ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

Design and Implementation of Parallel MAC Unit for FPGA-Based Deep Learning Algorithms Using VHDL

A.Naganjanamma[1], G.Lakshmi Bharath[2]

[1]PG Scholar, Dept.of ECE, Srinivasa Institute of Technology & Science, Kadapa.
[2]Assistant Professor, Dept.of ECE, Srinivasa Institute of Technology & Science, Kadapa.
Email: anjupmvu2695@gmail.com, laksmibharath.414@gmail.com

Abstract –

Deep neural network algorithms have proven their enormous capabilities in wide range of artificial intelligence applications, especially in Printed/Handwritten text recognition, Multimedia processing, Robotics and many other high-end technological trends. The most challenging aspect nowadays is to overcome the extremely computational processing demands in applying such algorithms, especially in real-time systems. Recently, the Field Programmable Gate Array (FPGA) has been considered as one of the optimum hardware accelerator plat form for accelerating the deep neural network architectures due to its large adaptability and the high degree of parallelism it offers. In this paper, the proposed 8-bits fixed-point parallel multiply accumulate (MAC) unit architecture aimed to create a fully-customize MAC unit for the Convolutional Neural Networks (CNN) instead of depending on the conventional DSP blocks and embedded memories units on the FPGAs architecture silicon fabrics. The proposed 8-bits fixed-point parallel multiply-accumulate (MAC) unit architecture is designed using VHDL language and can performs a computational speed up to 4.17 Giga Operation per Second (GOPS) using high-density FPGAs.

Keywords – MAC, GOPS, CNN, FPGA.

1. INTRODUCTION

The basic MAC unit comprises Multiplier, Adder and Accumulator. This unit computes the running sum of products, which is at the heart of algorithms such as the FIR and FFT. The MAC unit can be alienated into two main blocks, the multiplier and accumulator. The multiplier also further divided into several blocks such as partial product generator, reduction of partial products, etc. The partial addition block comprises of summation tree and final adder. The summation block is key part of the MAC unit; this block consumes most of the circuit power and delay and occupies most of the area as well. Several MAC unit models can be obtained by replacing the multiplier unit with various architectures. The simple MAC unit block diagram is shown in Fig.1.

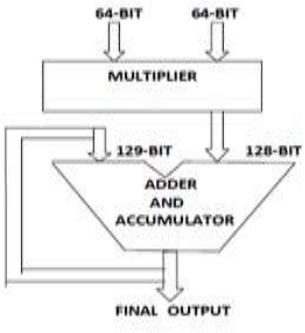

Fig.1.MAC Unit Block Diagram

Research and development in the field of digital electronics has revolutionized human life. Because of developments in IC technology isolated circuits are almost obsolete. The need
of scaling down of chip area is increasing very rapidly. It seems that high speed; less complexity; high density; low power consumption; price etc. will be the major factor of apprehension in the ultramodern device designing. These motivate us to do research on most basic and essential building block of processor. Since MAC unit computes the running sum of products which is the most widely executed operation. Therefore, we have chosen the MAC unit for optimization. In order to design an ALU for high performance applications like Digital Signal Processor (DSP), it is needed to incorporate Multiply-Accumulate (MAC) unit. Digital signal processors have diverse architectures and improved features than general purpose processors. This makes multiplier and-accumulator (MAC) an essential element of the digital signal processing such as filtering, convolution, and inner products. Most digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transform (DWT). Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic determines the execution speed and performance of the entire calculation. The capability to calculate with a fast MAC unit is essential to attain high performance in many

DSP algorithms.MAC unit are also indispensible element of digital filters.

2. 8-BIT FIXED POINT MAC UNIT

The convolutional layer is considered as the most demanding layer comparing to the other layers in any convolutional neural network architectures for its massive computational processing requirements due to the enormous number of multiplier and addition blocks needed to achieve the dot product functionality. Conventionally, the dot product functionality in the convolutional layer was attained using sequential MAC unit design such used in CPU or GPU hardware-based systems or by non-optimized parallel MAC unit in FPGA hardware-based systems due to the dependency of the hard DSP units in its silicon fabric. In general, the FPGA hardware-based systems are the optimum solutions from the computational speed point of view due to the design architecture flexibility and the parallelism capability offered by such platforms. The proposed parallel MAC unit is designed to boost the computational processing speed performance of the convolutional neural network by depending on a parallel and full-custom MAC unit to perform the two vectors dot product required in the CNN architectures, based on VHDL language, while keeping the independent to the FPGA silicon fabric as one of its main goals to overcome the issue raises when transferring the design from FPGA family to another due to the parameter variations of the generic DSP block in each FPGA family. The dot product of two vectors can be simply represented as the summation of the element-wise multiplication of these two vectors. So, if the first vector is x= [x0, x1, x2, ……., xn-1], and the second vector is y= [y0, y1, y2, ……., yn-1], and both the two vectors have the same vector length n, hence, the result of the dot product operation on these two vectors can be given as:

$$x.\,y = x^T\,y$$

The dot product is considered as the dominant operation in designing a deep neural network. To speed up the convolutional layer computational time using FPGA, a summation of multi-parallel dot product operation between a group of the input feature map element values and the filter (weight) coefficients is taken place to produce the proposed parallel MAC unit. In this paper, and due to the comparison purposes, a full custom MAC unit with 4 input feature map elements, has been designed. The proposed design has mainly 3 stages, as shown in Figure. In the first stage, four 8-bits elements from the input feature map are extracted and to be
multiplied with the corresponding four different 9bits weight coefficients, extracted from the weight bank, concurrently. The four results from the multiplier units from the first stage will be then added simultaneously. Finally, the result from the addition stage is being accumulated. The output of the proposed MAC unit operation can be described mathematically as indicated, where x is indicating the feature map element values, and y indicating the filter (weights) coefficient values.

$$x.\,y = \sum_{i=0}^{i=3} x_i\,y_i$$

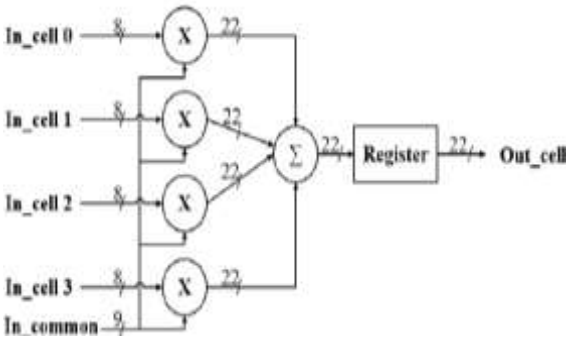

Fig 2. The Proposed 8-bits fixed-point MAC unit

The proposed architecture accepts four pair of operands which are multiplied and further added. Each operand is either considered as signed or unsigned numbers. Figure shows the 8-bit concurrent mac architecture. Here the multiplier used is booths multiplier to multiply two operands and the adders are carry look ahead adder and carry save adder used to add the multipliers output. In the architecture, the operands given are a1, b1, a2, b2, a3, b3, a4, b4 these are 8-bit inputs.

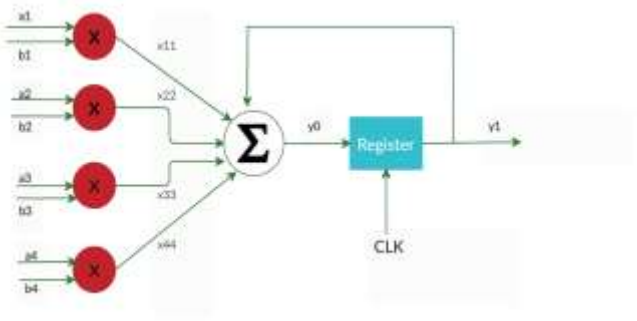

Fig 3. Concurrent MAC unit

The inputs a1, b1 are given to mul 1 and the output is taken as x11[15:0], the inputs a2, b2are given to mul 2 and the output is taken as x22[15:0], the inputs a3, b3 are given to mul3 and the output is taken as x33[15:0], the inputs a4, b4 are given to mul 4 and the output is taken as x44[15:0]. The output of mul 1, mul 2, mul 3 (x11, x22, x33) are given to carry save adder1 (CSA1) and the outputs are s1[16:0], c1[16:0]. The output of mul 4(x44) is added with 0 bit to become 18 bit (x5). x5and CSA1 adder outputs(s1,c1) are given to carry save adder 2(CSA2) and the outputs are s2[17:0], c2[17:0].s2, c2 are given to carry look ahead adder (CLA) and the output is Y0[18:0].The output of CLA (y0) along with clock and reset is given to the register where the result gets stored as yi[18:0].For one clock cycle 4 MAC operations are computed.

Four main considerations have been taken while designing the proposed MAC unit to achieve the optimum computational performance. 1. The first consideration is to synthesis the proposed design using different Intel Altera

FPGA families to recognize the effects that will occur to the overall system performance due to the change of the FPGA logic element (LE) or adaptive logic module (ALM) in the silicon fabric architecture, and the results showed that the proposed MAC design is consuming less than 1% in the three different FPGA families. 2. The second consideration is related to minimizing the combinational and sequential logic block resources of the proposed MAC design while not to depend on using any of the dedicated embedded multiplier elements nor the embedded DSP blocks to sustain the unconditional independence of such embedded blocks when switching between the different FPGA families. 3. The third and the fourth consideration are related to maximize the computational speed performance while achieving the minimum core dynamic thermal power dissipation. The maximum core speed performance of the proposed MAC unit is varied corresponding to the applied silicon fabric process technology; hence the maximum core frequency can be achieved by depending on the Aria 10 FPGA family which
fabricated on 20 nm processing technology.

TABLE 1.: THE PROPOSED 8-BITS FIXED-POINT MAC UNIT FLOW SAMURRY REPORT COMPARISON

	INTEL ALTERA FPGA Families		
	Cyclone IV E	*Aria 10*	*Stratix V*
Device	EP4CE115F29C7	10AX115R4F 40E3SG	5SGXEABN3F4 513YY
Total logic elements	1076 / 114,480 LE (< 1 %)	370 / 427,200 ALM (< 1 %)	356 / 359,200 ALM (< 1 %)
Total combinational functions	1038	710	711
Dedicated logic registers	168	174	174
Total memory bits	0 / 3,981,312 (0 %)	0 / 55,562,240 (0 %)	0 / 54,067,200 (0 %)
Embedded multiplier 9- bit elements or DSP blocks	0 / 532 (0 %)	0 / 1,518 (0 %)	0 / 352 (0 %)

3. RESULTS

The proposed 8-bit fixed point parallel multiply- accumulate unit is coded in VHDL programming language. The design is verified using a behavioural simulation in Xillinx ISE simulator tool and the design is synthesized using XST tool.

Fig.4.simulation results of serial MAC

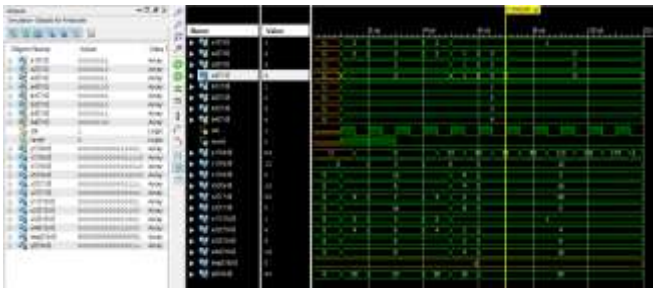

Fig.5.simulation results of concurrent MAC

The simulation result of proposed concurrent multiply-accumulate unit is shown in the fig 5. The proposed design is synthesized using Xillinx XST tool. The design is validated by choosing the Xillinx Virtex FPGA family (XC6VLX550T_2FF1759) the synthesis device utilization summary is as shown in the fig 6.

Fig.6. Device utilization summary

RTL Schematic of the proposed design is as shown in fig (7, 8) which is having eight 8 bit input operands and 18 bit output bit.

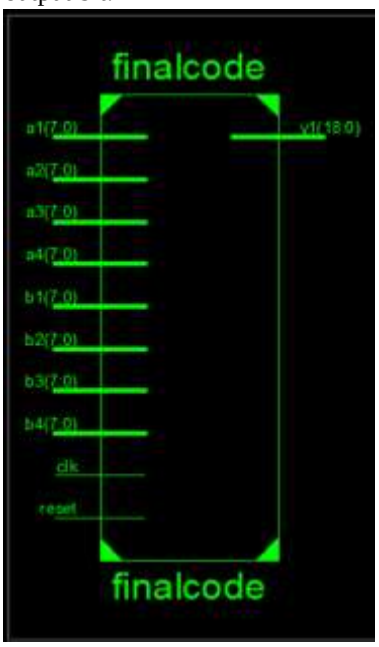

Fig.7. RTL schematic 1

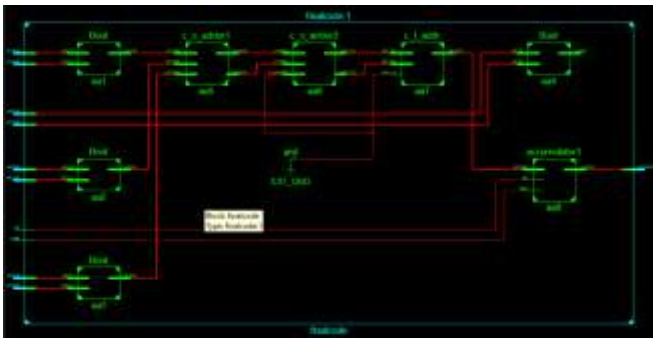

Fig.8. RTL schematic 2

The proposed design is mapped into Xillinx FPGA and technology schematic diagram is as shown in Fig.9.

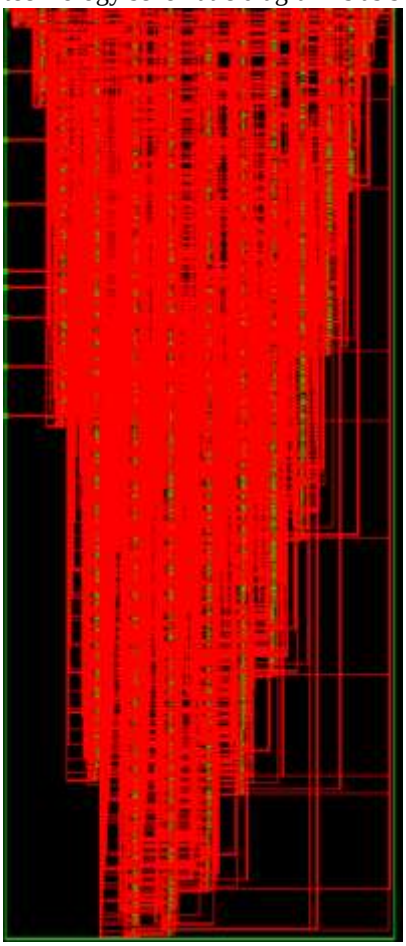

Fig.9. Technology schematic

Fig.10. Power consumption

CONCLUSION

The massive number of MAC units needed to create convolution neural networks are rapidly increasing and figuring a suitable solution for designing an optimized MAC unit is so essentials requirements. As proposed in this, the parallel MAC unit is giving a flexible and optimized solution to be used on any FPGA platform architecture regardless to the manufacturers with minimum logic resource requirements. Also, this proposal can be considered as a start point of a new generation of efficient MAC unit for CNN architectures which capable for replacing the conventional DSP block on the FPGAs to fulfill the requirements of such networks. The proposed MAC unit can achieve a computational speed performance using a concurrent architecture up to 4.17 GOPS with a maximum operation frequency obtained equal to 834 MHz using high-density FPGAs.

REFERENCES

[1] S. Moini, B. Alizadeh, M. Emad, and R. Ebrahimpour, "A Resource-Limited Hardware Accelerator for Convolutional Neural Networks in Embedded Vision Applications," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 64, no. 10, pp. 1217 - 1221, 04 April 2017 Oct. 2017.

[2] H. Wang, M. Shao, Y. Liu, and W. Zhao, "Enhanced Efficiency 3D Convolution Based on Optimal FPGA Accelerator," IEEE Access vol. 5, pp. 6909 - 6916, 28 April 2017 2017.

[3] K. Guo et al., "Angel-Eye: A Complete Design Flow for Mapping CNN Onto Embedded FPGA," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 35 - 47, 17 May 2017.

[4] G. T. Griffin Lacey, Shawki Areibi, "Deep Learning on FPGAs: Past, Present, and Future," 13 Feb 2016.

[5] T. Dettmers, "8-BIT APPROXIMATIONS FOR PARALLELISM IN DEEP LEARNING," presented at the ICLR 2016, San Juan, Puerto Rico, May 2 - 4, 2016.

[6] E. W. Yao Fu, Ashish Sirasao, Sedny Attia, Kamran Khan, and Ralph Wittig, "Deep Learning with INT8 Optimization on Xilinx Devices," UltraScale and UltraScale+ FPGAs, vol. v1.0.1, no. WP486, April 24, 2017.

[7] P. Gysel, M. Motamedi, and S. Ghiasi, "HARDWARE-ORIENTED APPROXIMATION OF CONVOLUTIONAL NEURAL NETWORKS," presented at the ICLR, San Juan, Puerto Rico 2016.

[8] A. Corporation, "Enabling High-Performance DSP Applications with Arria V or Cyclone V VariablePrecision DSP Blocks," White Paper, vol. WP01159-1.0, May 2011.

[9] Xilinx, "UltraScale Architecture and Product Data Sheet: Overview," vol. DS890 (v3.1), November 15, 2017.

[10] Xilinx, "UltraScale Architecture DSP Slice User Guide," vol. v1.5, no. UG579 October 18, 2017.

[11] R. P. D. M´ario V´estias, Jos´e T. de Sousa, Hor´acio Neto, "Parallel Dot Products for Deep Learning on FPGA," presented at the Field Programmable Logic and Applications (FPL), 2017 27th International Conference, Ghent, Belgium, Sept. 2017.