Turkish Journal of Computer and Mathematics Education V ol.12 No.12 (2021), 2315-2320
Research Article

Design and Analysis of systolic Array for Convolution Neural Networks

Nayana D K[1], Harshitha B N[2]

[1]Associate Professor, School of Electronics and Communication Engineering, REVA University, Bengaluru, India
[2]PG Scholar, VLSI and Embedded system, School of Electronics and Communication Engineering, REVA University, Bengaluru, India
[1]nayanajournal@gmail.com, [2]b.naidu.harshitha@gmail.com

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

Abstract—Convolutional neural networks (CNN) are used in many functions mainly Digital signal processing (DSPs), Networking, Image and video classification for its accuracy. CNNs need huge estimation and support. To fix hardware, Convolutional neural network accelerators required to be enhanced to bring down size of the memory, resource utilization and power usage. In this paper, the systolic array with Multiply-Accumulate unit (MAC) is designed. The input image of 128x128 size is considered and processed through main memory to systolic array. The systolic array performs the MAC unit operations rows and columns wise. The work is carried out using Xilinx ISE 14.7 environment and synthesised on Artix-7 FPGA.

Keywords—Accelerator, Convolutional neural network, Systolic array

1. Introduction

In present applications such as image, video, and other pattern recognition, convolutional neural network is frequently utilized as a huge enrolling architecture. A representative CNN device contains multi layered neurons which connects to send the data, which is inspired by optic nerve behaviour for accuracy. The rapid increase of new applications based on huge learning leads to the vast request for CNN on new devices, particularly the predicting part as further propagation, which is increasing and developing on small smart devices. It is hard to execute CNN of energy efficient and flexible on processing units like CPU or GPU because of variation and specific computation design. Unlike accelerators, FPGA using ASIC patterns are being regarded by more and more researchers.

The systolic array is considered because it is simple to build and gives an exceedingly systematic performance. It allows huge quantity of flexibility because it can accelerate any architecture that make use of compact matrix multiplication. The systolic array is a comparable architectural structure, which consists number of processing elements (major functions of PE includes floating point, multiplication and addition) with two dimensional matrix structure initiating a two dimensional pipeline. Every processing element is responsible for MAC unit (multiplication and addition) which makes faster matrix multiplication. The arrangement of systolic array is constant and gives better flexibility. The systolic array can achieve high resource utilization under high clock frequency.

2. Related work

Feng et al. [3] present Energy-Efficient and High-Throughput FPGA-based Accelerator for Convolutional Neural Networks. Accelerators using floating-point and fixed-point processing is done and runs at 166 MHz.

Xie et al. [4] describes the High Throughput CNN Accelerator Design Based on FPGA. The architecture for CNN accelerator to improve the utilization of on-chip resources and keep the DSP efficiency at a high level for most of the convolutional layers.

Cao et al. [5] present FPGA-based accelerator for convolution operations using systolic array architecture on Xilinx Zed Board device and achieving performance density with respect to DSP elements.

Tsai et al. [6] elaborates the FPGA Based accelerator for Deep Neural Networks (DNN), which offers low latency and lower chip area utilization. The Complex connection relationship and memory usage scheduling is considered in DNN to improve the design parameters.

Kyriakos et al. [7] present the High Performance Accelerator for CNN Applications. The CNN model is trained for the MNIST dataset and the VHDL design targets high throughput, low power while using only on

chip memory. The architecture uses parallel computations at the convolutional and fully connected layers and it has a highly pipelined output layer.

Lian et al. [8] explains the High-Performance FPGA-Based CNN Accelerator with Block-Floating-Point Arithmetic. An optimized block floating- point (BFP) arithmetic is adopted in our accelerator for Efficient inference of deep neural networks. The 8-bit BFP arithmetic with optimized rounding and shifting operation-based quantization schemes improves the energy and hardware efficiency by three times.

Cho et al. [9] describes the Implementation of Data-optimized FPGA-based Accelerator for Convolutional Neural Network, which focuses on reducing latency and memory usage, and the performance is compared with a conventional floating point design. The accelerator can achieve latency reduction up to 90% and memory usage reduction up to 40% without any accuracy loss, compared to the conventional design.

3. Methodology

The architecture of Hardware Accelerator for CNN is shown in figure 1. The proposed accelerator is a huge matrix multiplication unit (MMU) that performs 64x64 8 bit integer multiply and 32 bit integer accumulate per cycle. The proposed Hardware Accelerator uses a systolic array, which is a similar matrix of processing elements (PEs), and each element connected only to its neighbours. While executing, every processing element can alternatively read from its neighbours, determine a simple function, and save the result in its local memory.

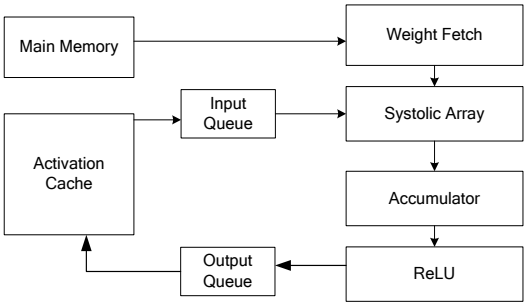

Fig 1: Block Diagram of Hardware Accelerator for CNN

The pipelining conceptions are used to expand he throughput of the design. The proposed Hardware Accelerator for CNN is having three main pipeline stages:

- Weight / Input loading
- Matrix multiplication
- Activation and store results to output queue.

Systolic Array Pattern: Every processing element has an accumulator and a multiplier, and it can determine a single MAC operation per cycle. Before executing weights must be loaded locally. In design, FIFO are used to send the data into the systolic array.

Weight Fetching**:** Before multiplication takes place, the weights must be prestored into the array. So load the weights similarly to data loading, using a queue at the array edge and every processing element sends its present weight to its neighbour.

MAC and ReLU: Weights and Activations can be measured to use 8 bit arithmetic.

4. Implementation and results

A hardware accelerator is a specific hardware unit that carries out a set of functions with improved performance or well energy efficiency than a CPU. Example of common accelerators are GPUs, digital signal processors (DSPs).

A. Main Memory Module:

The main memory is designed using Intellectual property (IP) cores by creating ROM module. Consider an input image with fixed size (128*128) and convert to text file format using MATLAB tool as shown in figure 2.

The text file contains image data in the form of pixels. The text file contains totally 16384 pixel information's. Again the text file is converted into the co-efficient (.coe) file format. Take a 14-bit counter

module for address to ROM module. Load the .coe file to the ROM module for memory initialization. The generated ROM module provides the final pixel information of input image.

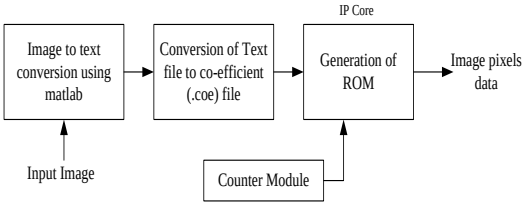

Fig 2: Main memory module

The RTL Schematic of Main memory module is shown in Figure 3. It mainly contains Counter and ROM module (IMAGE)

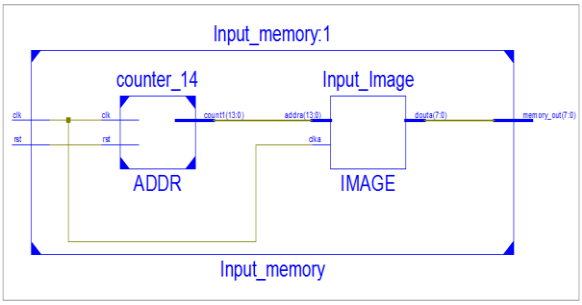

Fig 3: RTL Schematic of Main memory module

The Main memory utilizes 14 Slice registers, 13 Slice LUT's and 4 Block RAM's /FIFO units as shown in table1.

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	14	126800	0%
Number of Slice LUTs	13	63400	0%
Number of fully used LUT-FF pairs	0	27	0%
Number of bonded IOBs	10	210	4%
Number of Block RAM/FIFO	4	135	2%
Number of BUFG/BUFGCTRLs	1	32	3%

Table 1: Resource utilization of Main memory module

The Simulation results of main Memory module is shown in Figure 4. Once clock (clk) is activated with active low reset (rst), the memory operation starts. The address (addr) is issued by counter module to ROM Module. The 8-bit ROM output (memory_out) values are generated and it is matched with input file text is shown in Figure 5.

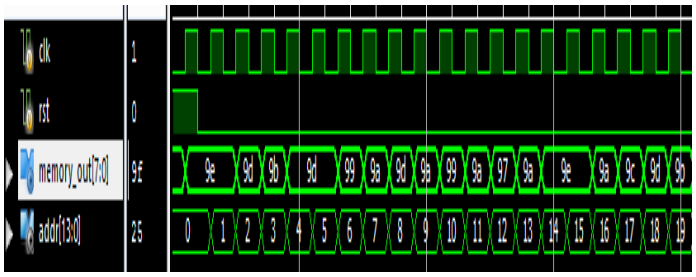

Fig 4: Simulation Results of Main memory module

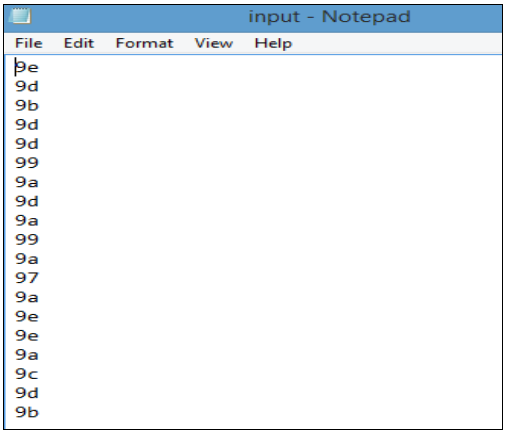

Fig 5: Input text file pixel values for simulation validation

B. **Systolic array using MAC unit:**

In Hardware accelerator, each PE has a multiplier and an accumulator, and that can determine a single MAC operation per cycle. Before execution begins, the weights must be loaded locally. For an N X N matrix multiplication, to generate the output wave it uses 2n-1 clock cycles and finish the multiplication.

It use a 128 x 128 8 bit Static RAM First in First out (FIFO) (16kB) to send the data into the systolic array. Without filling the data to synchronize data loading with the systolic array multiplier, it uses a real bit on each row to coincide the data. The input and output uses nearly 16KB of memory.

The overall Systolic array design is shown in Figure 6. It contains number of PE's connected in array format. Each PE acts a MAC unit. The Weight inputs are multiplied with data input values and multiplied results are added with accumulator input values to generate the MAC output. While executing, from left side the data flows and that is multiplied by prestored weights.

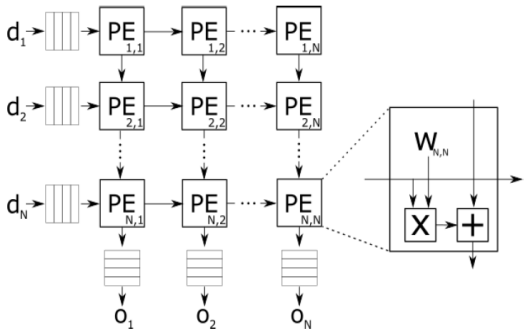

Fig 6: systolic array design.

The MAC unit top module is shown in Figure 7 and its simulation are shown in Figure 8. The MAC unit mainly contains clock (clk), reset (rst), enable (en), weight set signal (wg_set), 8-bit weight input (w_in), 8-bit data input (din), 16-bit accumulator input (acc_in) at input side. The MAC unit generates 16-bit accumulator (acc_out) signal at output side.

Once clock (clk) is activated with active low reset (rst), the MAC operation starts. Define the input signals and generates the 16-bit MAC unit output accordingly.

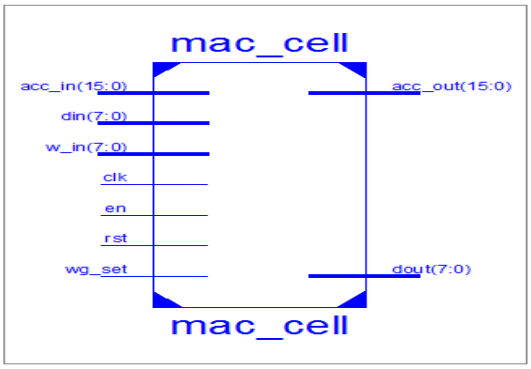

Fig 7: Top Module of MAC module

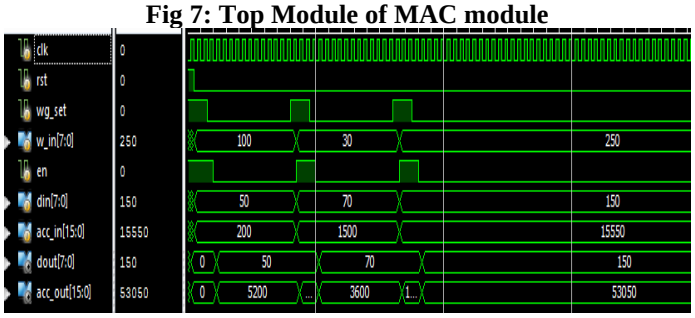

Fig 8: Simulation Results of MAC module

5. Conclusions

In this paper, the systolic array with MAC unit is designed and synthesized on Artix-7 FPGA. The input image of 128x128 size is processed in systolic array module. The complete systolic array with MAC unit is designed using HDL and Simulated on Modelsim environment. In future, the hardware accelerator module is designed using the systolic array with MAC unit for CNN applications.

6. Acknowledgment

I would like to thank my guide Nayana D.K., Associate Professor, School of ECE, REVA University for her advice and guidance. I would like to express my gratitude to Prashanth V. Joshi, Associate Professor and PG Co-ordinator, School of ECE, REVA University for his support. I am happy to extend my thanks to the Director of School of ECE R.C. Biradar and Assistant director K.M. Sudharshan.

References

1. C. Farabet, C. Poulet, J. Y. Han and Y. LeCun, "CNP: an FPGA-based processor for Convolutional Networks," 19th International Conference on Field Programmable Logic and Applications, pp. 32-37, August 2009.
2. J. Qiu, et al., “Going deeper with embedded FPGA platform for Convolutional Neural Network,” in Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, pp. 26-35, Feburary 2016.
3. Feng, Gan, Zuyi Hu, Song Chen, and Feng Wu. "Energy-efficient and high-throughput FPGA-based accelerator for Convolutional Neural Networks." In 2016 13th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), pp. 624-626. IEEE, 2016.
4. Xie, Liang, Xitian Fan, Wei Cao, and Lingli Wang. "High Throughput CNN Accelerator Design Based on FPGA." In 2018 International Conference on Field-Programmable Technology (FPT), pp. 274-277. IEEE, 2018.
5. Cao, Yunfei, Xin Wei, Tingting Qiao, and He Chen. "FPGA-based accelerator for convolution operations." In 2019 IEEE International Conference on Signal, Information and Data Processing (ICSIDP), pp. 1-5. IEEE, 2019.
6. Tsai, Tsung-Han, Yuan-Chen Ho, and Ming-Hwa Sheu. "Implementation of FPGA-based accelerator for deep neural networks." In 2019 IEEE 22nd International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), pp. 1-4. IEEE, 2019.
7. Kyriakos, Angelos, Vasileios Kitsakis, Alexandros Louropoulos, Elissaios-Alexios Papatheofanous, Ioannis Patronas, and Dionysios Reisis. "High performance accelerator for cnn applications." In 2019

29th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS), pp. 135-140. IEEE, 2019.

8. COMPREHENSIVE ANALYSIS OF BARRIERS IN IMPLEMENTATION OF LEAN MANUFACTURING IN INDIAN INDUSTRIES, Manojkumar J, Manivel Muralidaran V, Balaji M, International Journal Of Advance Research In Science And Engineering http://www.ijarse.com IJARSE, Volume No. 10, Issue No. 04, April 2021 ISSN-2319-8354(E).
9. Lian, Xiao Cong, Zhenyu Liu, Zhourui Song, Jiwu Dai, Wei Zhou, and Xiangyang Ji. "High-performance fpga-based cnn accelerator with block-floating-point arithmetic." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 27, no. 8 (2019): 1874-1885.
10. Cho, Mannhee, and Youngmin Kim. "Implementation of Data-optimized FPGA-based Accelerator for Convolutional Neural Network." In 2020 International Conference on Electronics, Information, and Communication (ICEIC), pp. 1-2. IEEE, 2020.