Design of a Coarse-grained Processing Element for Matrix Multiplication on FPGA

Yuichi Okuyama
Dept. of Computer Science and Engineering
The University of Aizu
Aizuwakamatsu, Japan
okuyama@u-aizu.ac.jp

Shigeyuki Takano
On semiconductor
Gunma, Japan
shigeyuki.takano@onsemi.com

Tokimasa Shirai
NJK Corporation
Tokyo, Japan
s1170239@gmail.com

***Abstract*—In this paper, we discuss and evaluate about a grain size of the PE of a matrix operation specific architecture with fused multiply add (FMA) units, RapidMatriX, on FPGAs. Recent FPGAs have many DSP blocks which are high-performance arithmetic units. Hereby, implementing functional units for matrix operation to array structure of the RapidMatriX, we propose to use DSP blocks efficiently by increasing grain size of FMA unit. We implement the RapidMatriX using the refined PEs on an FPGA. In addition, we evaluate the clock frequencies and the clock cycles of calculation. As a result, throughput of the PE for 4×4 matrix FMA is 3.14 times in comparison with the original PEs of scalar FMA for 8×8 matrix multiplication.**

***Keywords*-FPGA, matrix multiplication, SIMD processor;**

I. Introduction

Matrix operations have a keyrole for large-scale scientific computations such as fluid dynamics, and structural analysis etc. [1], [2]. These applications require a tremendous amount of matrix operations. Since early times, a lot of techniques for matrix operations have been proposed [3]. Many modern processors have a fused multiply add (FMA) instruction as a solution, such as Intel Itanium2[4], and Sony/Toshiba/IBM Cell B.E. [5]. An FMA instruction calculates $a \times b + c$ within a single latency of processor cycle. Though FMA helps fast matrix processing, the instruction must be executed at least n^3 times for an $n \times n$ matrix multiplication. This means the clock frequency dominates the speed of matrix processing in a single processor core. On the other hand, some architectures for matrix multiplication have been proposed to overcome this problem. These architectures have multiple processing elements (PEs) connected by linear, ring , mesh, torus, and hypercube topologies. RapidMatriX [6] is an architecture organized by a two-dimensional PE array with the torus topology. This architecture achieves fast matrix operations executing the FMA in parallel. Throughput of the multiplication is scalable, which means we can get better performance by increasing PEs. Sato et al. implemented RapidMatriX on a Field-Programmable Gate Array (FPGA), since FPGA has advantages in cost and reconfigurability. Instead of containing the potential of fast matrix processing, the implementation is not well optimized; it only shows the feasibility from the theory. Therefore, we focus on DSP (digital signal processor) blocks [7] of recent FPGAs. A single DSP block has several arithmetic circuits and inputs. In accordance with this structure, we enable PE to use DSP blocks efficiently by increasing the amount of calculation in each PE of RapidMatriX. Hereby, we aim to accelerate matrix operations. In this paper, we propose to merge scalar FMAs into an integrated matrix FMA of 2×2 and 4×4 in ALU of the PE in order to use the DSP blocks of the FPGA efficiently and achieve fast matrix operations.

II. RapidMatriX

RapidMatriX is a special processor for matrix operations, and consists of control unit and PE Array. Figure 1 shows a schematic diagram of RapidMatriX.

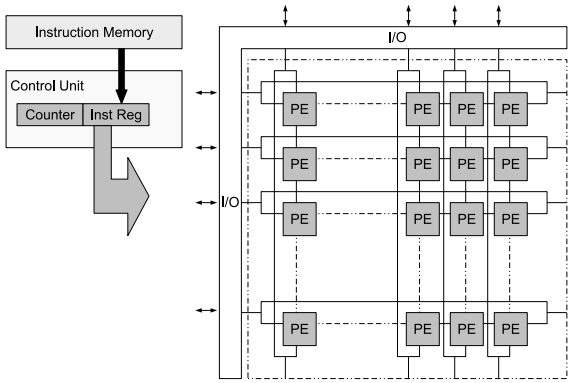

Figure 1. Schematic diagram of RapidMatirX

PE array employs the structure of systolic array. This structure has the following characteristics.

1) Structure organized by simple cells.
2) Control flow for simple regular data and control.
3) High-throuput calculation by parallel and pipeline.
4) Convolution of input, output and processing.

A. 2-Dimensional Torus

The PE array of RapidMatriX is organized by two-dimensional torus. All PEs connect with adjacent PEs in two-dimensional torus. Each PE executes FMA operation as Equation (1). Calculation of matrix multiplication is

978-1-4799-4305-0/14 $31.00 © 2014 IEEE
DOI 10.1109/MCSoC.2014.41

IEEE computer society

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on December 07,2020 at 13:51:02 UTC from IEEE Xplore. Restrictions apply.

achieved by executing FMA operations as shifting a_{ij} and b_{ij} of matrix elements in a horizontal and vertical direction respectively. This calculation steps is repeated n times for $n \times n$ matrix multiplication. Finally, the results of the matrix multiplication is kept in PEs.

$$FMA(a, b, c) : d = a \times b \ (\pm) \ c \tag{1}$$

a,b,c and d are scalar data

An I/O port on left and top edge can input data from outside by two-dimensional torus topology. In input data operation, RapidMatirX inputs data from left edge and top edge of torus topology shifting data in all PEs. It means RaipdMatriX can input to PE array of $n \times n$ from outside in n steps.

B. Structure of PE

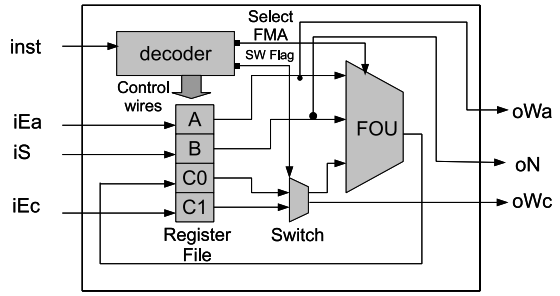

Figure 2. Structure of PE

As shown in Figure 2, a PE conteins register file, instruction decoder and Fused Operation Unit (FOU). A PE has input ports for data from south and east cells and for instructions. A PE has three output ports for data to west and north cells. The egister file consists of four register groups: three of them store data from input, and the last one stores result data from calculation in FOU. The instruction decoder determines the behavior of circuits by issuing control wires to register file, FOU and several selector, after instruction decode.

C. Instruction Set

PE is controlled by SIMD operations. Control Unit issues same instructions to all PEs, and PEs execute same processes. RapidMatriX support four instruction types as *LOAD*, *MOVE*, *FOU* and *NOP*.

- *LOAD* : Input data to the PE
- *MOVE* : Data transfer between adjacent PEs
- *FOU* : Calculation and data transfer between adjacent PEs
- *NOP* : No operation

LOAD instruction has the following two types.

- a : To write immediate data to all PEs
- b : To write immediate data to a selected single PE

In both processes, RapidMatriX can store an immediate data to an intended PE by selecting register type, register number and immediate data in instruction field. In addition, (b) selects PE for writing data by two parameters. In order to transfer data to adjacent PEs, *MOVE* instruction selects a source and a forwarding address of register. This instruction does not select a forwarding address of the PE because data is only transfered to adjacent PEs. *FOU* instruction selects register of input and output, in order to execute calculation and the process of *MOVE* instruction simultaneously. *NOP* instruction is utilized for controlling timing.

D. Calculation Procedure

This section shows the calculation procedure of RapidMatriX. The calculation procedure of matrix operations is as follows.

1) Execute skew operation (mentioned later) to matrix multiplicand and matrix multiplier
2) Input matrix multiplicand and matrix multiplier to each PE by *MOVE* instructions
3) Execute FOU instructions

First, the matrix is transformed by skew operation to execute matrix operation on torus topology. Skew operation changes general matrix form into matrix form for matrix operations utilizing torus. In skew operation, the row (i) of an A of matrix multiplicand shifts in $(i-1)$ left direction while the line (j) of a B matrix multiplier shifts in $(j-1)$ under direction. Therefore, after skew operation, if A and B is $n \times n$ matrices, these matrix elements are represented as following.

$$\begin{aligned} a'_{ij} &= a_{ij'(i,j)} \\ b'_{ij} &= b_{i'(i,j)j} \end{aligned} \tag{2}$$

where

$$j'(i,j) = \begin{cases} j-i, & \text{if } (j-i) \geq 1 \\ j-i+n, & \text{otherwise} \end{cases}$$

$$i'(i,j) = \begin{cases} i-j, & \text{if } (i-j) \geq 1 \\ i-j+n, & \text{otherwise} \end{cases}$$

On $n \times n$ PE array, RapidMatriX can calculate matrix multiplication by n times executing FOU operations, after transfer two matrices data that are performed by *MOVE* operations. For 2×2 matrix multiplication, RapidMatriX can calculate by skew operations shown in Figure 3 and *FOU* operations shown in Figure 4.

III. Coarse-grained PE for RapidMatriX

In this section, we consider coarse-grained PE adopted for structure of the recent FPGA. We define the matrix FMA which calculates fused multiply-add operation of matrix. Then, we will propose new structure for matrix multiplication using matrix FMA.

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on December 07,2020 at 13:51:02 UTC from IEEE Xplore. Restrictions apply.

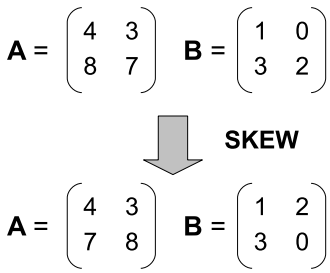

Figure 3. Skew operation for 2×2 matrix

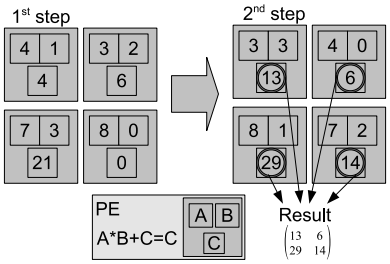

Figure 4. *FOU* operations of 2×2 matrix multiplication

A. DSP Blocks of FPGA

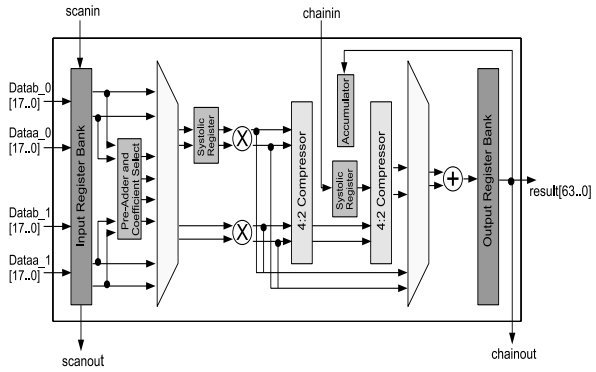

Figure 5. Structure of DSP block (Stratix V)

Figure 5 shows recent FPGA DSP block. A circuit for the FPGA architecture can execute high-performance processes using DSP blocks. A DSP block has several inputs and arithmetic circuits. Two or more blocks can be combined. In contrast, using simple circuits, cannot use all resources of the DSP.

B. Matrix FMA

Data transfer and FMA operation of each PEs are performed as scalar data. In this section, we propose a scheme to combine multiple PEs. We define A, B, C and D as the $n \times n$ matrix data, and matrix elements as a_{ij}, b_{ij}, c_{ij} and d_{ij}, in addition to multiply-accumulation of matrices $D = AB + C$ is defined as a matrix FMA. Matrix FMA is shown in following equation.

$$FMA(A, B, C) : d_{ij} = \left(\sum_{k=1}^{n} a_{ik} b_{kj} \right) (\pm) c_{ij} \tag{3}$$

We assume that the row and column are divided by m when executing an $n \times n$ matrix multiplication. The equation shows $(n/m) \times (n/m)$ matrix operations are required to calculate an $n \times n$ matrix operation.

Now, We define original matrices as A, B and C, and sub matrices as A'_{ij}, B'_{ij} and C'_{ij} $(1 \leq i \leq n/m, 1 \leq j \leq n/m)$. The calculation procedure of $C = AB$ can be descrived same form as alculation procedure for scalar matrix operations.

Altogether, FMA operation is shown in the following equation.

$$C'_{ij} = \sum_{k=1}^{n/m} A'_{ik} B'_{kj} \tag{4}$$

By using Equations 3 and 4, we can replace scalar FMA of current PE to matrix FMA. Moreover, we can change the grain size of PE array by replacing scalar data with matrix data.

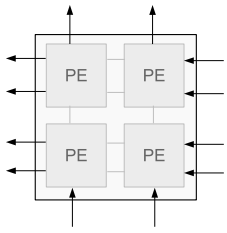

Figure 6. Single PE that is implement the FMA of 2×2

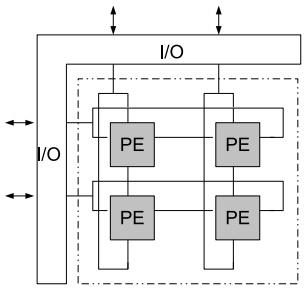

Figure 7. Schematic diagram of the PE Array using 2×2 matrix FMA for 4×4 matrix operations.

Figure 6 shows a single PE of 2×2 matrix FMA. Figure 7 shows structure of PE array for 4×4 matrix operation using 2×2 PEs. In this array, transferred scalar data become 2×2

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on December 07,2020 at 13:51:02 UTC from IEEE Xplore. Restrictions apply.

matrix data. The topology and a structure of control data are the same as the current PE array for 2×2 matrix operation.

C. Calculation Procedure with matrix FMA

We must change the calculation procedure to execute the calculation using matrix FMA. First, we must change skew operation by replacing scalar data between each PE. We simply apply block matrix defined in Equation (4) to Equation (2).

Next, we change scalar data of FOU instruction to matrix data in transferring data.

For example, using FOU of 2×2 matrix FMA for 4×4 matrix operation, Figure 8 shows the skew operation, and Figure 9 shows multiplication processes. This calculation finish in 2 steps.

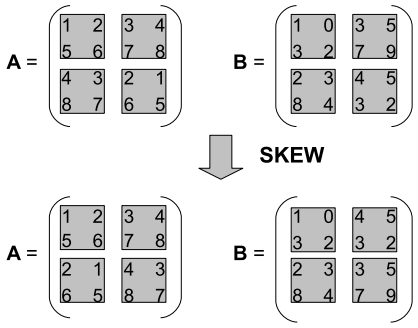

Figure 8. Skew operation (2×2 matrix)

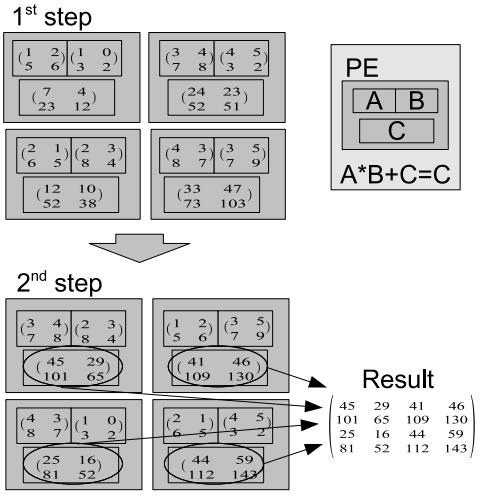

Figure 9. FOU operation (2×2 matrix)

D. Advantage of Proposed PE

RapidMatriX can increace the utilization of arithmetic units in DSP by using matrix FMA in FOU. Therefore, we assume that RapidMatriX can reduce calculation steps. The $n \times n$ matrix multiplication can be calculated in n steps, using n^2 PE with scalar FMA. On the other hand, this calculation can be calculated in n/m steps, using $(n/m)^2$ PE with $m \times m$ matrix FMA. Altogether, this way means to reduce the necessary calculation steps by increasing grain size of matrix FMA. Moreover, when implementing matrix FMA on Stratix V [8], the amount of utilization of DSP is smaller than current PE array using scalar FMA, because of increasing the amount of calculation within a single step. For those reasons, we estimate that RapidMatriX can perform high performance calculation if we increace the size and number of PEs using matrix FMA. In following, we implement a new RapidMatriX by using PE which calculates matrix operations.

E. Implementation for the Proposed PE

In this section, We describe implementation of FOUs: scalar matrix, 2×2 matrix, and 4×4 matrix FMA. We also describe how to implement PE array using these FOU in order to consider performance of proposed PE. We use minimal I/O Buffers to evaluate the results.

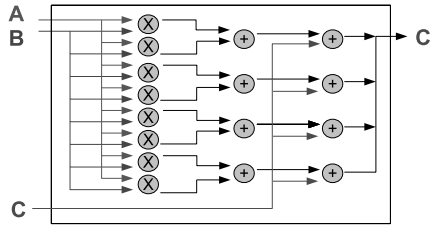

Figure 10. 2×2 matrix FMA

1) Data bus: Matrix elements are represented in 16 bit in this implementation. In data bus for PEs containing 2×2 matrix and 4×4 matrix FMA, the data buses contains 64 bits and 256 bits of data to connect each PE.

2) Register File: Register file is implemented by register groups of A, B, $C0$ and $C1$. These are described in Section 2. In order to realize multiplication with matrix FMA shown in Figure 10, A, B, $C0$ and $C1$ are 64 bit registers to conatin four 16 bit elements. Register file sends 192 data as three matrices to calculate $C = AB + C$ in FOU at one cycle of latency.

3) ALU: We implement three ALUs (FOUs); scalar FMA, 2×2 matrix FMA and 4×4 matrix FMA. The scalar FMA applies the FMA operation in 16 bits. On the other hand, 2×2 and 4×4 matrix FMA divide input data between 16 bits. Here, these data are matrix elements. Finding the results of calculation of each matrix element, ALU outputs matrix data of bit length same as input data.

IV. Evaluation

In this section, we design above circuit by Verilog HDL and evaluate the synthesis result on Quartus12.0 that is

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on December 07,2020 at 13:51:02 UTC from IEEE Xplore. Restrictions apply.

synthesis tool for Altera FPGAs. Moreover we show synthesis result for Stratix V (5SGXEA7N3F45C3). We evaluate following three circuits;

1) The results of synthesis of a single PE.
2) The results of synthesis of PE array for 8×8 matrix multiplication.
3) The number of 8×8 matrix multiplication which can be performed in a second.

Table I shows the synthesis results of a single PE for the FMA of matrix 1×1, 2×2 and 4×4. The amount of LUT increases in proportion to the amount of calculation of FMA. In addition, the clock frequency is decreasing gradually.

Matrix operations for the size of $m \times m$ uses $\lceil m^3/2 \rceil$ DSP blocks, since single block has two multipliers; therefore in these results, the number of DSP is 1, 4 and 32 in the matrix FMA of 1×1, 2×2 and 4×4, respectively.

Table II shows the maximum clock frequency and the amount of resources for 8×8 matrix multiplication. Here, the total number of DSPs is the number of DSPs in a single PE multipled by the number of PE. Also, 1×1, 2×2 and 4×4 clock frequencies were 332.89MHz, 286.20MHz and 261.71MHz respectively. In this case, the critical path of each PE array is the communication between PEs. Clock frequency is inversely proportional to the physical size of PE. Table III shows the throughput and performance gain by clock frequency and the number of processing steps. 8×8Mat/sec is the number of execution of 8×8 matrix multiplications per second.

The number of multiplication steps of PE array using $m \times m$ matrix FMA became $1/m$ times in comparison with PE array using 1×1 matrix FMA. Therefore, we estimate that performance of these circuits will be m times. However, from the results of Table 3, ideal performance gain is decreased gradually by increasing the number of elements. We estimated that the circuit delay and wire length between PEs got bigger by increasing the size of matrix FMA.

V. Conclusion

In this paper, we discussed and evaluated about grain size of the PE of RapidMatriX on an FPGA. PEs of RapidMatrix execute scalar multiply and accumulation. This RapidMatriX can accelerate matrix operation by locating PEs on a torus. On the other hand, recent FPGA has many DSP blocks, high-performance arithmetic units. For circuits design, efficient utilization of DSP blocks is effective. Hereby, implementing functional units for matrix operation to array structure of RapidMatriX, we proposed using DSP block efficiently by increasing the grain size of PEs. We implement RapidMatriX using PEs on an FPGA. In addition, we evaluate the clock frequencies and the clock cycles of calculation. As a results, throughput of PE of 4×4 matrix FMA is 3.14 times in comparison with the original PEs of scalar FMA , in 8×8 matrix multiplication.

FMA	LUT	DSP	Mem	Frequency (MHz)
1×1	430	1	0	422.44
2×2	1,515	4	0	353.48
4×4	5,804	32	0	279.96

Table I
Synthesis results of a single PE

FMA	# of PE	LUT	DSP	Frequency (MHz)
1×1	64	25,554	64	332.89
2×2	16	21,079	64	286.20
4×4	4	20,268	128	261.71

Table II
Synthesis results of PE Array for 8x8 matrix operations

FMA	# of cycle	Throughput (8x8Mat/sec)	performance gain (times)
1×1	8	41,611.25	1
2×2	4	71,550.00	1.72
4×4	2	130,855.00	3.14

Table III
Comparison of proceesing speed (PE Array for 8x8 matrix operations)

References

[1] T. kunihiko, "Proper orthogonal Decomposition in Fluid Flow Analysis," Fundamental Technology Research Center, Honda R&D Co.,Ltd., 14 January, 2011

[2] Chaojiang Fu, "Parallel Computing For Finite Element structural Analysis On Workstation Cluster," International Conference on Computer Science and Software Engineering, 2008

[3] S. Y. Kung, "VLSI Array Processors,"Prentice Hall, 1987

[4] "Intel Itanium Processor," Internet: http://www.intel.com/content/www/us/en/processors/itanium/itanium-processor-9000-sequence.html

[5] M.Gschwind, H.P.Hofstee, B.Flachs, M.Hopins, Y.Watanabe, and T.Yamazaki, "Synergistic Processing in Cell's Multicore Architecture," IEEE Micro, Vol. 26, Issue2, pp.10-24, March-April 2006

[6] Y. Sato, Y. Nomoto, T.Miyazaki, and S. G. Sedukhin, "2-D Array Processor Featuring Ternary Input Fused Operations," IPSJ SIG Tech. Reports, 2007-SLDM-131, pp.7-12

[7] "Digital Signal Processing Blocks in Stratix Series FPGAs," Internet: http://www.altera.com/devices/fpga /stratix-fpgas/about/dsp/stx-dsp-block.html, April. 19, 2010

[8] "Stratix V FPGAs: Built for Bandwidth," Internet: http://www.altera.com/devices/fpga/stratix-fpgas /stratix-v/stxv-index.jsp, Mar. 9, 2012

Authorized licensed use limited to: National Univ of Defense Tech. Downloaded on December 07,2020 at 13:51:02 UTC from IEEE Xplore. Restrictions apply.