Scalable Systolic Array Multiplier Optimized by Sparse Matrix

RiMing Jia[1], Tu Xu[1], YuChun Chang[1]

[1] School of Microelectronics, Dalian University of Technology, Dalian 116620, China
Email: jiariming@163.com

2021 IEEE 14th International Conference on ASIC (ASICON) | 978-1-6654-3867-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ASICON52560.2021.9620326

Abstract—Various artificial intelligence (AI) algorithms are proposed in recent years, the demand for computing complexity has also increased. Matrix multiplication is a computing unit commonly used in AI calculations. This paper proposes a novel sparse matrix multiplication optimized scalable systolic array multiplier, which has three characteristics compared to traditional matrix multipliers: 1. This paper deals with a sparse matrix multiplication optimization multiplier, which is achieved by the critical path. 2. This paper focuses on high-dimensional convolutional neural network computation and proposed a multiplier with an expandable feature, which can calculate the matrices that dimension below the multiplier itself. Compared with the inextensible multiplier, this improvement reduces power consumption when calculating low-dimensional data. 3. The matrix multiplier of the traditional systolic array has a pulsation relationship between columns, which introduced delay registers between adjacent columns. Therefore, we proposed a new structure based on the traditional systolic array which removes the delay module. We designed a 4*4 matrix multiplication and deployed it at the PYNQ-Z7020 field-programmable gate array (FPGA). The result shows that the proposed structure reduces the 9.2% calculation delay, saving 13.3% slice logic power consumption than the traditional multiplication.

1. Introduction

In recent years, the computing complexity of artificial intelligence (AI) has increased rapidly with the increase of algorithm complexity. Most of the AI algorithms are trained and deployed in the personal computer (PC). The traditional computer architecture is based on the Von Neumann system, which greatly limits the further development of computation and throughput. This limitation is generated by the complexity of data calculations brought by AI algorithms. This paper proposes a new structure of matrix multiplier to satisfy this requirement. In this paper, we deployed the structure that we proposed in a field-programmable gate array (FPGA).

This paper proposed a novel matrix multiplication with three changes. Firstly, we improve the systolic array structure. Figure 1 shows the traditional systolic array structure proposed in [6]. Matrix1 (M1) represents the multiplier matrix, and M1 (m, n) represents the data in the m-th row and n-th column of the multiplier matrix M1. Matrix2 (M2) has the same representation as M1. Multiplier (MUL) stands for the multiplier unit, which can multiply the input data and output the result. Register (REG) represents the register delay module, which is used for the systolic data pulse in this structure. Output_colum1 (OUT_col1) represents the output popped by the pipeline in the first column. The traditional systolic array structure has a pulsation register between adjacent columns to ensure that the timing of the pipeline calculation results of each column is ordered. The structure we proposed removes the delay registers between the columns and instead connects the input matrix multiplier in the same row. Figure 3 shows the improved systolic array structure. The second contribution of this paper is the optimization of sparse matrix multiplication.

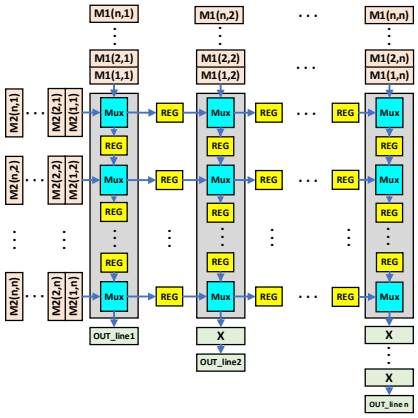

Fig. 1. Traditional Systolic Array Structure

In the field of AI algorithms, especially convolutional neural networks, the weight data has the characteristic of sparseness. Figure 2 takes YOLO v3 convolutional neural network as an example to extract part of its weight data. Figure 2 shows part of the sparse characteristic of yolov3 weights. Using traditional matrix multiplication to calculate the sparse matrix will causes elements with uneven computing resources. This problem reflected in the digital circuit system is that it will lead to a longer calculation path. Considering that there are many weight values is 0 (or extremely close to 0) in the sparse matrix. We optimize the sparse matrix computation by designing the calculation module of the compute unit

978-1-6654-3867-4/21/$31.00 2021 IEEE

(CU). CU is a basic operation unit of the proposed matrix multiplication multiplier, which is responsible for accelerating the sparse matrix multiplication, sparsity optimization, and scalability features.

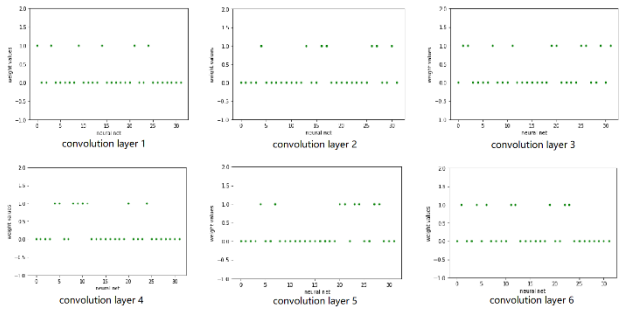

Fig. 2. Convolution Layer Weights' Sparse Analysis

The third improvement is that we design a scalable matrix multiplier. As stated before, the data dimension of AI algorithms is getting higher. Therefore, we proposed an expandable CU unit. Figure 5 shows the structure of the CU. Compared with the traditional multiplier, which cannot be expanded, the design we proposed has the characteristic of low power consumption and calculation high speed. The contributions are listed below:

- Improved the systolic structure, which makes the systolic array behaves better in timing.
- Come up with a sparse matrix multiplication optimized Compute Unit, which not only reduces the consumption but also makes it calculate faster.
- Designed a scalable feature, which is easier to calculate the high-dimension data.

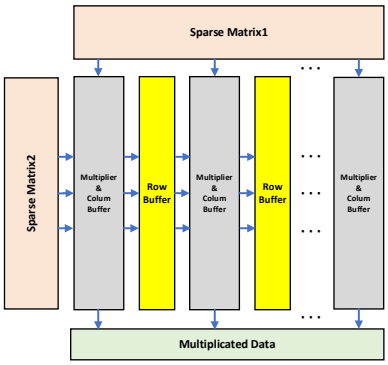

Fig. 3. Overview of Proposed Structure

2. Related work

Xilin Yi et al. proposed a low supply voltage digital signal processing (DSP) with error-resilient multiplication at [1]. The structure which [1] proposed reduces the power consumption by 26.7% compared with other multipliers. Qi Nie of Princeton University proposed a memory-driven data flow optimization framework at [2]. The improvement that [2] adopted is to reduce the access mode of the sparse matrix. The proposed framework is tested in a graphics processing unit (GPU) platform. Compared with the state-of-art matrix multiplier at the time (2019), the experiment results show that the novel framework has improved performance by three times, and reducing 5X static random-access memory (SRAM) data access. Reference [3] proposed a new type of INT8 calculation multiplier for AI computation, which uses memory and logic resources to balance the resources and power consumption of FPGA. Bahar et al. proposed a fast and scalable systolic array multiplier and deployed it on FPGA at [4].

Reference [5] proposed a new method to optimize matrix format conversion whose value is consists of 1 and 0. Jeng-Shyang Pan et al. optimizes the systolic array multiplier algorithm based on Toeplitz matrix-vector product (TMVP) at [6] and verified the optimized algorithm in the low-complexity systolic multiplication proposed in [6]. Reference [7] proposed a theory of multiplication algorithm based on FPGA. According to the experiment carried out in [7], the algorithm reduced the look-up table (LUT) resources by 41% and reduces route time by 5%, compared with the traditional 8-bit*8-bit multiplication. Linhuai Tang proposed a method that can handle multiple arbitrary length sets in [8]. Paper [11] focus on the critical path delay and proposed a placement algorithm for FPGA architectures, which can also improve the behavior of the matrix multiplier. Paper [13] proposed a fast block placement of FPGA which systolic arrays are resource-intensive based on evolutionary algorithms.

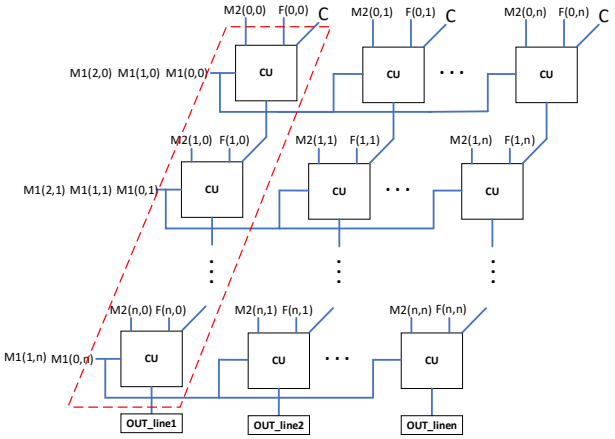

Fig. 4. Proposed Systolic Array Detailed Circuit

3. Improved Systolic array multiplier

A. Sparse Matrix Optimization

For large-scale convolutional neural networks, the matrices formed by the weight parameters are highly sparse. Sparse matrices have the characteristics of scattered data distribution and highly dimensioned data. A

sparse matrix is a matrix that the ratio of zero values to the total elements number of the matrix is greater than 0.95. The remaining non-zero elements are scattered in the matrix. A sparse matrix is widely used in the field of modern scientific computing. The convolutional neural network is a mainstream method for tasks such as object recognition and classification. More than 70% of weight matrixes are sparse matrixes. Therefore, neural network computation needs to optimize the multiplication between sparse matrixes. In this paper, we propose a novel sparse matrix multiplication optimization structure, which can reduce the power consumption in sparse matrix multiplication. Figure 6 shows the structure of the sparse matrix optimization circuit. The red rectangular box is the detailed calculation circuit of CU. Figure 3 shows the overview of the proposed structure. Sparse matrix1 and sparse matrix2 are input matrices. Figure 4 shows details about the systolic array's circuit. Figure 5 shows the gate level circuit of CU.

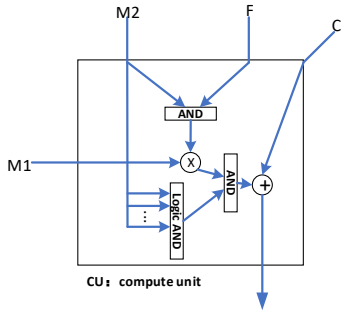

Fig. 5. The Structure of Compute Unit

In this paper, we designed an 8-bits input AND logic unit to achieve sparse matrix optimization. M2n represents the n-th position of matrix2. S_OUT is the output data of the logic gate. The relationship between M2n and S_OUT is shown in formula (1). All the elements are 8-bits.

$$S_{OUT} = M20\&M21\&M22 \ldots \&M2(N-1)\&M2(N) \quad (1)$$

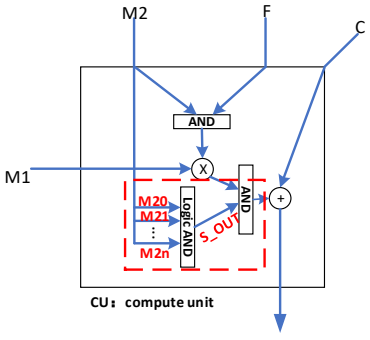

Fig. 6. Sparse Matrix Optimization Circuit

We supposed that there is a "0" value in Matrix2's elements. The structure cannot recognize the situation of elements' value is "0" without the sparse matrix optimization Compute Unit. It will transport the output data to port S_OUT normally. Compared to that, the structure that we proposed will judge the element firstly. Then, the Compute Unit will change the calculation path to the output. Therefore, our sparse optimized circuit can reduce the transmission delay when the input element is "0". We have declared before that zero elements are commonly consist in a sparse matrix, which makes zero-elements optimization is very important. In another situation. When the input elements are not "0". the Computing Unit would act like a normal multiplier calculator. The output of sparse-optimized circuit S_OUT should output logic "1", which cannot influence the final output of CU.

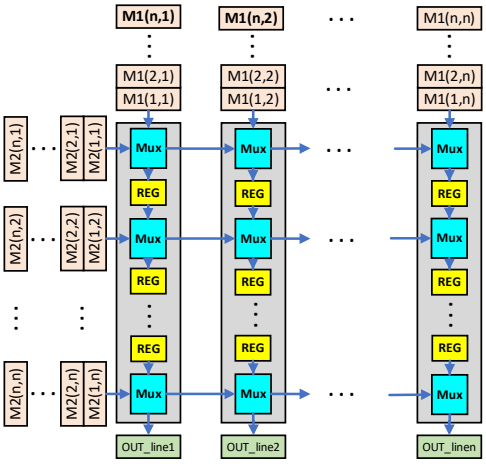

Fig. 7. Proposed Systolic Array Structure

B. Improved Systolic Array Structure

The traditional multiplicative systolic array is shown in Figure 1. Delay registers is designed between different columns to ensure the correctness of output timing. Inspired by the pipeline design method, we remove these delay registers. So the calculation system can output data at the same time. The improved systolic array structure used for matrix multiplication is shown in Figure 7.

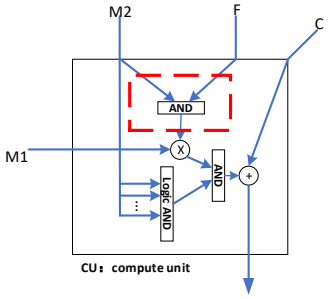

Fig. 8. The Circuit of Scalable Part

In the traditional multiplying systolic array, each column pipeline is not activated at the same time. The pipeline that starts firstly must be started after waiting for the end of the calculation. Which greatly limits the throughput of the computing system. The proposed systolic array structure removes the delay registers module between different columns, which makes every pipeline starts at the same time. Another benefit is that pipelines can pop-out data together, which reduced the latency of the matrix-multiplication system.

C. Scalable feature

A scalable multiplier is convenient for high dimension matrix multiplication. As stated before, the high dimension data is commonly used in the AI calculation system. Considering that the multiplication's dimension is wide. n.

The artificial Intelligence calculation shows that the data dimension is getting deeper and deeper in recent years. More and more high-dimensional metrics are required to be computed. We proposed a new scalable control circuit to improve versatility, which is important for high-dimensional matrix multiplication. Figure 8. shows the detailed circle of the scalable part. F is an input-flexible control bit, which can determine the activation of the multiplier. The structure we proposed can also calculate the matrix that dimension is lower than its.

4. Discussion

We deployed our matrix multiplication system and the traditional matrix multiplication in PYNQ-z7020 FPGA. Table 1 shows the usage of slice logic. The result of design compiler (DC) is shown in table 1. Compared with the resource and performance optimization reduction structure proposed in paper [8], the matrix multiplication we designed to use less slice logic resource. Table 2 shows the result of simulation in vivado 2020.1. As shown in table 2, compared with the traditional multiplication the structure we designed reduced 9.4% calculation delay in sparse matrix multiplication. Besides, the slice logic power dissipation is reduced by 13%. The delay has been reduced by 9.4%. As mentioned before, the reduction is consistent with our assumptions that critical data path optimization can improve the performance of delay.

Table 1. Result of DC

Combination cells	1826	Combination area	37088
Sequential cells	248	Buf/Inv area	1686
Slack (MET)	56.06	Total cell area	51398
Proposed Structure slices	1270	Struture proposed in [8] slices	1390[8]

Table 2. Simulation Result

Multiplier structure	On-Chip Data Path Delay	Slice Logic Used
Traditional	6.977 ns	1112
Sparse matrix optimized	6.365 ns	1270

5. Summary

This paper proposed a novel matrix multiplier with AI calculation optimization. To decrease the calculation delay, we improved the traditional matrix multiplier systolic structure and proposed an optimization according to the sparse matrix multiplier. The structure we proposed reduced the 9.4% calculation delay. We also reduced the On-Chip power consumption, which was achieved by making some changes based on the traditional systolic matrix multiplier. As shown in table 1, after we improved the structure, its power consumption reduced by 1.2W.

References

[1] Yi X , Pei H , Zhang Z , et al. Design of an Energy-Efficient Approximate Compressor for Error-Resilient Multiplications[C]// 2019 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2019.

[2] Nie Q , Malik S . SpFlow: Memory-Driven Data Flow Optimization for Sparse Matrix-Matrix Multiplication[C]// 2019 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2019.

[3] Langhammer M , Gribok S , Baeckler G . High Density 8-Bit Multiplier Systolic Arrays For Fpga[C]// 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2020.

[4] Asgari B , Hadidi R , Kim H . Proposing a Fast and Scalable Systolic Array for Matrix Multiplication[C]// 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2020.

[5] Petkov N. Systolic arrays for matrix I\/O format conversion[J]. Electron. Lett. 1988.

[6] Pan J S , Lee C Y , Sghaier A , et al. Novel Systolization of Subquadratic Space Complexity Multipliers Based on Toeplitz Matrix – Vector Product Approach[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019:1614-1622.

[7] Wibowo F W. Comparison of multiplication algorithms based on FPGA[C]//2018 2nd Borneo International Conference on Applied Mathematics and Engineering (BICAME). IEEE, 2018: 326-331.

[8] Tang L, Cai G, Zheng Y, et al. A resource and performance optimization reduction circuit on FPGAs[J]. IEEE Transactions on Parallel and Distributed Systems, 2020, 32(2): 355-366.