

# Real Chip Evaluation of a Low Power CGRA with Optimized Application Mapping

Takuya Kojima, Naoki Ando, Yusuke Matshushita, Hayate Okuhara,  
Ng. Anh Vu Doan and Hideharu Amano  
Keio University,3-14-1 Hiyoshi, Yokohama,223-8522, Japan  
wasmii@am.ics.keio.ac.jp

## ABSTRACT

Even though many optimization methods for CGRAs (Coarse-Grained Reconfigurable Architectures) have been proposed, aggressive power optimization still remains a complex problem to be solved. Moreover, the developments of these methods have mainly been proven on the basis of simulations. Therefore, the questions remains whether they can be applied for a real chip. Here, we consider a real implemented low power CGRA called CCSOTB2, and explore the possibility of the power reduction for this design. This paper proposes to use a metaheuristic method to optimize the power while considering all configurable factors of the CGRA, especially the mapping of an application. This methodology can generate mappings with their related pipeline structure and body bias control automatically. Optimized configurations to use on the real chip are obtained with this methodology and allow to measure the power consumption. The experimental results show a power reduction of 14.2% in average, when compared to a previously-used mapping method which cannot consider body bias and pipeline structure. In addition, the proposed method enables users to select a mapping from various solutions depending on performance requirement and trade-off possibilities (e.g. throughput vs power consumption).

## 1. INTRODUCTION

In the near future, IoT devices, sensor networks and wearable computing are expected to come into general use. Since such devices require high performance and low power consumption simultaneously, general-purpose processors are not suitable. Therefore, there is a necessity to have high energy efficient accelerators to carry out the computational-intensive parts of an application.

CGRAs (Coarse-Grained Reconfigurable Architecture) are an attractive type of platform to cope with these demands. Most of CGRAs have many processing elements (PEs) arranged in a 2-D grid with interconnections between them. A PE consists of a simple ALU, switching elements and distributed local memories. Changing the type of operations and their interconnection can provide reconfigurability and high energy efficiency.

The CMA (Cool Mega Array) architecture has been proposed as a low power CGRA [1]. The basic concept of CMA is to reduce unnecessary power consumption for computing. In order to do that, CMA does not allow dynamic reconfiguration which can consume a large amount of dynamic power. On the other hand, to avoid a decrease of flexibility, it has a tiny micro-controller to enable complicated data transfer between a data memory and the PE array. However, this

can cause a long critical path delay, limiting therefore its performance.

To address these drawbacks, VPCMA (Variable Pipeline Cool Mega Array) has been developed as an improved architecture based on CMA [2]. A PE array of VPCMA has a limited number of configurable pipeline registers. Therefore, its performance and throughput are enhanced while keeping the power overhead of the pipelining at a minimum. Also, VPCMA is designed using 65nm Silicon on Thin Buried Oxide (SOTB) technology, which is a kind of fully depleted silicon on insulator (FDSOI). This allows to control the body bias voltages, giving the possibility to balance the leakage power depending on the required performance.

In general, CGRA compilers are more complicated than those of regular CPUs because they has to consider the routing between PEs as well as assigning operations to PEs. This problem is known as being NP-complete. Therefore, many heuristic techniques which address the mapping problem having proposed [3, 4]. In addition to the general problem, optimization of the pipeline structure and the body bias should be performed to make the most of the features offered by VPCMA. Although an ILP<sup>1</sup>-based method consider both pipeline control and body bias control has been developed in [5], it has only been applied to a static application mapping produced by the Black-Diamond compiler [6] because of the complexity issues. Moreover, most studies, including [5], carry out evaluations based only on simulation results.

In this paper, we consider a real implemented VPCMA chip. Then, real chip experiments are conducted in order to clarify the effectiveness of VPCMA. The applications working on VPCMA are optimized by a genetic-algorithm-based mapping tool which can consider all of the offered possibilities: i) application mapping, ii) pipeline structure and iii) body bias voltages.

The rest of the paper is organized as follows. The next section discusses the background including an overview of VPCMA and related work. Then, an implementation of a real chip VPCMA is described in Section 3. Section 4 introduces an optimization flow. The experiments results are presented in Section 5 and we summarize this paper in Section 6.

## 2. BACKGROUND AND RELATED WORK

Although typical CGRAs support clock-by-clock reconfiguration, this feature consumes a large amount of power. Hence, some CGRAs design follow a static reconfiguration paradigm or a sporadic dynamic reconfiguration policy in order to improve energy efficiency. Such CGRAs can be called Straight Forward CGRAs (SF-CGRAs). A SF-CGRA is composed of a pipelined PE array, data memory and a permutation network. The permutation network is placed

<sup>1</sup>Integer Linear Program

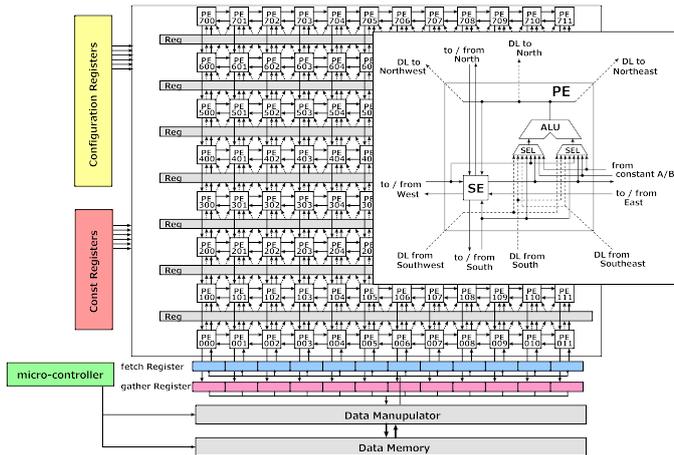


Figure 1: Diagram of VPCMA

between the data memory and the input/output of the PE array for flexible data transfer without dynamic reconfiguration. Input data from the data memory are forwarded to the pipelined array through the network, and then the output data are written back to the memory modules. In most CGRAs, the ALUs and interconnections of each PE can be configured independently with the configuration data. Examples of SF-CGRAs are Pipherench [7], Kilo-core [8], S5 engine [9], EGRA [10].

CMA architecture is also a kind of SF-CGRA with static reconfiguration [1]. The main feature of CMA is that each PE has no register file and solely consists of a combinational circuit. Therefore, a clock input is not required and CMA can reduce any unnecessary dynamic power other than that required for computing. However, CMA has a limitation on the performance improvement due to the use of a huge combinational circuit in its PE array which increases the delay time.

## 2.1 VPCMA Architecture

VPCMA is an improvement over CMA [2]. As shown in Fig. 1, it includes a large PE array, a micro-controller, a data manipulator and a banked data memory. Unlike the original CMA, the PE array of VPCMA has a limited number of pipeline registers between every row. In this work, we consider the  $8 \times 12$  PE array so that there are seven pipeline registers.

After the input data are transferred to the “Fetch register”, the PE array runs automatically. After a few cycles (i.e. the depth of pipeline), the output data are stored to the “Gather register”. Each PE consists of an arithmetic logic unit (ALU), input selectors, and a switching element (SE). There are 2 types of the interconnections between PEs. One is provided by SEs and establishes a 2-D mesh topology network as described with solid lines in Fig. 1. Another is called a direct link which enables a PE to transfer its ALU output to the adjacent PEs directly. As illustrated by dashed lines in Fig. 1, The direct links go to the north, northeast, and northwest directions.

Although the number of available pipeline registers are set to a bare minimum, the overhead of pipelining can still be high. Hence, users can independently define for each pipeline whether it is active or not. In this way, various pipeline structures can be available. The pipeline registers are implemented as illustrated in Fig. 2. Multiplexers select the data from either the registers or output of ALU according to the configuration data. If a pipeline register is set to inactive, the clock input is gated so that the dynamic power consumption is reduced. It is worth noting that no register is placed on the south direction path from the north PE,

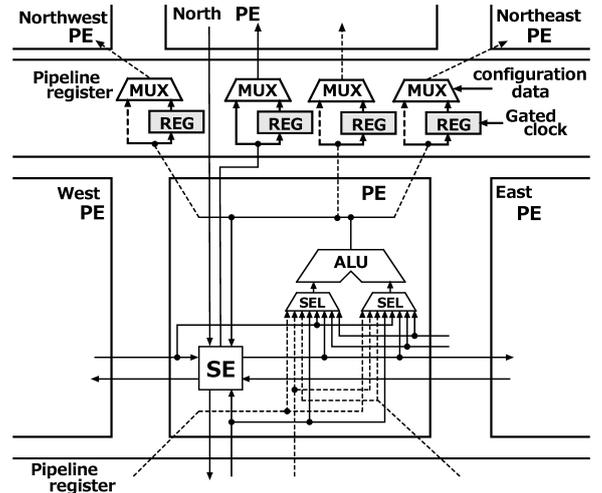


Figure 2: Details of a PE and Pipeline Registers in VPCMA

because it is used to forward computational results.

The “Fetch register” and “Gather register” are respectively connected to the input and the output of PE array. The micro-controller controls the data transfer between these registers and the 12-banked data memory based on micro-instructions. In this paper, the terms “Fetch” and “Gather” are used to refer to the instructions for the transfer to “Fetch register” and from “Gather register” respectively. If a “Fetch” instruction is issued, it is executed immediately. In contrast, a “Gather” instruction is not executed until the computation results are stored to the “Gather register”.

The data manipulator is a permutation network which enables flexible data transfer among the banked memory, Fetch register, and Gather register. It has 12 inputs and 12 outputs, and can send each input to any position of output according to a transfer table. “Fetch” and “Gather” instructions contain an operand to specify the transfer table. In case of “Fetch”, the data read from each memory bank are forwarded to the “Fetch register” through the data manipulator. Thus, VPCMA can deal with various application programs without dynamic reconfiguration.

## 2.2 Body Bias Control on SOTB

SOTB is a type of FD-SOI technology in which transistors are formed on thin buried oxide (BOX) layer. Its structure is shown in Fig. 3. SOTB has the advantage allow a wide control the delay and leakage power consumption with the bias voltage supplied to the body (back-gate) [11]. In Fig. 3, the body-bias voltages of NMOS transistor and PMOS transistor are denoted by  $V_{BN}$  and  $V_{BP}$ , respectively. In case of  $V_{BN} = 0$ , the threshold level of the transistor is normal. Reverse-bias ( $V_{BN} < 0$ ) raises the threshold, that is, the delay is extended while the leakage current is reduced. On the contrary, if forward-bias ( $V_{BN} > 0$ ) is given, the threshold becomes low, and the operational speed is improved with an increase of the leakage current. In order to balance both the NMOS and PMOS, the bias voltage is supplied so that  $V_{BP} + V_{BN} = V_{DD}$  is satisfied. Therefore, in this paper, the level of body bias is hereinafter indicated only by the value of  $V_{BN}$ .

Some research have used body bias control for reconfigurable devices such as FPGAs [12, 13]. This is because the critical path depends on the device configuration so that the delay time difference between the critical path and non-critical one can be big. Using body bias control can redress such an imbalance. However, in most of the studies, the body bias voltage is settled after the configuration is established. Simultaneous optimization of both the body bias and configuration is necessary to reduce the power consumption



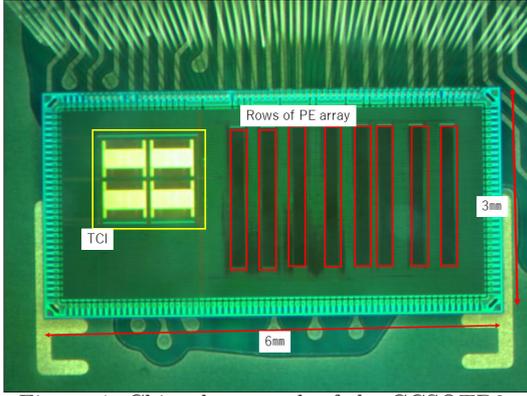


Figure 4: Chip photograph of the CCSOTB2

parts: a list containing the node (ALU) coordinates for each task of the DFG and a 7-bit vector expressing whether a pipeline register is activated. The crossover operation is a 1-point crossover applied separately on each part. The mutation operation for the coordinates list is either a swap or a new random coordinate; for the pipeline structure, it consists of a bit flip. The crossover and mutation probabilities are respectively 0.7 and 0.3 as these are commonly-used values [15]. This allows to explore both the application mapping and the pipeline structure.

As for the body bias control, the ILP introduced in [16, 17] is used, only slightly modified to take the particular body bias domains of CCSOTB2 into account. The choice on bias voltages is carried out so that the constraints on the required performance are met, in particular the critical path of a mapping has to be lower than the maximum allowed value defined by the desired working frequency of the application. The ILP is applied to find the exact optimal voltages for each explored solution.

This optimization flow therefore integrates all the steps needed to deploy a power- and performance-optimized application on CCSOTB2, that is the mapping, the pipeline structure selection, the body bias control, as well as the generation of the configuration bitstream since it also includes the routing of the PEs with respect to the task dependencies.

Although a mapping algorithm using a genetic algorithm for FPGA has already been proposed in [18], it aggregates two factors related to the performance (used resource area and routability between logic blocks) into an objective function with a weighted average, that is, a single objective optimization. Therefore, it cannot prioritize the power consumption over other factors. Thanks to NSGA-II, the proposed method enables users to choose a solution from a wide range of choices depending on various policies or trade-offs.

## 5. REAL CHIP EVALUATION

To evaluate the effectiveness of the optimization explained in Section 4, we carry out real chip experiments.

### 5.1 Experimental setup

First, an evaluation environment is built, as shown in Fig. 6, where a CCSOTB2 board and an FPGA with Artix-7 are attached on a mother board. The FPGA is used to transform test vectors to the CCSOTB2 board. The voltages such as  $VDD$  and  $VBN$ s are produced via power supply pins on the CCSOTB2 board.

Four image processing applications are used for evaluations, as described in Table 3. The application mappings produced by Black-Diamond [6] as well as the proposed method are used for comparison. Black-Diamond can not consider body bias effects and pipeline structure. That is the reason why it always gives a static mapping for each

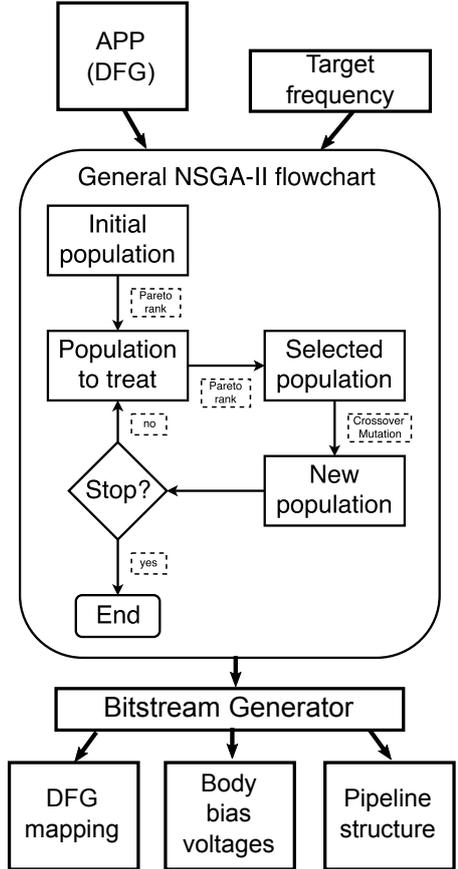


Figure 5: Optimization and generation of bitstream flow

Table 3: Application features

Application	Description
<i>gray</i>	24 bit (RGB) gray scale
<i>sepia</i>	8 bit sepia filter
<i>af</i>	24 bit (RGB) alpha blender
<i>sf</i>	24 bit (RGB) sepia filter

application and performance requirement.

The optimization explained in Section 4 requires several parameters such as the delay time and leakage power of each operation on a PE. These parameters should match real chip measurements as closely as possible. In our case, since it is difficult to measure them on the real chip, only the delay times have to be simulated using Synopsys HSPICE with  $VDD = 0.55$  V, where the body bias voltages  $VBN$  are changed with a step 0.2 V, from -0.8 V to 0.4 V. The leakage power are obtained based on real chip experiments and these results are shown in the next subsection.

In addition, the optimization needs a dynamic power model considering the glitch effects because a post layout simulation requires a relatively long time to evaluate the dynamic power. In this work, the model proposed in [14], which is based on real chip evaluations, is used.

### 5.2 Leakage power

To obtain the leakage power of a PE row, leakage currents of four domains (domain1~4) are measured. When a leakage current of a domain is measured, other domains are supplied with a strong reverse bias such as -2.0 V, and we consider the associated leakage current as negligible.

The measurement results are shown in Fig. 7. Each value is an average for each domain. In the case of domain1, the leakage power is divided by 5 because domain1 has 5 PE rows. Besides, we can observe an exponential increase of the leakage power with the body bias voltage ( $VBN$ ).

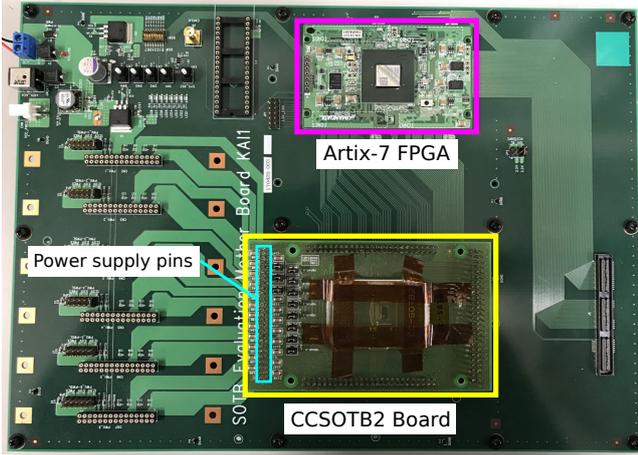


Figure 6: Evaluation Environment

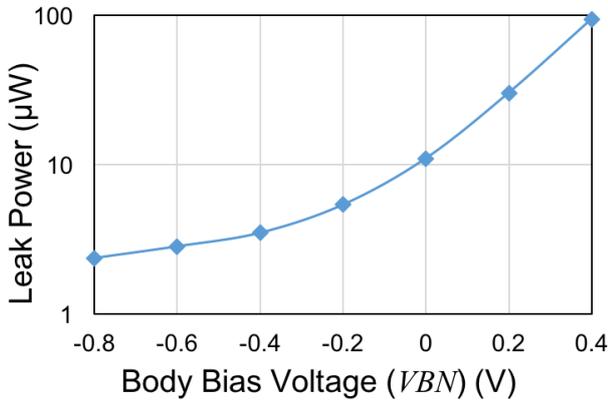


Figure 7: Measurement results of leakage power per PE row

### 5.3 Operating frequency

As results of the experiments, the micro-controller of CC-SOTB2 can operate at 30 MHz with  $V_{DD} = 0.55$  V. At this frequency, not all pipeline registers of the PE array are needed to be activated. In other words, the micro-controller is the performance bottleneck rather than the PE array. Hence, 30MHz is used for the target frequency of the optimization, which will act as a constraint in the optimization process of the mapping and select the bias voltages accordingly.

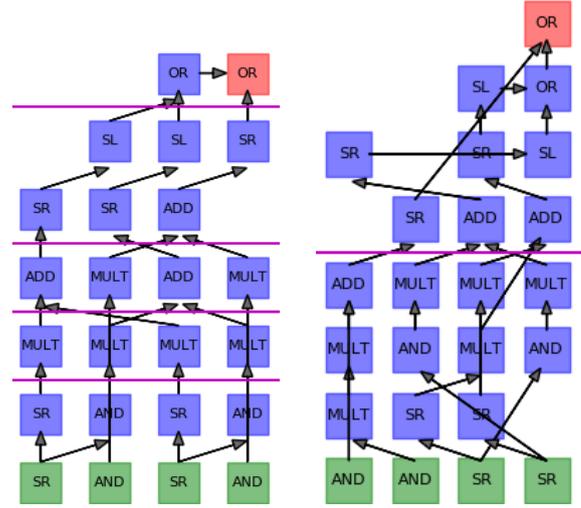
### 5.4 Optimization Results

To evaluate the effectiveness of the proposed method, we tested the proposed mapping tool for each application and measured the total power consumption of the PE array. As a comparison basis, we measure the power of the Black-Diamond mapping with the pipeline optimization proposed in [14] as well.

#### 5.4.1 Mapping Quality

Fig. 8(a) and (b) illustrate the difference in the mapping results of Black-Diamond and our new method, respectively. When using Black-Diamond, a programmer has to specify the position of PE for each operation manually. *af* can be mapped into seven rows of PE so that the mapping of Black-Diamond employs seven rows. However, the DFG of *af* has long delay operations such as the addition (“ADD”) and multiplication (“MULT”) in the middle of the mapping. Therefore, the middle rows of PE array are responsible for the critical path and more pipeline registers are needed to be activated. Furthermore, to prevent glitch propagation, more pipeline registers are likely to be used.

On the other hand, the mapping generated by the pro-



(a) Black-Diamond (b) Proposed method  
Figure 8: Difference in mapping results (*af*)

Table 4: Optimal Body bias voltages for each domain

	Domains			
	1 (1-5th rows)	2 (6th row)	3 (7th row)	4 (8th row)
<i>af</i>	0.0 V	0.0 V	0.0 V	-0.2 V
<i>gray</i>	0.0 V	-0.4 V	-0.4 V	-0.4 V
<i>sf</i>	0.0 V	-0.6 V	-0.2 V	-0.2 V
<i>sepia</i>	0.0 V	-0.8 V	-0.8 V	0.0 V

posed method uses eight rows of PE. As a result, the middle rows can be sparser and only one pipeline register is activated. In addition, the proposed method generates a mapping while considering the glitch effects. Thus, in spite of only one pipeline register being activated, the glitch propagation is restricted as well.

The proposed method produces the body bias assignment as described in Table 4. It is evident from the results that optimal body bias voltage depends on the application mapping. In case of the above *af* mapping, reverse bias is used only for domain4. Instead of reverse bias, the upper pipeline registers are deactivated so that the related dynamic power is saved. On the contrary, an optimal mapping for *gray* has three pipeline stages, that is, two pipeline registers are used. In the last pipeline stage (6-8th rows), only shift operations (e.g. shift-right “SR”) and logic operations (e.g. bit-wise OR “OR”), which have a shorter delay than arithmetic operations (e.g. addition “ADD”), are mapped. Thereby, reverse bias can be used for domain2,3 and 4. In this way, the proposed method can also provide a mapping considering the body bias effects and the pipeline structure.

As explained in Section 4, the proposed method requires application DFG as input. Hence, it carries out not only the routing among PEs but also the operation assignment for each PE automatically. Besides, its estimation of delay time is accurate according to the real chip experiments.

#### 5.4.2 Power

Fig. 9 shows the results of the power consumption of the PE array when the above body bias voltages are applied, alongside the power reduction ratio. In average, the power consumption is 14.2% lower than with the mapping of Black-Diamond. In the best case, 16.7% power reduction is can be achieved (with *af*). Although the power consumption of the micro-controller is not included, it is around 0.5mW for each application. When executing *af* at 30 MHz, a performance of 2160 MOPS (Million Operations Per Second) can be achieved. Therefore, the evaluated energy efficiency is

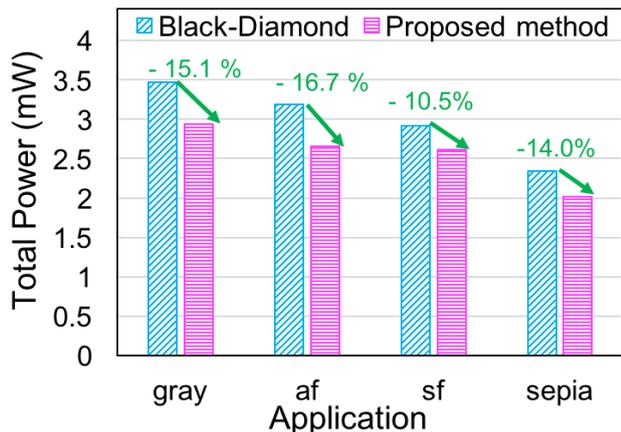


Figure 9: Comparison of the total power

680 MOPS/mW, considering the power consumption of the whole chip.

### 5.4.3 Trade-off between Throughput and Power

To maximize the data-level parallelism of an application, we assume that the mapping with the minimum width is selected to achieve the highest throughput. For example, *af* needs at least four columns of PEs, as shown in Fig. 8. So, the same mapping can be duplicated twice to the remaining eight columns. However, having the maximum throughput is not always necessary. Since the proposed method performs a design space exploration using a multi-objective optimization paradigm, it can also provide solutions that show trade-off information. Thus, mappings with other throughput values are also available.

For example, in the case of *gray*, the width of mapping used for the aforementioned power measurements is two, so, it can be duplicated 5 times to the PE array. In comparison, if a mapping with a width of 4 is used, the measures power consumption is 2.346 mW, which is 20.3 % lower than the mapping with a width of 2. These results show the trade-off possibilities offered by our proposed tool and the multi-objective optimization paradigm.

## 6. CONCLUSION

In this paper, we have proposed an optimization method for VPCMA which is an energy-aware CGRAs. In general, application mapping to the PE array is one of the most difficult problem. Furthermore, VPCMA has possibilities of power optimization by controlling its pipeline structure and body bias voltages. Therefore, we chose to use a genetic algorithm due to the complexity this optimization problem.

The proposed method can generate better mappings, pipeline structures and body bias assignments than that of the previously-used tool Black-Diamond, just by giving the data flow graph of an application and the target frequency. As the experimental results have shown, the obtained configuration can achieve in average 14.2% lower power consumption, when compared to the previous method with only pipeline optimization. In addition, the method provides various solutions thanks to the use of a multi-objective optimization methodology. Thus, we can select a solution from the generated set depending on trade-off possibilities.

## Acknowledgement

This work is supported by JSPS KAKENHI S Grant Number 25220002. This work is supported by VLSI Design and Education Center(VDEC), the University of Tokyo in collaboration with Synopsys, Inc and Cadence Design Systems, Inc.

## 7. REFERENCES

- [1] Nobuaki Ozaki, Yoshihiro Yasuda, Mai Izawa, Yoshiaki Saito, Daisuke Ikebuchi, Hideharu Amano, Hiroshi Nakamura, Kimiyoshi Usami, Mitaro Namiki, and Masaaki Kondo. Cool Mega-Arrays: Ultralow-Power Reconfigurable Accelerator Chips. *IEEE Micro*, Vol. 31, No. 6, pp. 6–18, Nov 2011.
- [2] Naoki Ando, Koichiro Masuyama, Hayate Okuhara, and Hideharu Amano. Variable Pipeline Structure for Coarse Grained Reconfigurable Array CMA. In *2016 International Conference on Field-Programmable Technology*, pp. 231–238, 2016.
- [3] Mahdi Hamzeh, Aviral Shrivastava, and Sarma Vrudhula. EPIMap: Using epimorphism to map applications on CGRAs. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pp. 1280–1287. IEEE, 2012.
- [4] Jiangyuan Gu, Shouyi Yin, Leibo Liu, and Shaojun Wei. Energy-aware loops mapping on multi-vdd CGRAs without performance degradation. In *Design Automation Conference (ASP-DAC), 2017 22nd Asia and South Pacific*, pp. 312–317. IEEE, 2017.
- [5] Takuya Kojima, Naoki Ando, Hayate Okuhara, Ng Anh Vu Doan, and Hideharu Amano. Body bias optimization for variable pipelined CGRA. In *Field Programmable Logic and Applications (FPL), 2017 27th International Conference on*, pp. 1–4. IEEE, 2017.
- [6] Vasutan Tunbunheng and Hideharu Amano. Black-diamond: a retargetable compiler using graph with configuration bits for dynamically reconfigurable architectures. In *Proc. of The 14th SASIMI*, pp. 412–419, 2007.
- [7] Herman Schmit, David Whelihan, Andrew Tsai, Matthew Moe, Benjamin Levine, and R Reed Taylor. PIPERENCH: A virtualized programmable datapath in 0.18 micron technology. In *Custom Integrated Circuits Conference, 2002. Proceedings of the IEEE 2002*, pp. 63–66. IEEE, 2002.
- [8] Benjamin Levine. Kilocore: Scalable, High Performance and Power Efficient Coarse Grained Reconfigurable Fabrics. In *Proc. of International Symposium on Advanced Reconfigurable Systems*, pp. 129–158, 2005.
- [9] Jeffrey M Arnold. S5: The Architecture and Development Flow of a Software Configurable Processor. In *Proc. of the 4th IEEE Int'l Conf. on Field Programmable Technology (ICFPT2005)*, pp. 120–128, December 2005.
- [10] Giovanni Ansaloni, Paolo Bonzini, and Laura Pozzi. Egra: A coarse grained reconfigurable architectural template. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 19, No. 6, pp. 1062–1074, 2011.
- [11] Ishigaki, Takashi and Tsuchiya, Ryuta and Morita, Yusuke and Sugii, Nobuyuki and Kimura, Shin'ichiro. Ultralow-power LSI Technology with Silicon on Thin Buried Oxide (SOTB) CMOSFET. *Solid State Circuits Technologies, Jacobus W. Swart (Ed.), ISBN: 978-953-307-045-2, InTech*, pp. 146–156, 2010.
- [12] Masakazu Hioki and Hanpei Koike. Low Overhead Design of Power Reconfigurable FPGA with Fine-Grained Body Biasing on 65-nm SOTB CMOS Technology. *IEICE TRANSACTIONS on Information and Systems*, Vol. 99, No. 12, pp. 3082–3089, 2016.
- [13] Lewis, David and Ahmed, Elias and Cashman, David and Vanderhoek, Tim and Lane, Chris and Lee, Andy and Pan, Philip. Architectural enhancements in stratix-iii<sup>TM</sup> and stratix-iv<sup>TM</sup>. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pp. 33–42. ACM, 2009.
- [14] Takuya Kojima, Naoki Ando, Hayate Okuhara, and Hideharu Amano. Glitch-aware variable pipeline optimization for CGRAs. In *2017 International Conference on ReConfigurable Computing and FPGAs (ReConFig)*, pp. 1–6, Dec 2017.
- [15] Lawrence Davis. Adapting operator probabilities in genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms*, pp. 61–69, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.
- [16] N.A.V. Doan, Y. Matsushita, N. Ando, H. Okuhara, and H. Amano. Multi-objective optimization for application mapping and body bias control on a CGRA. In *2017 IEEE 11th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSOC)*, pp. 143–150, Sept 2017.
- [17] T. Kojima, N. Ando, H. Okuhara, N.A.V. Doan, and H. Amano. Optimization of body biasing for variable pipelined coarse-grained reconfigurable architectures. *IEICE Transactions on Information and Systems*, Vol. E101-D, No. 6, June 2018.
- [18] S. N. R. Borra, A. Muthukaruppan, S. Suresh, and V. Kamakoti. A parallel genetic approach to the placement problem for field programmable gate arrays. In *Proceedings International Parallel and Distributed Processing Symposium*, April 2003.