann,
School of Computer and Communication Engineering,
Universiti Malaysia Perlis, Pauh Putra Campus,
02600 Arau, Perlis, Malaysia.
Email: leeyee4nn@gmail.com

1. INTRODUCTION

Various types of Artificial Neural Networks (ANN) had been applied for many kinds of applications, of which the Multilayered Perceptron (MLP) neural network is very popular. The MLP is made up of multiple layers of simple processing elements called nodes. The layers of MLP are one input layer, one output layer, and one or more hidden layers. The nodes and layers of MLP are arranged in a feedforward arrangement [1]. The MLP has a fully connected structure where each node is connected to every other nodes of next layer via weighted connections. The MLP's hidden layers perform a non-linear mapping of the input layer to the output layer [2].

Hybrid Multilayered Perceptron (HMLP) is a type of ANN that is based on the popular MLP. The HMLP enhances the existing MLP's ability by having the input layer directly connects to the output layer through some weighted connections. The additional weighted connections of HMLP effectively form a linear model in parallel to the non-linear MLP [3]. The HMLP had been applied for different applications, such as heart anomaly detection using electrocardiogram (ECG) data [4], classification of acute leukemia disease [5] and forecasting of car speed [6]. The HMLP had also been implemented on hardware, the most recent hardware implementation of HMLP was on a Rabbit Core Module RCM4100 microcontroller [7].

Field Programmable Gate Array (FPGA) is a reconfigurable digital logic device that can be used for implementation of various digital systems. The concurrent nature of FPGA provides engineers a suitable platform to design and implement any concurrent digital system such as the MLP. A Xilinx Zynq FPGA-SoC device was used to implement a MLP for gas classification application in a Wireless Gas Sensor Network system [1] and for arrhythmia detection from electrocardiogram (ECG) signals [2]. The DE2-70 FPGA board, with an Altera Cyclone II FPGA device on board, was the platform of choice to compare the performance of an MLP implemented on hardware-based FPGA and a same MLP implemented on NiosII softcore processor on FPGA. The comparison result indicated that MLP implemented directly on FPGA hardware executes significantly faster than MLP which implemented on NiosII but at the expense of greater resource utilisation [8].

Based on existing works on implementation of MLP on FPGA, the prospect of using the FPGA as a platform to implement the HMLP seems very promising. From previous literature, no existing work on implementing HMLP directly on FPGA was reported [9]. This paper is a continuation of works reported in [9] and proposes an FPGA-based architecture for implementing the HMLP on an FPGA.

The Introduction should provide a clear background, a clear statement of the problem, the relevant literature on the subject, the proposed approach or solution, and the new value of research which it is innovation. It should be understandable to colleagues from a broad range of scientific disciplines. Organization and citation of the bibliography are made in Vancouver style in sign [1], [2] and so on. The terms in foreign languages are written italic (italic). The text should be divided into sections, each with a separate heading and numbered consecutively. The section/subsection headings should be typed on a separate line, e.g., 1. Introduction [3]. Authors are suggested to present their articles in the section structure: Introduction - the comprehensive theoretical basis and/or the Proposed Method/Algorithm - Research Method - Results and Discussion – Conclusion.

Literature review that has been done author used in the chapter "Introduction" to explain the difference of the manuscript with other papers, that it is innovative, it are used in the chapter "Research Method" to describe the step of research and used in the chapter "Results and Discussion" to support the analysis of the results [2]. If the manuscript was written really have high originality, which proposed a new method or algorithm, the additional chapter after the "Introduction" chapter and before the "Research Method" chapter can be added to explain briefly the theory and/or the proposed method/algorithm [4].

2. HYBRID MULTILAYERED PERCEPTRON NEURAL NETWORK

Figure 1 illustrates the structure of the HMLP [3]. The HMLP consist of 3 layers, one input layer, one hidden layer and one output layer. Being based on the MLP, the structure of HMLP greatly resembles MLP's structure with addition of several weighted connection from the input layer directly to the output layer. HMLP's additional weighted links is shown as dashed lines in Figure 1.

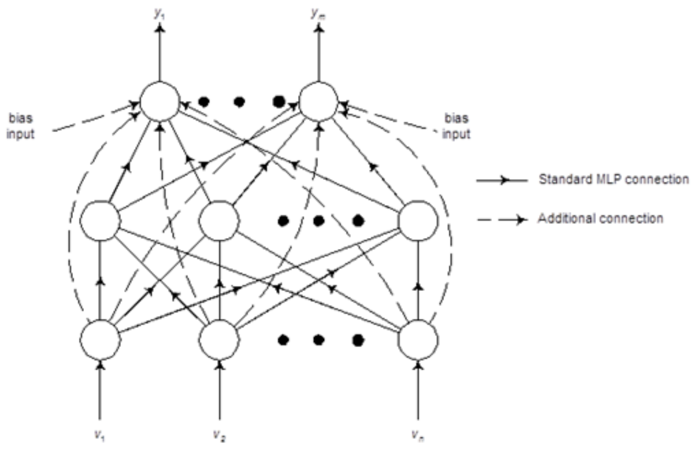

Figure 1. Structure of HMLP compared to MLP structure [3]

Based on the structure of HMLP, the output equation of a HMLP's k-th output neuron, $\hat{y}k$, with 1 hidden layer is given as (1).

$$\hat{y}_k = \sum_{j=1}^{n_h} w_{jk}^2 F\left(\sum_{i=1}^{n_i} w_{ij}^1 v_i^0 + b_j^1\right) + \sum_{i=1}^{n_i} w_{ik}^{\ell} v_i^0 + b_k^2 \text{ ; for } 1 < k < n_o \quad (1)$$

where n_i, n_h and n_o are the number of input nodes, hidden nodes and output nodes of the HMLP, respectively; w^1_{ij}, w^2_{jk} and w^ℓ_{ik} are the weights from input layer to hidden layer, the weights from hidden layer to output layer, and the weights of additional connection from input layer to output layer, respectively; b^1_j and b^2_k are bias input to the j-th hidden node and the k-th output node, respectively; and v^0 is the input to the i-th input nodes of the HMLP. $F(\cdot)$ is the activation function at the hidden nodes. In this paper, the activation function is selected as a sigmoidal function given as (2).

$$F = \frac{1}{1+e^{-x}} \tag{2}$$

The weights w^1_{ij}, w^2_{jk}, w^ℓ_{ik}, and biases b^1_j and b^2_k are unknown, and should be selected to minimise the prediction error. Similar to the MLP, HMLP's topology means that each node of the HMLP processes its respective output without interacting with other nodes of the same layer. This means that, in order to harness the inherent parallelism offered by the HMLP or other neural networks with MLP-like structure, implementing the HMLP on a concurrent system such as the FPGA will be more favourable than running the HMLP on sequential system such as a PC, a general purpose microprocessor or a microcontroller.

3. DESIGN OF FPGA-BASED ARCHITECTURE OF HMLP

The architecture of the proposed HMLP for FPGA is planned using top-down approach and is heavily based on HMLP's structure in Figure 1 and HMLP's output (1). The development of the architecture for the HMLP is done via bottom-up approach, whereby the lowest-level modules are described first. Next higher-level modules are later described level-by-level, until finally the top-level module which encompassed all submodules that make up the HMLP is described.

The HMLP is designed to be modular such that if modification to a module (such as activation function module) is required, the modification can be performed on the target module while retaining all other modules of the whole structure. The HMLP architecture is designed with concurrency in mind to observe the inherent parallelism of a HMLP. The architecture is designed such that the hidden nodes will execute multiply-add operation in parallel with other hidden nodes. While hidden nodes are processing their outputs, the output nodes will perform multiply-add operation of the weighted input layer to output layer connection at the same time, as shown in Figure 2. This concurrent architecture reduces the overall processing time, but at the expense of increased logic element utilization.

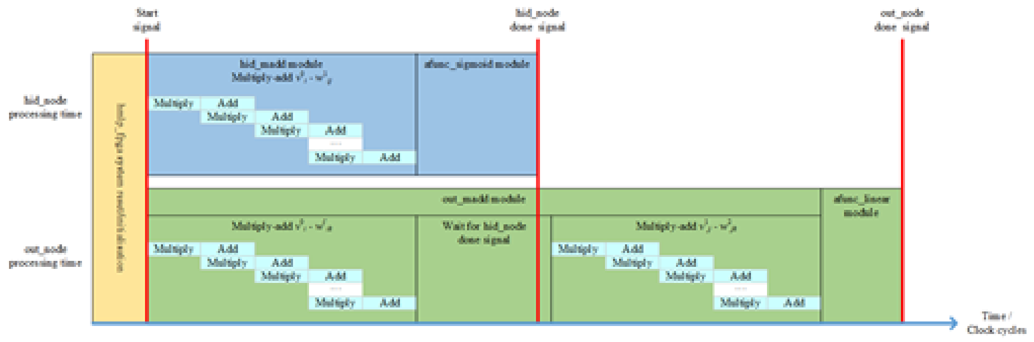

Figure 2. Timeline of hmlp_fpga execution showing the start signal, hid_node done signal, out_node done signal and relative processing time of hid_node and out_node

3.1. Modules and Sub-Modules

Figure 3 shows the modules within the top-level module hmlp_fpga that make up the HMLP architecture for FPGA implementation and all major internal connections. The modules that make up hid_node and out_node are shown in Figure 4.

Based on HMLP's structure, the hidden and output nodes, represented by the modules hid_node and out_node respectively, execute most arithmetic operations of the HMLP. Both hid_node and out_node modules comprise of a multiply-add module and activation function module.

All nodes of a given layer are connected to adjacent layers via weighted connections. These connections are described in higher level modules, which mainly operate as to route the input signals, weights, biases and other intermediate signals to the correct hidden or output node.

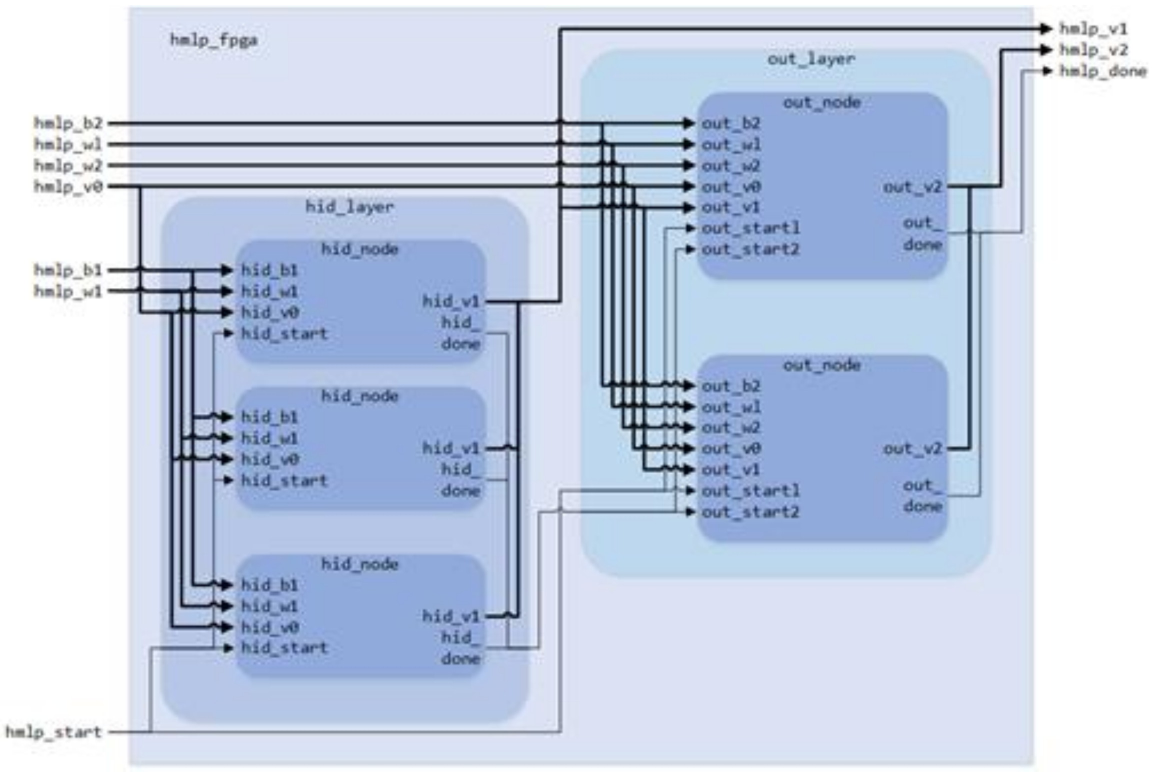

Figure 3. Modules within top level module hmlp_fpga

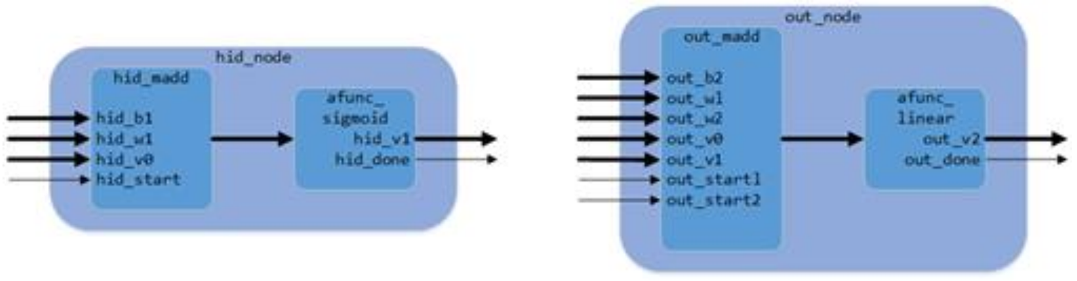

Figure 4. Lower-level modules within hid_node and out_node

Low-level modules hid_madd and out_madd are described to perform multiplication of the input signals to its respective weights and then to sum the input-weight products at the HMLP's hidden and output nodes, respectively. These modules will multiply its input-weight pair one-by-one and accumulate all product of the multiplication. Module out_madd perform almost similar operation to hid_madd but with the extra processing of weighted connection from input layer to output layer of the HMLP. Floating-point multiplication IP and addition IP are used to implement the multiplication and summation operations. A state machine controls the overall operation of the hid_madd and out_madd modules. The accumulated value is then passed to next module that applies the activation function to the accumulated value.

The summed values from hid_madd and out_madd are passed to next modules, afunc_sigmoid and afunc_linear. Both are the modules describing the operation of the sigmoidal activation function at the hidden nodes and the linear activation function at the output nodes respectively. These activation function modules are described as separate module from the multiply-add modules because if different activation function is required, the new activation function can be swapped in while all other modules can be retained. In afunc_sigmoid module, the sigmoidal activation function is implemented using floating-point IPs for exponent, addition and inversion operations with a state machine to control the flow of data through the afunc_sigmoid module and generate necessary control signals. In afunc_linear module, no arithmetic operation is performed, only a state machine is described to control the flow of data through the afunc_linear module and generate necessary control signals. The activation function's output from afunc_sigmoid or afunc_linear module will be the output of individual hid_node or out_node module.

Next higher-level module described the HMLP's hidden node and output node. Module hid_node encapsulates hid_madd and afunc_sigmoid modules that perform the operation of the hidden nodes. Likewise, out_node module encapsulates out_madd and afunc_linear that performs the operation of output nodes of the HMLP.

The hid_layer and out_layer modules repetitively generate the hid_node module and out_node module that formed the HMLP's hidden layer and output layer, respectively. Top level module hmlp_fpga connects all modules, input signals, output signals and intermediate signals to implement the HMLP structure as in Figure 1 on FPGA.

3.2. Floating Point Number Representation

Real numbers representation is crucial and on an FPGA, all real numbers need to be described in binary. In this architecture, floating point number representation is utilised to reduce development time instead of describing a unique number representation. Using standardised number representation, such as the IEEE754 standard [10] that define the bit ordering of 32-bit single precision floating-point numbers, allow for better integration with other devices, avoid potential confusion when communicating with different systems, and makes custom designs more adaptable for future developments.

Arithmetic operations on the floating-point numbers is done through the use of Floating-Point Arithmetic IPs from Altera (now Intel FPGA) that meets most of the IEEE754 standard [11]. These floating-point arithmetic IPs are used in the hid_madd, out_madd and afunc_sigmoid modules to perform operations of multiplication, addition, exponent, and inversion.

4. SIMULATION

The architecture of the HMLP is described in VHDL and compiled on Altera Quartus II version 15.0 Web Edition. The architecture is then simulated using ModelSim 10.3d software. A VHDL testbench tb_hmlp_fpga is described to provide the necessary stimuli to the HMLP's top level module hmlp_fpga. For the simulation, inputs to hmlp_fpga are set by tb_hmlp_fpga and the outputs of hmlp_fpga are observed from the simulation waveform. The inputs and weights data for hmlp_fpga's simulation is generated using MATLAB. Figure 5 shows the ModelSim simulation output of hmlp_fpga for n_i = 3, n_h = 3, and n_o = 2. The start signal, hid_layer done and final done signal is pointed out by its respective cursor.

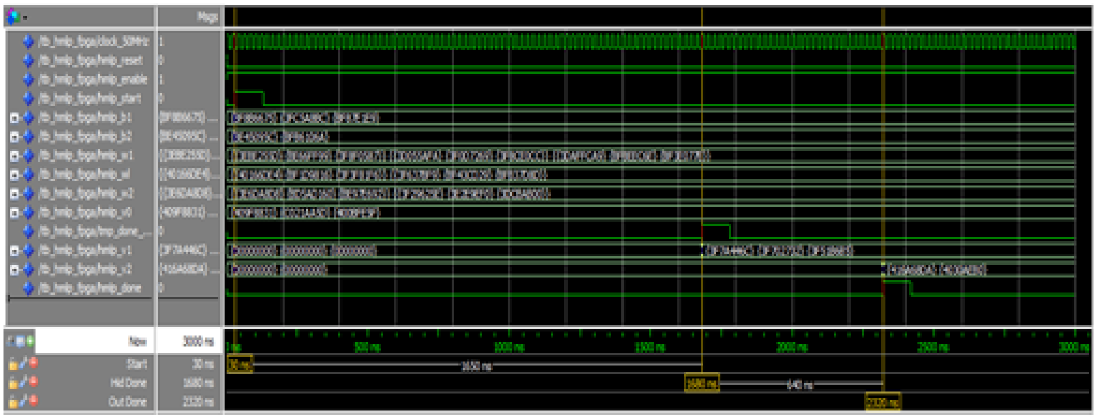

Figure 5. Simulation waveform of the designed HMLP architecture

The simulation is repeated three times with different n_i, n_h and n_o combinations. For each n_i, n_h and n_o combination, five different input datasets are randomly generated using MATLAB and coded into the simulation testbench tb_hmlp_fpga. Simulation output is compared to the result computed by MATLAB. The simulation shows that hmlp_fpga produced similar outputs to MATLAB results. Some deviation between simulation output and MATLAB result do exist, but the difference is very small and can be considered to be negligible.

5. IMPLEMENTATION ON FPGA

After the simulation stage, the designed HMLP architecture is then compiled for the target FPGA development board. All compilation, netlist generation, place and route, and generation of configuration bitstream are done using Quartus II. For the purpose of testing the HMLP architecture on the FPGA, a synthesisable testbench is described to assign the inputs and weights data to the HMLP and display the output data in human readable form.

5.1. FPGA Device/Development Doard

The target FPGA development board for testing the HMLP architecture is the DE1-SoC from Terasic. The DE1-SoC has an Altera 5CSEMA5F31C6 FPGA chip as the main FPGA device and other necessary circuitries for the Cyclone V FPGA-SoC to function. DE1-SoC is also equipped with various input/output components and devices for users to explore the FPGA-SoC [12].

The on-board slide switches, push-button switches, LEDs, 7-segment displays and clock generator are used to test the designed HMLP architecture. The synthesisable testbench takes user inputs from the switches and pushbuttons, assigns the HMLP’s test inputs and weights to the hmlp_fpga module, and lastly displays the HMLP outputs from hmlp_fpga module accordingly on the 7-segment displays and LEDs.

5.2. Compilation Report

A full compilation report of the hmlp_fpga module and the synthesisable testbench for the Cyclone V FPGA is shown by Altera Quartus II after successful compilation. Useful details such as the target FPGA’s resource utilisation can be extracted from the compilation report. Overall, resource utilisations of the designed HMLP architecture including the synthesisable testbench are shown in Table 1.

Table 1. CycloneV SE Resource Utilisation taken from Quartus II Compilation Report

HMLP structure	ALM	Total register	Total block memory bits	Total DSP block	HMLP Compute time
$n_i = 3$ $n_h = 3$ $n_o = 2$	5534 / 32070 (17%)	8576	2964 / 4065280 (0%)	53 / 87 (61%)	115 clock cycles (2.3 µs on 50 MHz clock)
$n_i = 8$ $n_h = 3$ $n_o = 2$	5661 / 32070 (17%)	8683	2964 / 4065280 (0%)	53 / 87 (61%)	150 clock cycles (3.0 µs on 50 MHz clock)
$n_i = 3$ $n_h = 4$ $n_o = 2$	7137 / 32070 (22%)	11053	3928 / 4065280 (0%)	70 / 87 (80%)	157 clock cycles (3.14 µs on 50 MHz clock)
$n_i = 8$ $n_h = 3$ $n_o = 3$	5588 / 32070 (17%)	8585	2964 / 4065280 (0%)	53 / 87 (61%)	150 clock cycles (3.0 µs on 50 MHz clock)

For a HMLP with greater n_i, n_h or n_o, it’s FPGA resource utilisation will be greater. Greater n_i will add to compute time but has minimal impact on resource utilisation; increasing n_h will add compute time and uses more resource; whereas increasing n_o has no impact on compute time but increases resource usage.

The ALM stand for Adaptive Logic Module in Altera FPGAs [13]. In this design, most of the FPGA resources is used up by the Altera’s Floating-Point IPs, especially the IPs for exponent and inversion operation to compute the afunc_sigmoid sigmoid activation function module at the hidden nodes. This is proven by the significant increment of resource utilisation when n_h is increased from 3 to 4 as in Table 1.

5.2. FPGA Implementation Outcome

From the simulation and FPGA implementation outcomes, the time needed for the HMLP architecture to compute its output value, in term of number of clock cycles, is determined by (3).

$$\text{hmlp\_fpga output latency} = 4 + 69 + 7n_i + 7n_h \quad (4)$$

Based on (3), the afunc_linear module takes 4 clock cycles to produce stable result at its output port. 69 clock cycles are the duration needed by afunc_sigmoid module to finish its operation and produce stable result on its output port. $7n_i$ and $7n_h$ are the output latency of the hid_madd and out_madd modules respectively. The output latency of these multiply-add modules varies depending on n_i and n_h because multiply-add is a sequential operation, thus increasing n_i or n_h will add to the overall output latency of the hmlp_fpga module.

For a HMLP with $n_i = 3$, $n_h = 3$, $n_o = 2$, and system clock of 50 MHz, the time needed for the HMLP compute its output is 2.3 μs (115 clock cycles).

The concurrent structure of the HMLP allow for all nodes to process its output concurrently, thus reducing the overall processing time. The processing time for multiply-accumulate operation performed by the hid_madd and out_madd modules is greatly affected by the number of input to the respective module. This is because multiply-add operation is a sequential operation. More inputs hid_madd and out_madd have, its processing time will be greater. To reduce the processing time of hid_madd and out_madd modules, the multiply operation and add operation is pipelined, such that these modules can calculate the product of next input-weight pair while adding and accumulating previous multiplication product as shown in Figure 2.

As in the simulation, the compilation and FPGA implementation of the designed HMLP architecture is repeated three times with different n_i, n_h and n_o combinations. For each n_i, n_h and n_o combination, five different input datasets are used. The datasets used for FPGA implementation are the same dataset used for simulation. The outcome of the designed HMLP architecture on FPGA is compared with the simulation result. The output value from the FPGA implementation is found to be the same as its simulation result.

6. CONCLUSION

This paper had described the development of architecture of HMLP neural network for implementation on Cyclone V FPGA. The HMLP architecture is designed to be concurrent and modular. The proposed architecture is compiled using Altera Quartus II and simulated on ModelSim. The simulation reveals that the proposed HMLP architecture is able to produce desired result. Next, HMLP is implemented on the Cyclone V FPGA device on board DE1-SoC development and education kit. The FPGA produced same outputs as compared to the simulation and MATLAB.

ACKNOWLEDGEMENTS

Appreciations are extended for the financial support from the Ministry of Education, Malaysia through the MyBrain15 scheme, and the tools, equipment and expertise at Universiti Malaysia Perlis.

REFERENCES

[1] X. Zhai, A. A. S. Ali, A. Amira, and F. Bensaali, “MLP Neural Network Based Gas Classification System on Zynq SoC,” *IEEE Access*, vol. 4, pp. 8138–8146, 2016.
[2] M. Wess, P. D. S. Manoj, and A. Jantsch, “Neural network based ECG anomaly detection on FPGA and trade-off analysis,” in *Proceedings - IEEE International Symposium on Circuits and Systems*, 2017.
[3] M. Y. Mashor, “Hybrid multilayered perceptron networks,” *Int. J. Syst. Sci.*, vol. 31, no. 6, pp. 771–785, 2000.
[4] J. Adnan, K. A. Ahmad, M. H. Mat, Z. I. Rizman, and S. Ahmad, “Cardiac abnormality prediction using HMLP network,” in *AIP Conference Proceedings*, 2018, vol. 1930.
[5] N. H. Harun, M. K. Osman, M. Y. Mashor, and R. Hassan, “Classification of acute luekemia using HMLP network trained by genetic algorithm,” *Adv. Sci. Lett.*, vol. 23, no. 4, pp. 2648–2652, 2017.
[6] Z. Saad, M. Y. Mashor, and W. Khairunizam, “Formation of momentum and learning rate profile for online training and testing of HMLP with ALRPE,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9490, pp. 268–275.
[7] S. M. Sharun, M. Y. Mashor, S. Yaacob, A. Zul Azfar, and N. Hamzah, “Development of Attitude Control Simulator for InnoSAT Satellite System,” in *The 2nd International Malaysia-Ireland Joint Symposium on Engineering, Science and Business 2012 (IMiEJS2012)*, 2012, pp. 225–234.
[8] B. Mohamed, A. Issam, A. Mohamed, and B. Abdellatif, “Comparison of hardware and NIOS II based software implementation of MLP on the FPGA plateform,” *J. Theor. Appl. Inf. Technol.*, vol. 72, no. 3, pp. 376–384, 2015.
[9] L. Y. Ann, P. Ehkan, and M. Y. Mashor, “Possibility of hybrid multilayered perceptron neural network realisation on FPGA and its challenges,” in *Lecture Notes in Electrical Engineering*, 2016, vol. 362, pp. 1051–1061.
[10] IEEE, “IEEE Standard for Floating-Point Arithmetic,” *IEEE Std 754-2008*, pp. 1–70, Aug. 2008.
[11] Altera, *Floating-Point IP Cores User Guide*. 2016.
[12] Terasic, *DE1-SoC User Manual*. Terasic Technologies, 2016.
[13] Intel FPGA, *Cyclone V Device Handbook Volume 1: Device Interfaces and Integration*. 2018.

BIOGRAPHIES OF AUTHORS

	Lee Yee Ann is a Ph.D candidate at School of Computer and Communication Engineering, Universiti Malaysia Perlis (UniMAP), Malaysia. He obtained Diploma in Computer Engineering (2010) and B.Eng in Computer Network Engineering (2013), both from UniMAP. His field of research includes Digital Design, FPGA, and Embedded System. He is a graduate member of Board of Engineers Malaysia (BEM).
	Phaklen Ehkan received the BEng Electrical-Electronic Engineering (UTM), MSc. IT (UUM) and Ph.D in Computer Engineering (UniMAP - University of Birmingham, UK). He worked as an Engineer/Sr. Engineer in MNC- Electronic Industries for six years before joined the University Malaysia Perlis as a lecturer in 2003. Currently, he is an Associate Professor attached to School of Computer and Communication Engineering, UniMAP. His research interests include Reconfigurable Computing and FPGA, Digital Design and Embedded System, Digital and Image Processing, System on Chip (SoC), Smart System and IoT. He has published over 80 articles in International Journals and Scopus indexed Proceedings. Dr. Phaklen Ehkan is currently a Chartered Engineer (UK), Professional Technologist (MBOT), graduate member of BEM, member of IEEE, BCS and IACSIT.
	Mohd Yusoff Mashor is a Professor at Universiti Malaysia Perlis in the School of Mechatronic Engineering. His main research fields are artificial intelligence, control systems and image processing. His researches are mostly applied to medical diagnostic systems and satellite control systems, where artificial intelligence are intensively utilised. He obtained his bachelor degree (in 1990) from Westminster University, UK in Control and Computer Engineering, M.Sc (in 1991) from Sheffield University, UK in Control and Information Technology, and Ph.D (in 1995) from Sheffield University in Artificial Intelligence. He has served as a lecturer in Control Systems and Electronics for more than 20 years. He started his career as a lecturer at Universiti Sains Malaysia and currently serves Universiti Malaysia Perlis.
	Siti Maryam Sharun received B.Sc in Electrical & Electronics Engineering (1997) and M. Education (1999) from Universiti Teknologi Malaysia, and Ph.D in Mechatronic Engineering (2013) from UniMAP. Her field of specialisation are Control System, Artificial Intelligence, and Computer Technology. Dr. Siti Maryam Sharun was a lecturer at Politeknik Shah Alam, a senior lecturer at Politeknik Sultan Abdul Halim Mua'dzam Shah, and is currently a Senior Lecturer at Faculty of Inovative Design & Technology, Universiti Sultan Zainal Abidin (UNISZA), Malaysia. She is a member of IAENG.