A 297MOPS/0.4mW Ultra Low Power Coarse-grained Reconfigurable Accelerator CMA-SOTB-2

Koichiro Masuyama, Yu Fujita, Hayate Okuhara, Hideharu Amano
Dept. of ICS, Keio University, Yokohama Japan
Email: {wasmii, leap}@am.ics.keio.ac.jp

Abstract—**Cool mega array-SOTB-2 (CMA-SOTB-2) is an ultra-low energy coarse grained reconfigurable architecture (CGRA) for advanced sensor networks, the Internet of Things, and wearable computing. It uses a large processing element (PE) array with combinatorial circuits and a micro-controller for data transfer between data memory and the PE array. To improve the energy efficiency of the previous prototype, the CMA-SOTB, the performance of the micro-controller was improved by introducing parallel data memory access with data manipulators and optimization of both instruction sets and micro-architecture. A delay learning mechanism that finds the optimal delay time for the computation in the PE array is also introduced. Standard cell libraries of the 65nm silicon on thin buried oxide (SOTB) process have been optimized for under-milliwatt operation. A real chip evaluation shows that more than 250-MOPS performance was achieved with only a 0.4-mW power budget by independently controlling the body-bias voltage for the micro-controller and the PE array. The energy efficiency is almost double that of the previous prototype, the CMA-SOTB.**

I. Introduction

Recent advanced sensor networks, the Internet of Things, and wearable computing require a certain degree of computing performance with ultra-low energy consumption. Accelerators that are conventionally used for high performance computation have received attention as a method to achieve the required performance with extremely low power consumption instead of tiny micro-controllers working with an operational clock in the hundreds of kilo Hertz. Coarse grained reconfigurable arrays (CGRAs) are possible candidates for such an accelerator with extremely low power computation[1][2]. They are suitable for image processing or signal processing with integer or fixed-point computation, which are common calculations in such application fields, and their energy efficiency is much higher than that of other accelerator solutions like GPU or many-core processors.

As such an accelerator candidate, Cool Mega Array (CMA)[3], a type of CGRA, has been proposed, and some prototypes have been implemented and evaluated. CMA provides a large processing element (PE) array consisting of combinatorial logic and a small micro-controller that manages data distribution and collection between data memory and the input/output of the PE array. The third generation CMA, CMA-SOTB-2 optimizes the performance and energy balance between memory accesses and the computation in the PE array based on the experience of previous prototypes. A novel silicon on insulator (SOI) technology called silicon on thin buried oxide (SOTB) which can control the performance and leakage current by body biasing investigated with the previous prototype was introduced. A parallel data transfer mechanism is introduced to enhance the memory access performance while the training register is newly provided to decide the data acquisition timing of the PE array. As a result, 297 MOPS sustained performance was achieved for a power consumption of only 0.4 mW. It is remarkable since the practical performance was achieved with extremely low power as well as its high degree of overall performance per power (742.5 MOPS/mW). It can achieve an image processing with practical speed just by a lemon battery.

The contribution of the paper is as follows:

- A data manipulator that allows parallel data reading and writing is proposed for improving the performance of the micro-controller.
- An automatic delay detection mechanism is proposed and implemented to decide the data acquisition timing.
- The standard cells and body biasing were optimized for under-milliwatt operation.
- A 297 MOPS performance was achieved with only 0.4 mW power consumption.

The rest of the paper is organized as follows: The concept of CMA and previous prototypes are described in Section 2. Section 3 describes CMA-SOTB-2 which we newly implemented this time. Section 4 discusses its evaluation with a real chip and compares its performance with that of conventional CMA-SOTB. The key points are summarized and future work is mentioned in Section 5.

II. CMA-SOTB

A. The CMA architecture

The CMA architecture[3] was optimized for near-threshold computing in which its supply voltage cannot be lowered any further. It executes a fixed number of computations in the required time with the minimum amount of energy. A key concept of the CMA architecture is reducing any energy usage other than that required for computation. The PE array is built with combinatorial circuits to eliminate the power needed to store the intermediate results in registers and to distribute a clock to all PEs. The dataflow graph of the target application is directly mapped on the PE array. Registers are only provided at the inputs/outputs of the PE array. Computation starts when all data are set up in the input "launch register," and the outputs of the PE array are stored into the "result register" with a

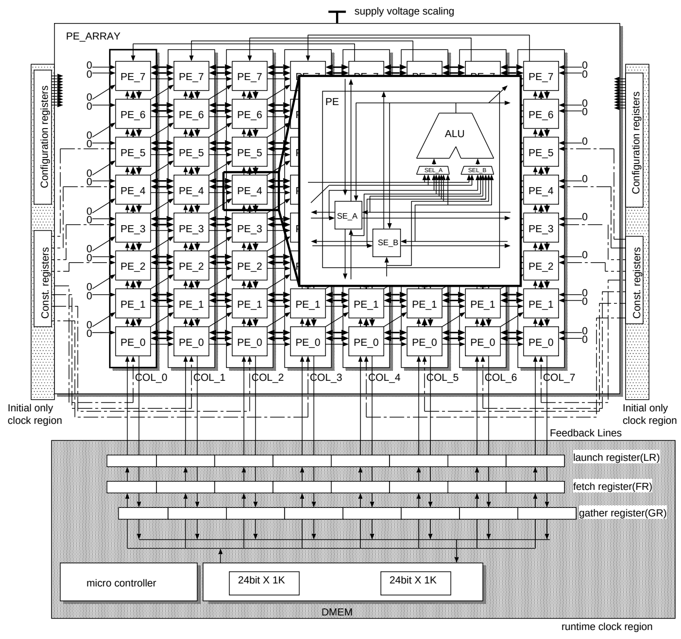


Fig. 1. Block diagram of CMA

certain delay time. The energy overhead caused by glitches in the large combinatorial circuits can be reduced by carefully setting the configuration data of switching element so as not to propagate glitches[4].

A micro-controller reads the data from the data memory (DMEM) and distributes to fields of the launch register attached to the input of the PE array. It also writes computation results in the results register attached to the output of the PE array into the data memory. It flexibly manages the data transfer between the memory and registers by using mapping registers and vector operations. The aforementioned structure enables the implementation of various application programs without a power hungry dynamic reconfiguration in the PE array.

Another key concept of the CMA architecture is optimizing the energy of each target application by balancing the performance of the PE array and micro-controller. For application with a high degree of arithmetic intensity, the performance of the PE array is enhanced by using a power budget, while the power of the micro-controller is lowered. However, when the application requires a lot of data sets for a computation, the power budget is used for the micro-controller that manages the data transfer between data memory and launch/result registers.

B. Previous prototypes

The first prototype, CMA-1[3][4] was designed and implemented with Fujitsu's 65nm process.

Figure 1 shows the block diagram of the CMA-1, the first prototype CMA architecture. A PE consists of a simple 24-bit ALU that executes multiply, add, subtract, shift, and logic operations, and a switching element (SE). Like other CGRAs, the operation of the PE and SE is selected by the configuration data, which is given from the outside configuration registers. It has an 8×8 PE array connected with a network using a two-channel island-style interconnection and direct links that connect to the north-east and east of the PE. The SEs transfer the input data from the PE in the south, west, and east of the PE and the output data of the ALU to the PE in the appropriate direction according to the configuration data.

The micro-controller is a tiny microprocessor that executes a 14-bit micro-code stored in 128-entry micro-memory. It has 16 general purpose registers and 8 special purpose registers storing pointers of data memory, bit-map vectors, and stride lengths for a stride data transfer. It reads eight data from the data memory and sets the launch register with a single instruction. A dedicated memory controller triggered with the instruction executes the data transfer with eight clock cycles. Also, the data in the result register can be written back to the data memory with a single instruction handled by another controller. Because the data memory is a single-read/single-write dual-port memory, the reading and writing data can be done in parallel. In CMA-1, two banks of 1K-entry 24-bit dual-port memory are provided for overlap operation of streaming data input/output and computation. The micro-controller executes special micro-codes for setting registers and for controlling the loop with a single single operation. The performance balance between the micro-controller and PE array was achieved by changing the supply voltage independently. By careful optimization of the configuration data in PE array and introducing wave-pipelining, 2.72GOPS of sustained performance was achieved with 11.2mW (233MOS/mW) power budget.

In order to run CMA with further low supply voltage, a novel fully depleted SOI CMOS technology Silicon on thin buried oxide (SOTB)[5][6] was introduced, and CMA-SOTB was developed[7]. In CMA-SOTB, the balance of the PE array and micro-controller was controlled by the body biasing for each module instead of the power supply voltage. Although the optimization technique using bias-voltage was investigated on a real CMA-SOTB chip, the prototype was not successful from the viewpoint of low power processing. Since the same architecture for the normal bulk process was directly used in SOTB technology, the performance of PE array was much larger than that of the micro-controller which cannot be compensated by the bias voltage control. Also, the threshold level of experimental standard cells of SOTB was relatively low resulting a large leakage power even with the zero body bias voltage. As a result, CMA-SOTB only achieved 192MOPS/mW sustained performance that was smaller than that of CMA-1. Based on the experience of CMA-SOTB, we designed and implemented CMA-SOTB-2, the third generation prototype of CMA.

C. Data-flow graph mapping

As with other CGRAs, the operation of each PE and the interconnections between them are controlled by configuration data prepared before execution. In CMA, configuration registers are located outside the PE array (see Figure 1), and all configuration signals are sent from them. The delay caused by long wires from outside the PE array doesn ' t degrade operational speed, because the configuration data aren ' t changed during execution. We use the configuration multicast method RoMultiC[8] for loading the configuration data. We assign a 2D bitmap to each PE, and we multicast the configuration data to the configuration registers that have column and row bits set to 1.

To get the configuration data, we use the Black Diamond compiler[9], which uses a C-like programming language. It compiles the described program and generates the configuration data for the PE array.

III. CMA-SOTB-2

1) SOTB technology: Silicon on thin buried oxide (SOTB)[5][6] is a novel fully depleted SOI CMOS technology developed by the low-power electronics association and project (LEAP). In SOTB, transistors are formed on a thin buried oxide layer, as shown in Figure 2.

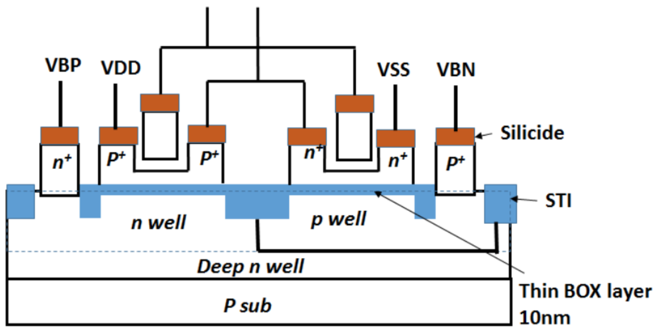


Fig. 2. Cross-sectional view of the SOTB device

By using an ultra-thin FD-SOI layer and the BOX layer, we can suppress the detrimental short channel effect (SCE) in the SOTB. Because impurity doping (a halo implant) to the channel is not necessary, the variation in the threshold voltage by the RDF can be reduced. A multi-threshold voltage design is easily available by doping an impurity into the substrate directly under the thin BOX layer. Thus, we can extensively control the range of body (back-gate) bias and optimize the performance and power consumption after fabrication. The detail of SOTB technology and other benefits for low power computation are shown in paper[5][6].

An SOTB transistor has a back-gate bias contact provided to its well. For an NMOS transistor, VBN is given to its p-well. Here, zero bias (VBN=0) means the transistor works with its normal threshold. When the reverse bias (VBN of negative value) is given, the threshold is increased. Thus, the leakage current is reduced, but the delay is stretched. However, with the forward bias (VBN of positive value), the threshold is decreased. Thus, the leakage current is increased, but the operational speed is enhanced. For a PMOS transistor, VBP is given to its n-well, and zero bias corresponds to the supply voltage, that is, $VBP = VDD$. Reverse bias means a VBP of larger than VDD, and the forward bias represents the case where a VBP of smaller than VDD is given.

In both CMA-SOTB and CMA-SOTB-2, independent body bias is given to the PE array and micro-controller/data memory. Here, bias voltages for the PE array are referred to as $VPNC$ and $VPBC$, and those for the micro-controller are $VPNM$ and $VPPM$. By controlling the body bias separately, we can optimize the energy consumption while keeping the required performance. For a target application with strong arithmetic intensity, the PE array is given a forward bias ($VBNC > 0$, $VBPC < VDD$) while the micro-controller/data memory is given a reverse bias ($VBNM < 0$, $VBPM > VDD$). In contrast, if the data transfer has a bottleneck, the forward bias is given to the micro-controller/data memory, and the reverse bias is given to the PE array. A method for controlling the body biasing was proposed in our previous study[7].

A. Improving the micro-controller performance

1) Data manipulator: The micro-controller used in CMA-1 transferred a data-item from/to the memory with a clock cycle. It means that at least eight clock cycles are required to load and store data to and from input and output of the PE array. Considering the large delay time of PE array, eight clock cycles latency of accessing data memory was balanced for CMA-1 in which the micro-controller worked with high operational clock. It contributed to reduce the power for accessing the memory and improved the flexibility of memory accesses. Stride memory accesses and indexed memory accesses were executed with a single instruction. However, in the CMA-SOTB, a large forward bias is required for high speed operation of the micro-controller, resulted to degrade the total energy performance.

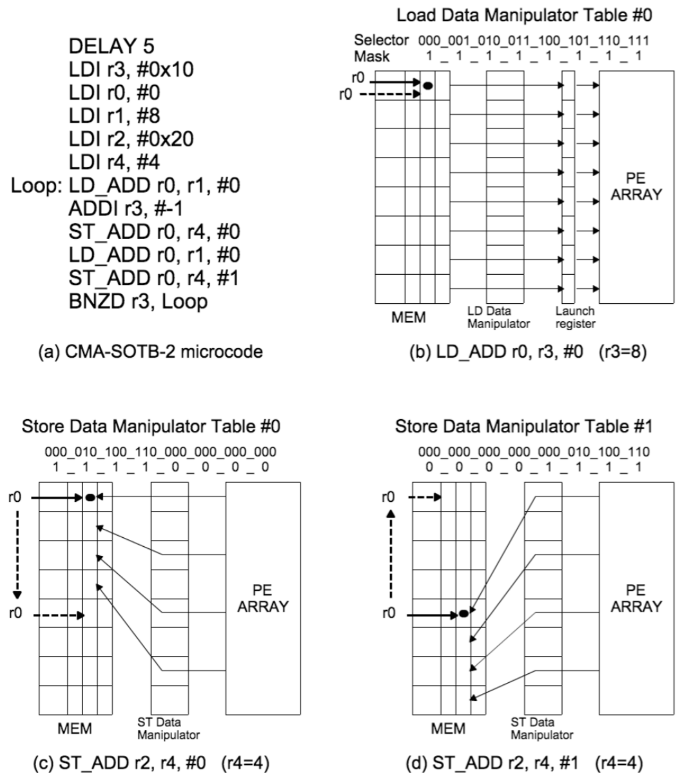
Thus, in CMA-SOTB2, parallel data accessing from/to multiple memory banks must be provided like other powerful accelerators. We introduced a simple data manipulator in order to reduce the power consumption as possible. It is consisting of a set of multiplexers that can send the data from an arbitrary output of the memory bank to the field of the launch register only one clock cycles. The input selection of each multiplexer is defined by the data manipulator control table. It includes the input selection control data of each multiplexer and the bit-map indicating which output of the multiplexer should be written into the field of the launch register.

A dedicated instruction “LD_ADD” has three operands: a register for the address pointer, a register for the stride length, and a literal for the entry number of the manipulator control table. For example, when the following instruction is executed, eight data are read out from eight continuous addresses pointed by r0 in parallel and are sent to the field of the launch register according to the data in entry 0 of the data manipulator control table, as shown in Figure 3.

```
LD_ADD r0, r3, #0
```

Then, r0 is incremented by the number held in r3. Figure 3(b) is the simplest example used in the alpha blender shown in Figure 3 (a), and eight data from each bank are directly forwarded to each field of the launch register. Note that the start address represented by r0 can be set without restriction, and the target field is selectable by the bit map in the control table. If we have to read irregularly aligned data for a single launch, we must use LD_ADD multiple times until every required field of the launch register is filled with the input data. The computation in the PE array automatically starts when the launch register is ready.

The result data from the PE array are written into the DMEM directly by the instruction “ST_ADD,” which has the



```
      DELAY 5
      LDI r3, #0x10
      LDI r0, #0
      LDI r1, #8
      LDI r2, #0x20
      LDI r4, #4
Loop: LD_ADD r0, r1, #0
      ADDI r3, #-1
      ST_ADD r0, r4, #0
      LD_ADD r0, r1, #0
      ST_ADD r0, r4, #1
      BNZD r3, Loop
```

Fig. 3. Examples of data manipulator

same format as "LD_ADD" with a pre-defined delay time. Another data manipulator is provided at the output of the PE array, and the data are written back into the eight-interleaved memory modules according to the data manipulator control table. Note that the dual-port memory is used for the DMEM, and the reading and writing data can be executed in parallel. Figure 3(c) and (d) show the example. Four results computed with the PE array in parallel are stored in the continuous address of the DMEM. Two entries, each of which has a different selector and mask, are used for storing four data into different banks. Like the "LD_ADD" instruction, if the target addresses of the DMEM are not regular, multiple instructions must be executed.

2) Finding the delay cycles: The delay time corresponding to the computation time in the PE array depends on the data flow graph mapped on the PE array. The results from the PE array are written back to the DMEM after the delay cycles from executing the "LD_ADD" instruction. Finding appropriate delay cycles for a given application has been a challenge in the CMA architecture. The safe method is summing up the maximum number of cascade connected PEs and corresponding delay of the operations[4]. However, it includes a large margin that might degrade the performance. Although methods using delay line[10] and automatic delay adjusting methods[11] were proposed, they are difficult to implement.

We designed a simple method by using the training register. As shown in Figure 4, the output of the PE is stored in the training register with a delay specified by a training counter before real data acquisition. When data are written into the memory, the content of the training register is compared with the result data. If they match, the training counter is decremented, and the storing timing of the training register becomes earlier in the next trial. Otherwise, the training counter is incremented, and it is never decremented again during the execution. After executing an application, the training counter keeps the largest delay cycles required in the application. When the same application is executed again, we can use the delay cycles in the training counter as the pre-defined delay cycles. The evaluation in this paper is done using this optimization.

TABLE I. SPECIFICATIONS OF CMA-SOTB-2

CMA-SOTB-2	PE array	8×8
	DMEM	24 bits $\times$ 256 words
	μ-controller	16 bits $\times$ 128 words
	Library	LPT-8
Chip	Process	LEAP 65nm SOTB 7-metal
	Size	5mm $\times$ 5mm
	I/O	208 pins
Tools	Design	Verilog HDL
	Synthesis	Synopsys Design Compiler 2011.09-SP2
	P&R	Synopsys IC Compiler 2010.12-SP5

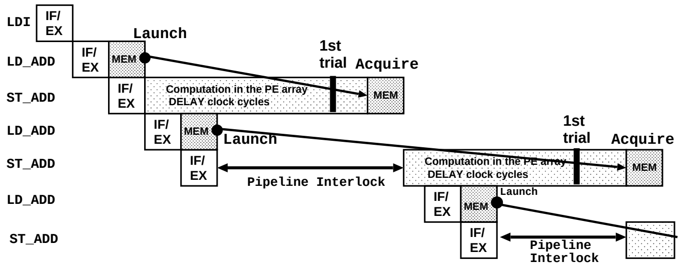


Fig. 4. CMA-SOTB-2 pipeline execution and operation of training counter

B. Chip implementation

Table I shows the specifications CMA-SOTB-2. In the $5mm \times 5mm$ chip, almost half of the area was used for both architectures because the SOTB is still in the experimental process and because the choice of chip size was limited.

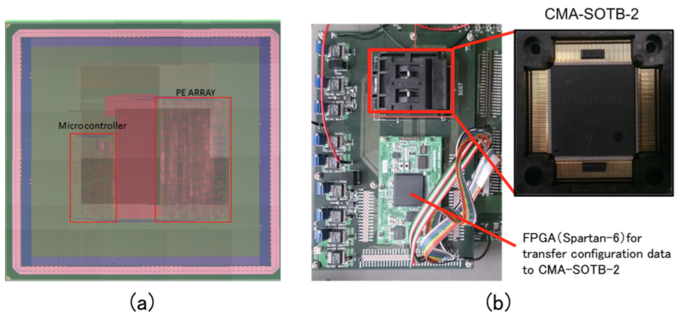

PE ARRAY
Microcontroller
(a)
CMA-SOTB-2
FPGA(Spartan-6)for transfer configuration data to CMA-SOTB-2
(b)

Fig. 5. (a) Chip photograph of CMA-SOTB-2 and (b) Testing environment

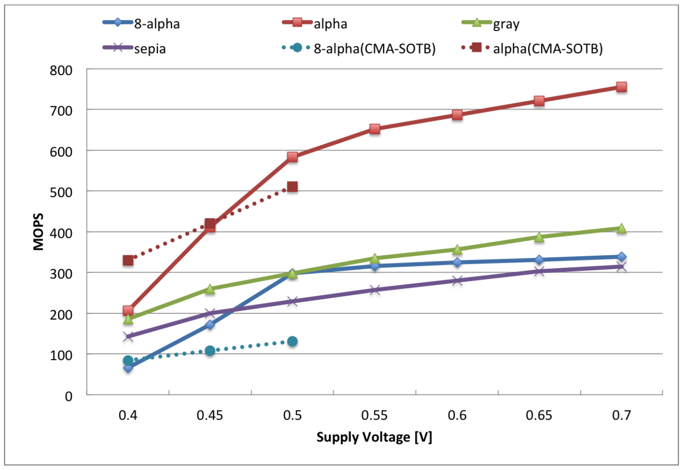


Fig. 6. Performance vs. supply voltage

IV. Real Chip Evaluation

In this evaluation, four simple image processing programs, 8-alpha, alpha, gray, and sepia, were used. In alpha, gray, and sepia, the RGB brightness was compressed into a 24-bit word, while an 8-bit word was used in 8-alpha.

TABLE II. Application programs

8-alpha	8-bit alpha blender
alpha	24-bit (RGB) alpha blender
gray	24-bit (RGB) gray scale
sepia	24-bit (RGB) sepia filter

The testing board shown in Figure 5(b) was used for evaluation. The test vectors generated in the FPGA were inserted to CMA-SOTB-2 and executed results were returned and checked in the same FPGA.

A. Performance comparison

The performance of the four application programs is shown in Figure 6. Here, million operation per second (MOPS) was used as the measure of performance. Note that only the computations executed on the PE array were evaluated, and operations for control and reading/writing data in the micro-controller were not included. That is, the performance was proportional to the utilization of PEs and the operating frequency of the micro-controller. The performance of the CMA-SOTB-2 increased rapidly from 0.4 to 0.5 V. From 0.5 to 0.7 V, the slope became gentle, but the performance still increased linearly. As a result, it achieved 206 MOPS at 0.4 V and 754 MOPS at 0.7 V in *alpha*.

Because of the improvement in the micro-controller, the CMA-SOTB-2 achieved larger MOPS than that of the CMA-SOTB with more than 0.45-V supply voltage. At 0.5 V, the performance of the CMA-SOTB-2 was about twice that of the CMA-SOTB in *8-alpha* and about 18% larger in *alpha*. The difference comes from the fact that the utilization of the PE in *alpha* was much larger (48) than that in *8-alpha* (16). Because the optimization for the micro-controller mainly improves the data transfer between the PE array and MEM, it is not very efficient for programs with high arithmetic intensity, where the computation in the PE array tends to cause a bottleneck in the performance. The training register can slightly improve the performance of the PE array. This is the reason for the CMA-SOTB-2 achieving better performance for high arithmetic intensity programs.

With a 0.4-V power supply, the performance of *alpha* with the CMA-SOTB-2 was less than that with the CMA-SOTB. This comes from the difference in the cell library used in the CMA-SOTB and CMA-SOTB-2. The threshold level of the cell library used in the CMA-SOTB was lower than those used in the CMA-SOTB-2 and was optimized for high performance operation with a low supply voltage at the expense of a large leakage power, shown later.

B. Energy efficiency

The energy efficiency (MOPS/mW) of 4 application programs is shown in Figure 7 and Figure 8. The solid lines show the MOPS/mW of CMA-SOTB-2, and the dashed lines show that of previous prototype, CMA-SOTB. That is, from 100 to 500 MOPS were achieved at 0.450.5 V supply voltage, reaching the maximum energy efficiency. Figure 7 shows that 470 MOPS/mW (260 MOPS/0.55 mW) sustained performance was achieved at a supply voltage of 0.45 V, when *gray* was executed. Compared with the CMA-SOTB, the maximum energy efficiency was more than triple both for *8-alpha* and *alpha*.

The energy efficiency can be improved without degrading the performance by optimizing the bias voltage. The optimization policy proposed for CMA-SOTB[7] was used.That is, the supply voltage was fixed first, then the bias for the PE array and micro-controller was optimized according to the arithmetic intensity of application programs. The results of the optimization for 4 application programs are shown in Figure 8. An improvement from 30% to 200% energy efficiency was achieved due to the optimization. As a result, 743 MOPS/mW (297 MOPS/0.4 mW) sustained performance was achieved for *gray* at the 0.5 V supply voltage. Although a processor[12] and an FPGA[13] with larger energy efficiency were reported, CMA-SOTB-2 is remarkable since it can achieve a high degree of performance with an extremely small power budget. It is especially useful when the supply energy budget is limited or an energy harvesting battery with a large internal resistance is used.

C. Power consumption

Figure 9 shows the breakdown in the power consumption at 0.4 V and 0.5 V of *alpha* with a zero body bias (zero bias) and the optimized bias (bias). The frequencies were 30 MHz at 0.4V and 85 MHz at 0.5 V. An optimized bias was found by varying both the bias for the PE array and micro-controller/memory. Because of the body-bias control, the leakage power was greatly reduced. As a result, by body biasing, the power of the CMA-SOTB-2 was reduced by 18% at 0.4 V and 6% at 0.5 V.

V. Conclusion

CMA-SOTB-2, an improved version of the CMA-SOTB, was designed, implemented, and evaluated. By allowing parallel memory access by using data manipulators, the data transfer

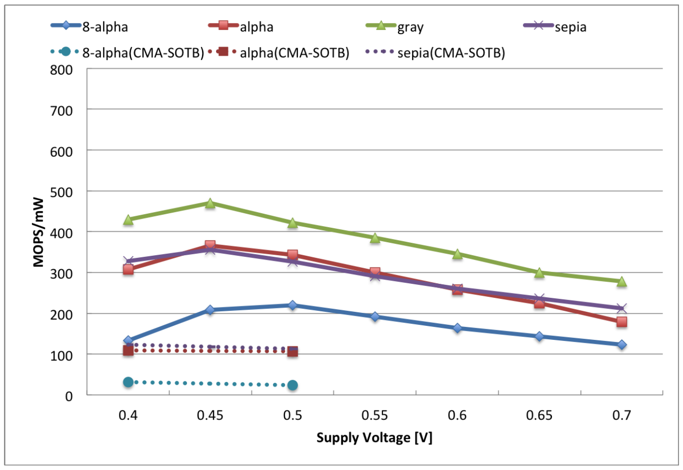


Fig. 7. Energy efficiency without body bias

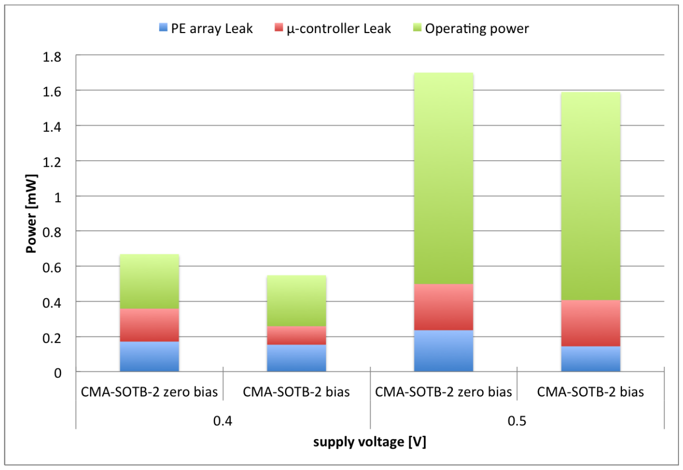


Fig. 9. Breakdown of power consumption

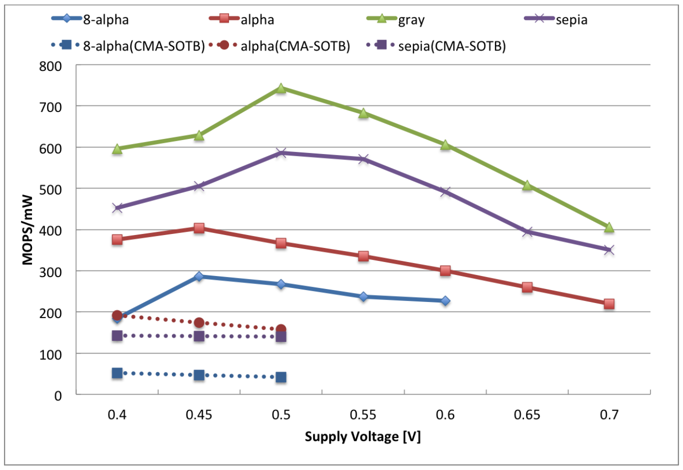


Fig. 8. Energy efficiency with the optimized body bias

speed of the micro-controller can be improved. A training register was introduced to find the delay time of the PE array during execution. A cell library that uses higher threshold transistors was also introduced.

The performance of a real chip with the 65nm SOTB-CMOS process was almost twice that of SOTB when a data transfer centric application *8-alpha* was executed, and at most 600 MOPS performance was achieved with an application program that was highly arithmetic intensive *alpha* at only 1.6 mW. By controlling the body-bias voltage, 743 MOPS/mW (297 MOPS/0.4 mW) sustained performance was achieved for *gray* at 0.5-V supply voltage.

Now, the CMA-SOTB-2 was evaluated with only application programs involving simple image processing. Evaluations using more practical application programs are our future work.

Acknowledgment

This work was done in "Ultra-Low Voltage Device Project" of LEAP funded and supported by METI and NEDO. This work is also supported by VDEC, the University of Tokyo in collaboration with Cadence Design Systems, Inc.

References

[1] Y.Yuyama, et.al. , "A 45nm 37.3GOPS/W Heterogeneous Multi-core SoC," in *Proc. of ISSCC*, 2010, pp. 100–101.

[2] M.Konijnenburg, and et.al. , "Reliable and Energy-Efficient 1MHz 0.4V Dynamically Reconfigurable SoC for ExG Applications in 40nm LP CMOS ," in *Proc. of ISSCC*, 2013, p. 24.6.

[3] N. Ozaki et al., "Cool Mega Arrays: Ultra-low-Power Reconfigurable Accelerator Chips," *IEEE Micro, vol.31, No.6*, pp. 6–18, 2011.

[4] N. Ozaki, et al., "Cool Mega-Array: A highly energy efficient reconfigurable accelerator," *FPT 2011*, pp. 1–8, 2011.

[5] Takashi Ishigaki, et al., "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.

[6] R. Tsuchiya, et al., "Silicon on thin BOX : a new paradigm of the CMOSFET for low-power and high-performance application featuring wide-range back-bias control," *Tech. Dig. Int, Electron Devices Meet., 0-7803-8684-1, San Francisco*, pp. 631–634, 2004.

[7] H. Su, H. Amano, "Body bias control for a coarse grained reconfigurable accelerator implemented with Silicon on Thin BOX technology," *Proc. of FPL*, pp. 1–6, 2014.

[8] V.Tunbunheng, M.Suzuki, H.Amano, "RoMultiC: Fast and Simple Configuration Data Multicasting Scheme for Coarse Grain Reconfigurable Devices," in Proc. of IEEE FPT. Springer, Berlin, 2005, pp. 129–136.

[9] V. Tunbunheng and H. Amano, "Black-Diamond: a Retargetable Compiler Using Graph with Configuration Bits for Dynamically Reconfigurable Architectures," in *Proc. of The 14th Workshop on Synthesis And System Integration of Mixed Information technologies (SASIMI)*, 2007, pp. 412–419.

[10] A.Tsusaka, et al., "A hardware complete detection mechanism for an energy efficient reconfigurable accelerator CMA," *Proc. of FPL*, pp. 1–4, 2013.

[11] R.Uno, et al., "A Speculative Gather System for Cool Mega-Array," *Proc. of ICFPT*, pp. 346–349, 2013.

[12] Fang-Li Yuan, et al., "A 13.1GOPS/mW 16-Core Processor for Software-Defined Radios in 40nm CMOS," *VLSI Circuits Digest of Technical Papers*, pp. 1–2, 2014.

[13] Cheng C. Wang, et al., "A 1.1 GOPS/mW FPGA Chip with Hierarchical Interconnect Fabric," *VLSI Circuits*, pp. 136–137, 2011.