2023 8th International Conference on Computers and Devices for Communication (CODEC) | 979-8-3503-1717-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/CODEC60112.2023.10465738

VLSI Architecture of High Throughput Processing Unit for Efficiently Executing 2D Convolution for Deep Neural Network based Applications

Bhanuprakash Reddy Peddasubbanna Gari*, Suman Samui† and Soumen Garai‡
Department of Electronics and Communication Engineering
National Institute of Technology Durgapur, West Bengal 713209, India
Email:*pbhanupr@gmail.com, †samuisuman@gmail.com, ‡soumengoroi@gmail.com

***Abstract*—Convolution serves as a foundational component enabling Deep Neural Networks (DNN) to extract meaningful features from input data. This paper focuses on the real-time architecture of a high-throughput processing unit (PU) designed for 2D convolutions. Three distinct PU topologies are presented: (a) a non-pipelined Multiply-Accumulate (MAC) based PU, (b) a 2-stage pipelined MAC based PU, and (c) a 5-stage pipelined MAC-based PU, which significantly reduces convolution process delay. These proposed architectures are implemented on Field Programmable Gate Arrays (FPGAs) using the Xilinx Vivado tool. Resource utilization and latency are thoroughly assessed and compared across various PU topologies on Artix-7 and Zynq-7 FPGA chips. Additionally, throughput is estimated and compared for different kernel and image sizes, revealing a substantial average improvement of 96.36% achieved by the pipelined MAC with its five-stage design compared to the non-pipelined MAC-based processing unit.**

***Index Terms*—Deep Neural Network, 2D convolution, Multiply−accumulate (MAC) unit, Pipelining, Pipelined MAC, Field Programmable Gate Arrays (FPGAs).**

I. Introduction

In recent years, the utilization of deep learning and deep neural networks (DNNs) has rapidly increased across various domains, such as object detection, object recognition, semantic segmentation, medical imaging, and more. This surge is attributed to their ability to achieve greater accuracy in less time than human expertise. However, this superior performance comes at the cost of significant computational complexity, memory use, and power consumption due to the billions of parameters in the network. In that complex process, convolution is the most significant process, which is done several times to get accurate results. However, convolution alone requires significant memory use, power consumption, and complexity. Convolution is the process of producing an output vector/matrix using an input kernel/filter vector and input data vector. The current work presents the implementation of a high throughput process unit for 2D convolution in FPGA. We have studied how a pipeline technique [1] can be used to provide high throughput and maximize performance in the process unit of convolution by partitioning the critical path. Increasing the throughput utilizing pipelining in the convolution process unit (PU) differs this study from other former studies. The main contributions of this work are as follows:

- This paper explored various convolution techniques that are used for CNN accelerators. Based on our analysis, Multiply−accumulate (MAC) has been used for the design of convolution architecture. Thus, this paper proposes a multi-staged MAC that can be used to construct a faster processing unit for 2D convolution.
- Multiplier has a more significant delay than adder in MAC. To address this issue, we have proposed a multi-phase signed multiplier method, wherein the delay at each stage is significantly lower than that of the adder.
- Finally, the proposed MAC with other MACs is implemented in FPGA boards. Those MACs are compared in terms of latency and throughput parameters to distinguish which MAC is the best suited for the convolution layer.

979-8-3503-1414-4/23/$31.00 © 2023 IEEE

II. Background and related works

Convolution is the pivotal process in the field of Deep Neural Networks. It is used for many applications like reducing the image size, edge detection of the objects in an image, sharpening an image, blurring the image, etc., in many image processing applications of Deep Neural Networks. This image processing is the basic process for many DNN applications like object recognition, face recognition, medical imaging, etc., thus proving that 2D convolution is the basic process for many DNN networks.

2D convolution involves sliding a small matrix called filter (or kernel) over an input image, performing element-wise multiplication between the overlapping kernel and image regions, summing the results to get an output value, and moving the kernel across the image and repeating the process to get an output processed matrix. This process extracts features, filters, or detects patterns. Optional padding maintains input size, and the stride controls kernel movement. The result is an output feature map that summarizes the convolved features. By taking the consideration of stride and zero padding, the output size can be determined as shown below:

$$Output_size = \left(\frac{A + 2 * Z_P - f}{str}\right) + 1 \tag{1}$$

where A denotes the input image size, Z_P is the Zero padding, f is the filter/kernel size and str is stride. These types of similar operations can be seen in Dilated convolution, deconvolution [2] and fully connected Neural Network layers From [3], its concluded that FPGA has better memory usage and performance than GPU and CPU for 2D convolution. Thus, this paper selected the FPGA implementation of the proposed architecture, which consists of MAC, a best-suited process block for convolution. [4] MAC, called multiply-accumulate, is a block that does multiplication and accumulates the multiplied result. Thus, one MAC operation produces one output value. But producing every output value takes a finite number of clock cycles. We use the pipeline method to reduce the cycle required and the clock period to reduce this. We witnessed extensive research on the effective implementation of 2D convolution architecture. [2], [5], [6], [7] are the different types of implementations of this 2D convolution architecture. These have their structure to implement the convolution using Multipliers and adders. Compared to that, this paper only uses one multiplier and adder to achieve efficiently less area, power and high throughput.

Authorized licensed use limited to: Universiti Malaysia Perlis. Downloaded on September 09,2026 at 08:14:27 UTC from IEEE Xplore. Restrictions apply.

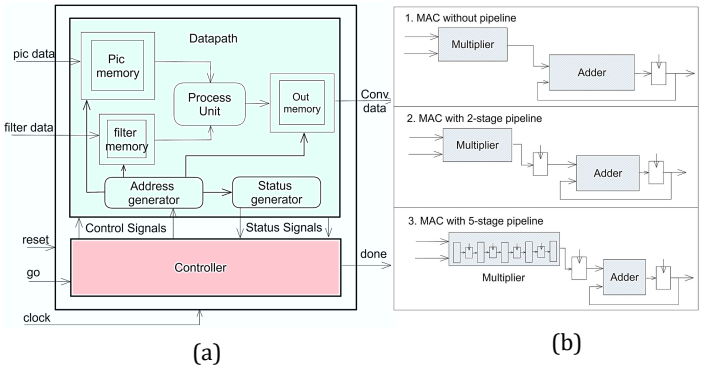


Fig. 1. (a) Proposed architecture for 2D convolution. (b) Topologies of MAC used in Process Unit(PU)

III. Proposed Architecture

The proposed architecture mainly consists of datapath and the controller. The datapath contains memory and processing units, as shown in the figure. The memory is designed to store picture, filter, and convoluted output data. The memories have row and column addresses, and the size is enough to do zero padding. The PU is designed only for the convolution process, and a controller is used to control this process. Here, we make use of three topologies of MAC for the processing unit for convolution, as shown below. The MAC is made up of a modified signed Dadda multiplier [8] and the Carry Select Adder (CSELA) with 4-bit Carry Look-ahead Adder. Dadda Multiplier is used because it can be pipelined with minimum no. of stages compared to other multipliers, and it is in 8x8 Dadda multiplier; it takes four independent process stages as in [8] which makes it 4-stage pipeline system. 4-bit CLA is the fastest 4-bit adder, and CSELA with 4-bit CLA is a faster adder. Address generator contains row, column address and stride generator used to access data from the memory or to the memory. A status is generated from the status generator by comparing row, column address and strides with its predefined limiters of address. Coming to the controller, it increments the row and column address and stride accordingly by considering stride and also activates the memories accordingly using memory select. The algorithm of the controller is given below.

IV. Experimental Results

All the proposed hardware structures are implemented using Verilog HDL (hand-coded) and with the memory sizes for pic, filter and output as 6×6, 3×3 and 4×4 with word size of 8 bits. The code verified and simulated in Xilinx Vivado Design suite. Then the synthesis of this architecture is done in the Artix7-AC701 Evaluation Platform (xc7a200tfbg676-2) and ZYNQ-7-ZC702 Evaluation Board (xc7z020clg484-1) and the results are shown Table-I and Table-II. Compared to the non-pipelined PU, the PU based two-stage pipelined MAC (MAC-I) had a 9.58% improvement in throughput whereas the proposed PU based on 5-stage pipelined MAC (MAC-II) has a 98.7% improvement for the Artix-7 board. For the Zynq-7 board, the improvement in throughput is 9.31% and 94.02% for PU based on MAC-I and MAC-II respectively.

V. Conclusions

This article presents a real-time implementation of a high throughput PU that performs 2D convolution on Artix-7 and Zynq-7 FPGA boards. The proposed convolution PU is mainly constructed by MAC, which uses a five-stage Dadda multiplier and CSELA adder components. It empowers the throughput by reducing the delay of the multiplier. The proposed 5-stage MAC-based PU performs significantly better compared to the PU based on non-pipelined MAC and the two-stage pipelined MAC.

TABLE I
Delay and utilization of each module used in the Process Unit

Module Name	Artix-7 (xc7a200tfbg676-2)		Zynq-7 (xc7z020clg484-1)	
	Delay (in ns)	Slices (LUT)	Delay (in ns)	Slices (LUT)
Adder	6.6	41	7.9	41
Dadda Multiplier	8.8	102	10.3	102
5-stage Dadda Multiplier	1.4	100	1.8	100

TABLE II
Utilization of different topologies of MAC used in PU

Design type	Artix-7		Zynq-7	
	Slice registers (Out of 134600)	Slice registers (Out of 269200)	Slice registers (Out of 53200)	Slice registers (Out of 106400)
PU using non-pipelined MAC	531	779	531	779
PU using Pipelined MAC	515	796	515	796
PU using Proposed pipelined MAC	517	962	517	962

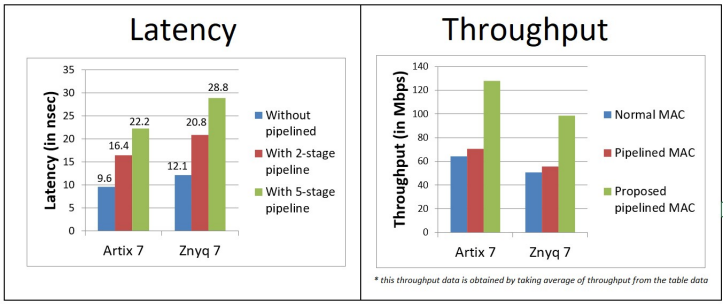


Fig. 2. Comparison of latency and throughput for different MAC used in PU

References

[1] C. V. Ramamoorthy and H. F. Li, "Pipeline architecture," *ACM Computing Surveys (CSUR)*, vol. 9, no. 1, pp. 61–102, 1977.

[2] L. Bai, Y. Lyu, and X. Huang, "A unified hardware architecture for convolutions and deconvolutions in cnn," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[3] B. Cope *et al.*, "Implementation of 2d convolution on fpga, gpu and cpu," *Imperial College Report*, pp. 2–5, 2006.

[4] A. Bhadra and S. Samui, "Design and analysis of high-throughput two-cycle multiply-accumulate (mac) architectures for fixed-point arithmetic," in *IEEE Calcutta Conference (CALCON)*. IEEE, 2022, pp. 267–272.

[5] Y. Liang, L. Lu, Q. Xiao, and S. Yan, "Evaluating fast algorithms for convolutional neural networks on fpgas," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 4, pp. 857–870, 2019.

[6] J. Wang, J. Lin, and Z. Wang, "Efficient convolution architectures for convolutional neural network," in *2016 8th International Conference on Wireless Communications & Signal Processing (WCSP)*. IEEE, 2016, pp. 1–5.

[7] M. Mody, M. Mathew, S. Jagannathan, A. Redfern, J. Jones, and T. Lorenzen, "Cnn inference: Vlsi architecture for convolution layer for 1.2 tops," in *2017 30th IEEE International System-on-Chip Conference (SOCC)*. IEEE, 2017, pp. 158–162.

[8] M. Chanda, S. Jain, and A. Dandapat, "Implementation of modified low power 8 x 8 signed dadda multiplier," *International journal of electronics and electrical engineering*, vol. 11, no. 01, pp. 8–14, 2010.

Authorized licensed use limited to: Universiti Malaysia Perlis. Downloaded on September 09,2026 at 08:14:27 UTC from IEEE Xplore. Restrictions apply.