International Journal of Modern Electronics and Communication Engineering (IJMECE) ISSN: 2321-2152
Volume No.-7, Issue No.-3, May, 2019

Optimizing the Convolution Operation for Image Processing using Efficient MAC unit

Pooja Nayak. N
Mtech Student, ECE
Bangalore Institute of Technology, India
Poojanayakn056@gmail.com

Dr. Vijay Lakshmi.D
Assistant Professor, ECE
Bangalore Institute of Technology, India
vijibit@gmail.com

Abstract: As convolution contributes most activities in picture handling administrators, the convolution influences the performance and execution of a CNN accelerator. Convolution includes multiplication and accumulator tasks with four dimensions of Loop, which results in a vast plan space. Earlier works either utilize constrained Loop improvement strategies, e.g., Loop unrolling, tiling, and Interchange, or just tune a portion of the plan factors after the accelerator architecture and data flow are fixed. Without completely concentrating the convolution Loop advancement before the equipment configuration stage, the subsequent accelerator can scarcely abuse the information reuse and exploit data management effectively. This Project beats these hindrances by quantitatively investigating and advancing the structure goals (e.g., memory access) of the CNN accelerators dependent on different plan factors. Then, Project aims to propose a specific data flow of CNN acceleration to minimize the data communication while maximizing the resource utilization to achieve high performance.

Keywords: Accelerator architectures, Convolution operation, Loop optimization, MAC unit, FPGA

I. INTRODUCTION

The field-programmable Gate array (FPGA) are quick tuning into the stage of decision for re-configurability also, shorter structure time over application-specific integrated circuits (ASICs). The advanced FPGAs permit customization of the design and can exploit the accessibility of hundreds to a large number of on-chip DSP blocks. However, critical challenges stay in mapping CNNs onto FPGAs. The state of-the-art CNNs require Matrix multiplication over a large number.

The restricted computational assets also, capacity limit on FPGA make the task of optimal mapping of CNNs (e.g., minimizing latency subject to energy constraints or vice versa) a complex and multidimensional optimization problem. The surprising expense of off-chip correspondence is another real obstacle to accomplishing higher execution and lower energy. For these reasons, energy-efficient hardware acceleration of CNNs on a FPGA requires simultaneous maximization of resource utilization and data reuse, and minimization of data communication.

Over 90% of the activities in a CNN include convolutions. In this way, Thus project concentrate on the administration of parallel calculations and the organization of data storage and access over different dimensions of memories, e.g., off-chip dynamic random access memory (DRAM), on-chip memory, and local registers.

This offers Large design space comprising of different decisions for executing parallelism, sequencing of calculations, and partitioning the large data set into smaller chunks to fit into on-chip memory. These issues can be taken care of by the current loop optimization techniques, such as loop unrolling, loop tiling, and loop interchange. Without fully studying the loop operations of convolutions, it is difficult to efficiently customize the dataflow and architecture for high throughput.

The design objectives of CNN accelerators (e.g., latency, memory) are quantitatively estimated based on the configurations of the design variables. An efficient convolution acceleration strategy and dataflow is proposed aimed at minimizing data communication and memory access.

II. CONVOLUTION OPERATIONS

Convolution is the principle activity in CNN calculations, which includes 3-D multiply and-Accumulate (MAC) activities of input matrix. To effectively map and perform the convolution loop, three loop enhancement systems to be specific, loop unrolling, loop tiling, and loop exchange, is utilized to customize the calculation.

1) Loop Unrolling: Unrolling convolution loop provides different parallelization of calculations, which influences the ideal PE for information reuse opportunities and memory access designs.

2) Loop Tiling: On-chip memory of FPGAs is sufficiently expansive to store the whole information of CNN calculations. Subsequently, it is sensible to utilize denser outside DRAMs to store the matrix and the intermediate results of all layers. loop tiling is utilized to partition the whole information into numerous blocks, which can be used in the on-chip memory. With appropriate assignments of the loop tiling size, the area of information can be expanded to decrease the number of

DRAM accesses, which causes long latency and high-power utilization.

3) Loop Interchange: Loop interchange decides the request of the consecutive calculation of the four convolution loops. There are two sorts of loop exchange, to be specific, intratile and intertile loop orders. Intratile loop order determines the pattern of data movements from on-chip buffer to PEs. Intertile loop order determines the data movement from external memory to on-chip buffer.

Convolution Operation includes multiplication and addition hence as the picture measure expands the quantity of adders and multipliers required will be expanded as shown in fig 1 . In this way result in an increased area and furthermore memory unit. Consequently there is a requirement for improved convolution operation and reusage of adders and multiplier blocks.

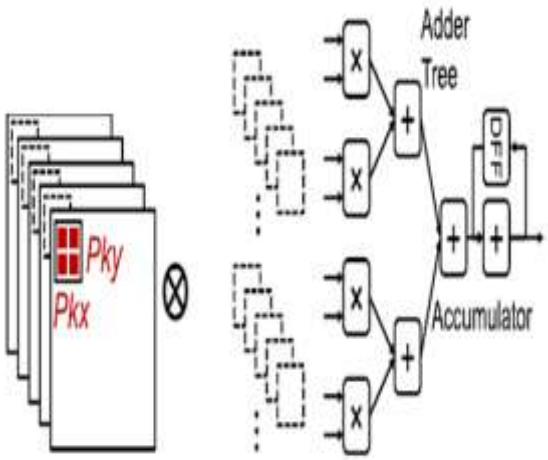

Fig 1 Convolution Operation for Image

III. PROPOSED METHOD

In investigation, both the information and intermediate results are thought to be put away in outer memory (DRAM), which is a need when mapping extensive scale CNNs on moderate FPGAs. The expenses of DRAM get to higher latency and power consumption than on-chip memory gets to and consequently it is critical to reduce the use of outer memory gets to improve the latency and power consumption effectively. The minimum number of DRAM accesses is achieved by having sufficiently large on-chip buffers and proper loop optimization of convolution operation. General CNN Accelerator is shown in fig 2

As detailed CNN calculations include an large sum of information and loads. For them, the on-chip memory is lacking to store every information, thus requires an external memory. In this way, CNN accelerator comprises of three dimensions of storage: 1) external memory; 2) on chip memory; and 3) Registers

The major stream is to get data from outside memory to on-chip memory, and after that feed them into registers and PEs. After the PE computation completes, results are traded back to on-chip memory and to the external memory if required, which will be used for consequent layer.

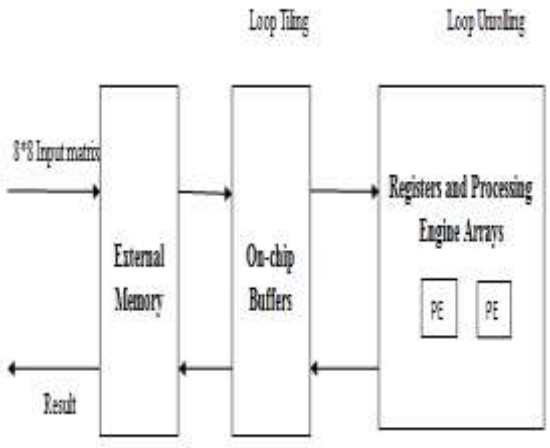

Fig 2 CNN accelerator

The Image is converted to matrix and taken as an input for CNN Accelerators. The Matrix is processed in Mac unit. MAC unit performs multiplication and accumulation process. Basic MAC unit consists of multiplier, adder, and accumulator as shown in fig 3

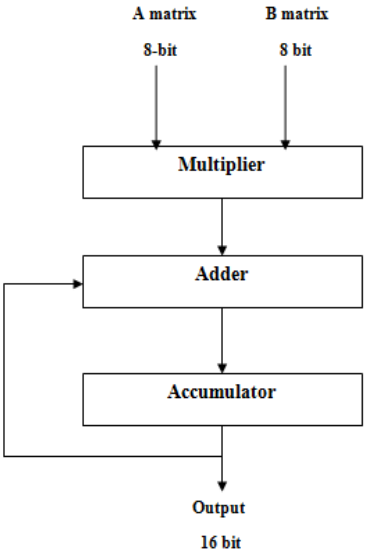

Fig 3 MAC UNIT

Reusing operators lessens the quantity of read tasks of on-chip memory. There are mostly two sorts of information reuse: spatial reuse and fleeting reuse. Spatial reuse implies that, in the way of perusing information from on-chip memory, a single operator is utilized for numerous parallel tasks inside one clock cycle. Then again, transient reuse implies that a single operator is utilized for various sequential cycles. In the Project we focused on MAC unit with various adders so it results in a lesser area and power.

IV. ANALYSIS OF EXPERIMENTAL RESULTS

The Image of size 1024*768 pixels is converted to 8*8 matrix using Matlab shown in fig 4. This Matrix is taken as an input to General CNN accelerator. The simulation result of CNN accelerator is shown in fig 5.The area can be reduced using Carry skip adder or Ripple carry adder in the MAC unit in comparison with formal MAC unit with adder and multiplier. The RTL Schematic is shown in fig 5.

Fig 4 Image converted to Matrix

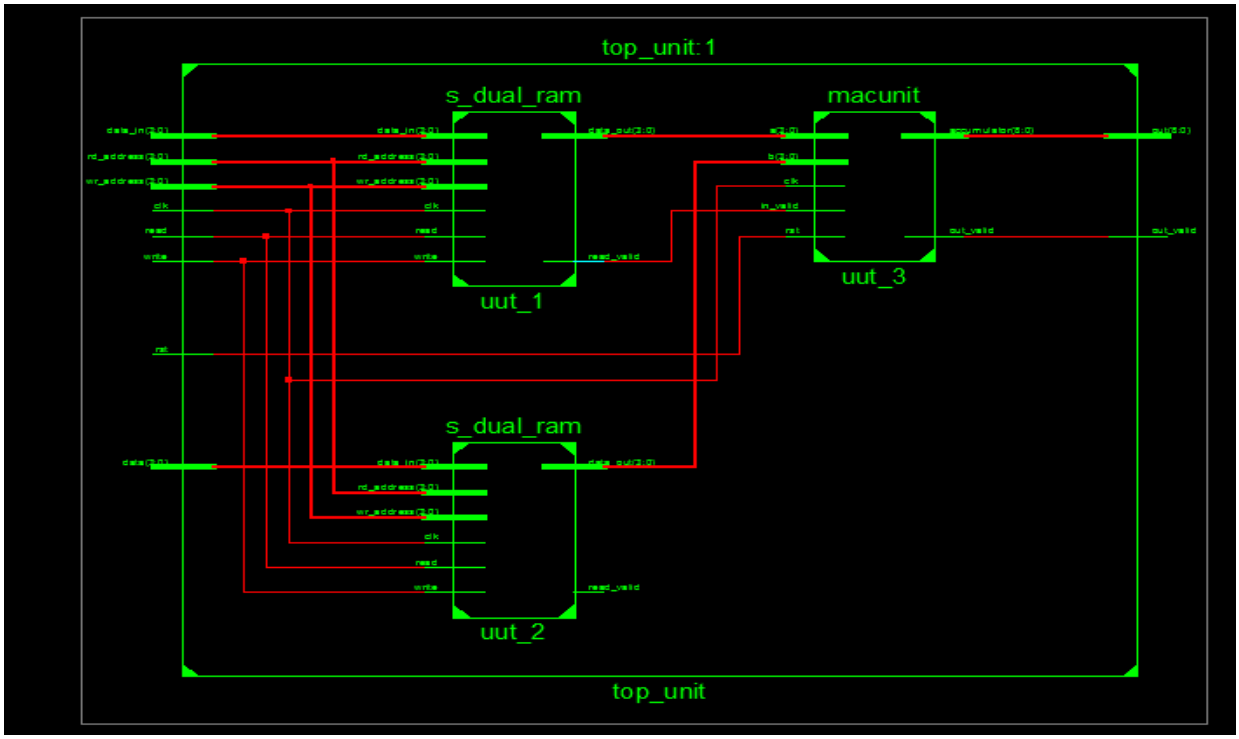

Fig 5 RTL view of CNN Accelerator

International Journal of Modern Electronics and Communication Engineering (IJMECE) ISSN: 2321-2152
Volume No.-7, Issue No.-3, May, 2019

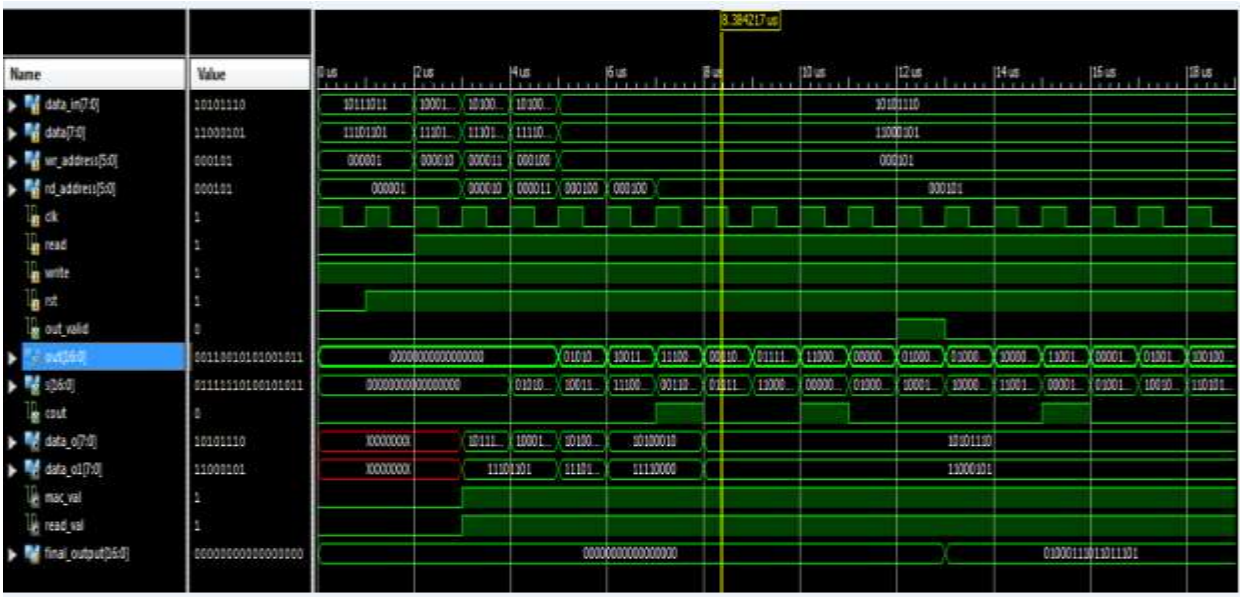

Fig 6 Simulation Result of CNN Accelerator

V. CONCLUSION

The CNN accelerator with convolution advancement and MAC unit with various adder architecture can increase the throughput and reduce region yet not really latency. Thus project proposed loop optimization method and exploited logic elements to implement efficient MAC unit. The relationship between accelerator objectives and design variables are quantitatively investigated. A corresponding new data flow and architecture is proposed to minimize area and enhance throughput.

REFERENCES

[1] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, 2015.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. Adv. Neural Inf. Process. Syst. (NIPS), 2012, pp. 1097–1105.

[3] M. Lin, Q. Chen, and S. Yan. (Mar. 2014). "Network in network." [Online]. Available: https://arxiv.org/abs/1312.4400

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. Int. Conf. Learn. Represent. (ICLR), 2015.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[6] D. F. Bacon, S. L. Graham, and O. J. Sharp, "Compiler transformations for high-performance computing," ACM Comput. Surv., vol. 26, no. 4, pp. 345–420, Dec. 1994.

[7] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energyefficient reconfigurable accelerator for deep convolutional neural networks," IEEE J. Solid-State Circuits, vol. 51, no. 1, pp. 127–138, Jan. 2017.

[8] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in Proc. ACM/IEEE Int. Symp. Comput. Archit. (ISCA), Jun. 2016, pp. 367–379.

[9] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays (FPGA), Feb. 2015, pp. 161–170.

[10] C. Zhang, D. Wu, J. Sun, G. Sun, G. Luo, and J. Cong, "Energy-efficient CNN implementation on a deeply pipelined FPGA cluster," in Proc. ACM Int. Symp. Low Power Electron. Design (ISLPED), Aug. 2016, pp. 326–331.

[11] N. Suda et al., "Throughput-optimized OpenCL-based FPGA accelerator for large-scale convolutional neural networks," in Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays (FPGA), Feb. 2016, pp. 16–25.

[12] U. Aydonat, S. O'Connell, D. Capalija, A. C. Ling, and G. R. Chiu, "An OpenCL deep learning accelerator on Arria 10," in Proc. ACM/SIGDA Symp. Field-Program. Gate Arrays (FPGA), Feb. 2017, pp. 55–64.

[13] K. Guo et al., "Angel-Eye: A complete design flow for mapping CNN onto embedded FPGA," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 37, no. 1, pp. 35–47, Jan. 2018.

[14] Y. Ma, N. Suda, Y. Cao, J.-S. Seo, and S. Vrudhula, "Scalable and modularized RTL compilation of convolutional neural networks onto FPGA," in Proc. IEEE Int. Conf. Field-Program. Logic Appl. (FPL), Aug./Sep. 2016, pp. 1–8.

[15] Y. Ma, Y. Cao, S. Vrudhula, and J.-S. Seo, "Optimizing loop operation and dataflow in FPGA acceleration of deep convolutional neural networks," in Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays (FPGA), Feb. 2017, pp. 45–5.