Glitch-aware Variable Pipeline Optimization for CGRAs

Takuya Kojima, Naoki Ando, Hayate Okuhara and Hideharu Amano
Dept. of ICS, Keio University, Yokohama Japan
email: wasmii@am.ics.keio.ac.jp

***Abstract*—Although some coarse grained reconfigurable arrays (CGRAs) have a function to unify multiple processing elements (PEs) to enhance the energy efficiency, it sometimes causes propagation of glitches widely resulting in the power increases. We propose a dynamic power model considering glitches and an optimization technique using it for CGRAs. The model aims to estimate the energy consumption from the switching counts of a PE array approximately. The model and optimization were applied to a real chip of the low power CGRA called the VPCMA (Variable Pipeline Cool Mega Array). Compared with the energy estimation with a post-layout simulation, the model could estimate it with more than 10000 times faster with smaller error from the results of the real chip measurement. The optimized pipeline structure using the proposed method achived better energy consumption compared to fixed pitch pipeline structures in most cases.**

I. Introduction

The forthcoming IoT (Internet of Things) devices and wearable computers require higher performance yet extremely low power consumption computing. CGRAs (Coarse-Grained Reconfigurable Arrays) can be an architecture to satisfy such demand because of their high energy efficiency. Most of the CGRAs consist of an array of small processing elements (PEs) which can execute simple computational operations with ALUs, and distributed memory modules connected together with an interconnection network. Changing the type of operations and their interconnection enables users to execute applications efficiently.

VPCMA (Variable Pipeline Cool Mega Array) [1] is a low power CGRA based on the concept of CMA (Cool Mega Array) [2]. As other CGRAs, it has a large PE array and a tiny microcontroller with banked data memory modules. Unlike dynamically reconfigurable CGRAs, each PE is not configured dynamically and also it doesn't provide a register file. The PE array is pipelined and its structure is altered so as to fit target algorithms and required performance. When a pipeline register is bypassed, the previous stage and the following stage are unified, and PEs in the unified stages are connected without registers. Although the clock frequency tends to be lowered with such a stage unification, it is useful to balance the delay of each stage, resulting to reduce the energy without degrading the performance. The trade-off of the pipelined unification of VPCMA was discussed in [1], and the similar concept has been researched on more general CGRAs with dynamic reconfiguration mechanisms[3][4].

The important factor on such unification of PEs is effect of the glitch propagation. The glitch is undesired switching caused by the difference of delay with large combinatorial circuits. Since it is widely propagated in multiple PEs, the increasing power consumption often degrades the effect of reducing power by the unification. All of the previous researches did not take care of it quantitatively, while some of them pointed out its influences[1]. Here, we propose a dynamic power model considering glitch propagation. Then, an optimization method for the pipeline structure of the VPCMA is introduced. The proposed model and optimization results are examined with real chip experiments.

The rest of the paper is organized as follows. Section II introduces the VPCMA. Then, a dynamic power model and an optimization method are proposed in Section III. An implementation of the VPCMA chip is described in Section IV-A, and experimental results are presented in Section V. After discussion on related work in Section VI, we conclude this paper in Section VII.

II. The Architecture of Variable Pipeline Cool Mega Array

The VPCMA falls into a category of Straight Forward CGRAs (SF-CGRAs), which are consisting of a pipelined PE array, memory modules and networks for exchanging data between them. In order to handle such data transfer efficiently, a permutation network is provided at the input/output of the PE array. Data read out from the memory modules are forwarded to the pipelined array, and then the computational results are written back to the memory modules. The operation of PEs and the interconnection network between PEs are controlled by the configuration data, and sometimes switched dynamically. CGRAs such as Piperench [5], Kilo-core [6] and S5 engine [7] are also classified into this category. Some of these SF-CGRAs can be treated as a class of VLIW (Very Long Instruction Word) computers.

The VPCMA architecture is a simple SF-CGRA aiming to save any energy consumption other than that needed for computation. Each PE is consisting of combinatorial circuit so that the clock distribution is unnecessary.

Fig. 1 shows the diagram of the VPCMA. It is consisting of a large PE array with pipeline registers, a microcontroller and banked data memory. Once all data are set up in the input "Fetch register", computation begins and the outputs of the PE array are stored in the "Gather register" with a certain delay time. The details of a PE is also described in Fig. 1. An arithmetic logic unit (ALU), input selectors, and a switching element (SE) are placed in each PE. The interconnection network between PEs is a mixture of direct interconnections and an island-style network. The output of

978-1-5386-3797-5/17/$31.00 © 2017 IEEE

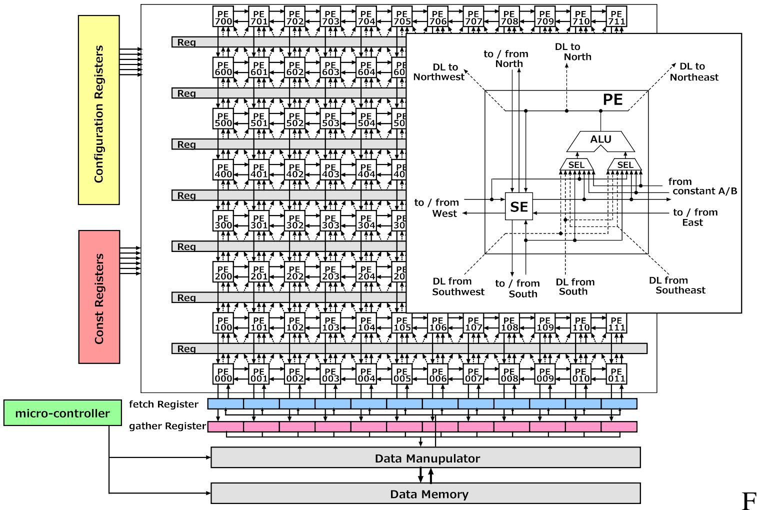

Configuration Registers
Const Registers
PE 700 PE 701 PE 702 PE 703 PE 704 PE 705 PE 706 PE 707 PE 708 PE 709 PE 710 PE 711
PE 600 PE 601 PE 602 PE 603 PE 604
PE 500 PE 501 PE 502 PE 503 PE 504
PE 400 PE 401 PE 402 PE 403 PE 404
PE 300 PE 301 PE 302 PE 303 PE 304
PE 200 PE 201 PE 202 PE 203 PE 204
PE 100 PE 101 PE 102 PE 103 PE 104 PE 105 PE 106 PE 107 PE 108 PE 109 PE 110 PE 111
PE 000 PE 001 PE 002 PE 003 PE 004 PE 005 PE 006 PE 007 PE 008 PE 009 PE 010 PE 011
Reg
DL to Northwest
to / from North
DL to North
DL to Northeast
PE
ALU
SEL
SEL
SE
from constant A/B
to / from West
to / from East
DL from Southwest
to / from South
DL from South
DL from Southeast
micro-controller
fetch Register
gather Register
Data Manupulator
Data Memory

Fig. 1: Diagram of VPCMA

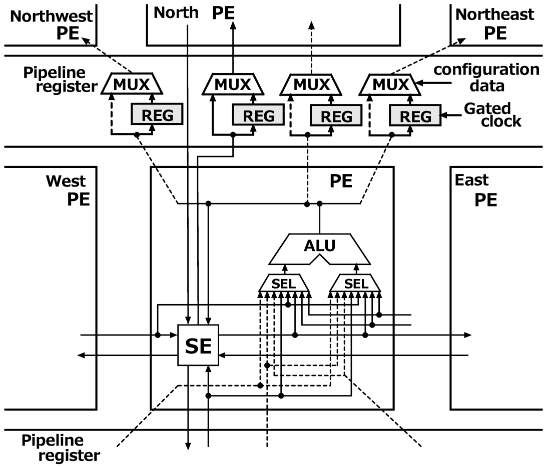


Fig. 2: Details of a PE and Pipeline Registers in VPCMA

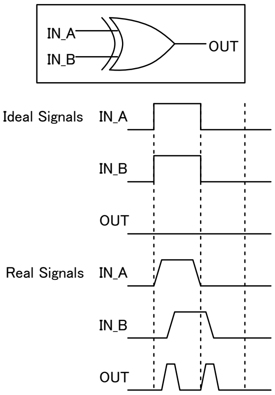


Fig. 3: An example of glitch generation

an ALU is directly transferred to the adjacent PEs: the north, northeast, and northwest PEs as described as dashed lines in Fig 1. These paths are called direct links. SEs in each PE provide both vertical connection and horizontal connection of the island-style network. The results computed in the ALU can be directly forwarded to the "Gather registers" through paths to the south direction.

As illustrated in Fig. 1, the pipeline registers are put between every row of the PE array. Here, we assume the 8 × 12 PE array with seven pipeline registers. Users can decide whether to activate each of them independently so that its pipeline structure can be altered widely. Fig. 2 shows an implementation of pipeline registers. To change its behavior alternatively: *latch* or *bypass*, the data from either the registers or output of ALU are chosen by the multiplexer according to the configuration data. In case of bypassing mode, the clock-gating is applied to the registers to reduce the dynamic power consumption. There is no register on the south direction path from the north PE, because it is used only to transfer computational results.

The "Fetch register" is connected to the input of the PE array and the "Gather register" is connected to the output of it. The input data are sent from the banked data memory (MEM) to the "Fetch register" by an instruction of the micro-controller, and results data are sent from "Gather register" to MEM in the same manner. In this paper, "Fetch" and "Gather" indicate the former and latter data transfer respectively. After "Gather" operation is issued, its execution is kept waiting for some clocks corresponding to pipeline latency of PE array.

A permutation network called the data manipulator provides flexible data transfer among the banked memory, Fetch register and Gather register. In case of "Fetch", the data read from each memory bank can be transferred into any position of Fetch register. Each mapping between memory bank and Fetch register is specified by mapping tables and a micro-instruction indicates which of tables is applied. In this way, various application program can be implemented without power-hungry dynamic reconfiguration.

III. Dynamic power model considering glitch propagation

A. Glitch propagation

Glitches are undesired signal transitions caused by the difference of signal propagation which consumes a certain power by switching without contribution to the computation. They can be generated by every gate, but especially XOR gates often become a source. Fig. 3 shows an example of glitch generation at an XOR gate. In an ideal case, "OUT" does not transit from "0". However, because of the delay of "IN_B", short pulses are generated on "OUT". They are called glitches.

Likewise, the different delay times between inputs of the PEs results in such glitches. If a pipeline register is bypassed, they are propagated to the PEs in the following stages and increase the power consumption. In contrast, although activating pipeline registers restricts the propagation, it requires the power for clock distribution and storing data into registers.

B. Proposed Power Model

For finding the pipeline structure which can reduce the power considering the aforementioned trade-off, a dynamic power model considering glitches is required.

The dynamic power P_{dyn} of CMOS VLSI is generally

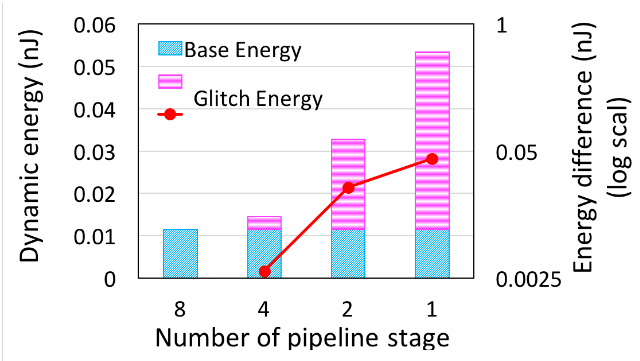


Fig. 4: An example of glitch generation

calculated by:

$$P_{dyn} = \alpha C f V_{dd}^2 \tag{1}$$

where α is switching activity, C is load capacitance, f is clock frequency, and V_{dd} is supply voltage.

Based on Equation (1) and assuming a fixed V_{dd}, the dynamic power of combinatorial circuits ($P_{dyn,comb}$) in the PE array is simplified as:

$$P_{dyn,comb} = E_{sw} S_{total} f, \tag{2}$$

where E_{sw} is the average energy consumption per a switching, and S_{total} is switching count of the PE array including glitches. E_{sw} is depending on the process technology and circuit structure in a PE. Here, we assume that it is a constant value, and can be obtained from the simulation of the target chip. Note that $P_{dyn,comb}$ only represents switching power of combinatorial circuits. The power for clock tree and storing/reading registers is considered later.

Thus, we focus to obtain an approximate value of S_{total} close to the real value as possible. In order to model the influence of the glitch, we simulated the target VPCMA shown in Section IV-A with various number of pipeline stages.

Fig. 4 shows an example of consumed energy in combinatorial circuits with various number of pipeline stages. The application program is *gray* shown in Section V-A. With 8-stage pipeline structure, each PE is divided with pipeline registers, thus the propagation of glitch is minimized. This result is shown with the blue part of the bar, and the pink part is the difference of the energy when the pipeline stages are unified. That is, it represents the energy by propagating glitches. Note that it is only the energy of combinatorial circuits, and one for clock tree and registers are excluded. Fig. 4 also shows the difference of energy in the log-scale. In this example, it increases over linearly to the increasing number of unified stages, yet the increase is not exactly exponential. Thus, we assume that the glitch basically increases exponentially as the number of unified PEs increases, since the total number of switching signals is accumulative in each PE. However, a part of propagated glitches are shut out with PEs and interconnection networks. We also assume that glitches generated by the previous PEs are relative to the largest switching count of inputs when the PE receives data from multiple PEs. Considering them, we represent the switching activity of the PE array as the following equations.

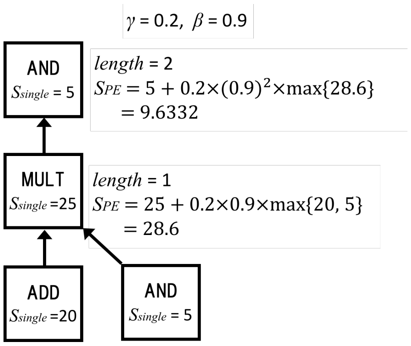


Fig. 5: An example of the glitch propagation model

$$\begin{aligned} S_{total} &= \sum_{i=0}^{n}\sum_{j=0}^{m} S_{PE}(i,j) && (3)\\ S_{PE} &= S_{single}(op) + \beta\gamma^{length} \max_{dir} S_{prev}(dir) && (4)\\ dir &= \{\text{SOUTH, WEST, EAST,DL\_South,} \\ &\quad\ \text{DL\_Southwest, DL\_Southeast}\} && (5) \end{aligned}$$

where:

- $S_{PE}(i,j)$ is a switching count of a PE at the i-th row and the j-th column, i and j are integers ranging [0,n] and [0, m] reflecting the $m \times n$ PE array structure.
- S_{single} is a switching count of a PE without glitch propagation and *op* represents an operation assigned to the PE.
- S_{prev} is a switching count of a PE in the previous stage, β and γ are propagation factors, *length* is distance from the nearest active pipeline register, and *dir* is input direction of the previous PE.

In Equation (4), the last term represents the switching caused by the glitch propagation. If a pipeline register between a PE and connected PE in the previous stage is activated, S_{prev} is considered to be zero. Also, when no operation is assigned into the PE in the previous stage, S_{prev} is also considered as zero.

Fig. 5 shows an example of calculating S_{total} with the model. A square in the figure represents a PE with its operation and switching count (S_{single}). Let the propagation factor β be 0.9, and γ be 0.2, respectively. Since PEs in the first stage have no input propagated glitches, $S_{PE} = S_{single}$. PEs in the second stage has two data inputs from two different PEs. Here, larger switching count is adopted, and thus the calculation is done as shown in the figure. The switching count in the third stage is done in the same manner, and so the total switching count S_{total} becomes $20 + 5 + 28.6 + 9.6332 = 63.2332$.

Here, we assume that β and γ are fixed value independent from the application programs, and obtained with the simulation results or real chip measurements. Other parameters are fixed when the target application program and its mapping are fixed. It means that the total switching count can be estimated when an application program and its mapping are fixed.

C. Finding the optimal pipeline structure

The VPCMA produces $2^7 = 128$ patterns of the pipeline structure since it provides 7 pipeline registers. The static power

TABLE I: Specifications of the VPCMA

Design	Verilog HDL
Simulator	Cadence NC-Verilog
Process	Renesas SOTB 65 nm / LPT-8
Synthesis	Synopsys Design Compiler 2016.03-SP4
Place and route	Synopsys IC Compiler 2016.03-SP4
Chip size	6mm × 3mm

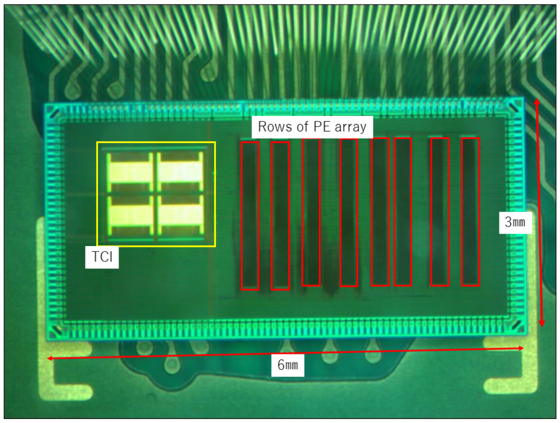


Fig. 6: Chip photograph of the VPCMA

is constant regardless of the pipeline structure. For a target application program, the mapping is assumed to be fixed, thus, we can find the optimal pipeline structure which satisfies the conditions expressed as follows:

$$\min(P_{dyn,comb}) + P_{dyn,preg} \times N_{preg} \tag{6}$$

subject to

$$D_l = \sum D_{PE}(op) \tag{7}$$
$$D_l \leq D_{req}, \quad \forall \text{ datapath } l \tag{8}$$

where $P_{dyn,preg}$ is a dynamic power of a pipeline register, N_{preg} represents the number of active pipeline registers, and D_{PE} is the delay of a PE depending on the executed operation. Thus, D_l, the combinatorial delay time of the l-th path must be less or equal to the maximum allowable delay time D_{req}, the inverse of the required frequency f. Like the switching counts, D_{PE} also depends on the assigned operation, and so is fixed after the mapping.

IV. Target VPCMA

A. chip implementation

The real chip of the VPCMA was implemented with specifications described in Table I. Fig 6 is the photograph of the VPCMA. In the photograph, portions surrounded by red frames correspond to rows of the PE array. It includes a channel for the wireless inductive coupling through interface (TCI) which is not in the scope of this paper.

The chip was implemented with Renesas 65nm SOTB process. The SOTB is an FD-SOI technology in which transistors are formed on a thin buried oxide (BOX) layer as illustrated in Fig. 7. Its transistor structure enables to operate at a relatively high clock frequency with low supply voltages. Among the numerous benefits of SOTB [8], the bias voltages supplied to the body (VBN and VBP) are adjusted to control the delay and leakage power consumption. When VBN is set to zero and VBP is set to the same voltage as supply voltage VDD, the threshold level of the SOTB is just like the common CMOS processes, and it is called zero-bias. Since this work focuses on reducing the dynamic power, zero-bias is always applied here.

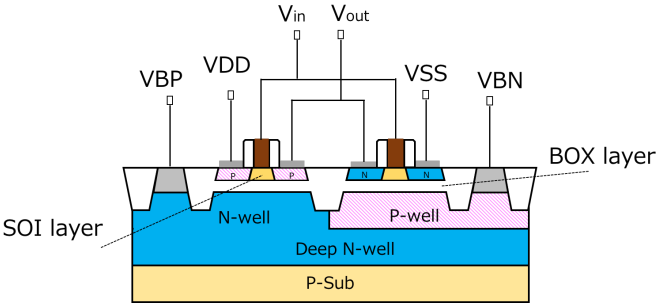


Fig. 7: Cross-sectional view of the SOTB MOSFET

B. Obtaining model parameters

1) Architecture depending parameters: In the proposed model, a switching energy of an output E_{sw}, and glitch propagation parameters β and γ are depending on the process technology and hardware structure of the target system. Although they can be obtained from the simulation, we evaluated the real chip, and computed them from the results. The energy consumption of 128 pipeline structures were measured, and least square method is applied. The results are as follows:

E_{sw} = 0.1117, β = 1.325 and γ = 0.053.

Although the measurement requires a certain effort, once they can be obtained, they can be used as constants in all target applications and required frequency.

2) $S_{single}(op)$: $S_{single}(op)$ is depending on the operation of the PE, and it should be evaluated for each operation. Table II shows the average switching counts in a PE when an operation is executed. For each operation, RTL simulation with NC-Verilog is done 100000 times for random input values, and the switching counts at the output of PEs were recorded.

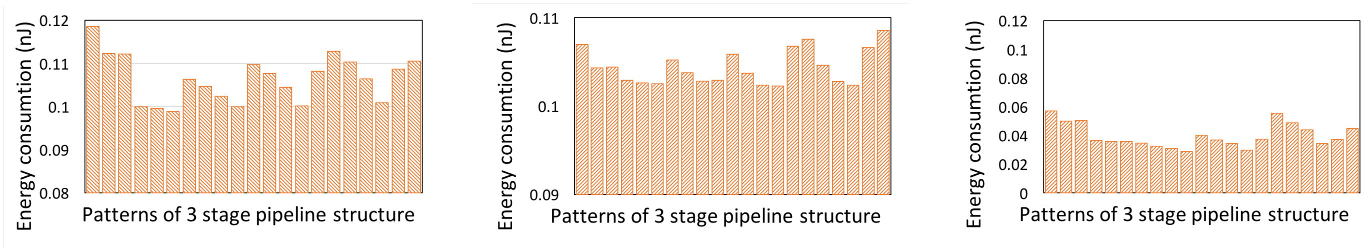
In comparison with shift operations (SL, SR and SRA) logical operations (AND, OR and NOT) bring more switching counts. The switching counts of arithmetic operations (ADD, SUB and MULT) are the largest, and about triple of those of shift operations.

V. Evaluation

The benefits of the model are (1) the power can be estimated without executing time consuming post-layout simulation, and (2) the optimized pipeline structure is searched automatically for a given application program with a required frequency. Here, the accuracy and computation time of the model are evaluated, then, obtained pipeline structures from the model are examined.

TABLE II: Average switching counts of each operation in a PE

op	ADD	SUB	MULT	SL	SR	SRA	AND	OR	NOT
switching	17.1693	20.0153	31.4623	6.79133	4.97267	7.318	5.217	16.9223	11.1277



(a) Real chip measurement

(b) Propose model

(c) PrimeTime simulation

Fig. 8: Results of the dynamic energy consumption

TABLE III: Simulated applications

Application	Description	Used row
gray	24 bit (RGB) gray scale	8 rows
sepia	8 bit sepia filter	6 rows
sf	24 bit (RGB) sepia filter	8 rows
af	24 bit (RGB) alpha blender	7 rows
dct	8-point DCT	8 rows

A. Dynamic power estimated from the model

Five application programs listed in Table III were executed on the VPCMA chip. To measure the energy consumption of all pipeline structure, a low clock frequency (5MHz) (D_{req} = 200 ns) with a low supply voltage (0.55 V) was used. First, we will compare the energy measured by the real chip, one from the model, and one with the post-layout simulation. The performed application is *gray*, and the number of pipeline stage is three. For three stages, the pipeline structure can take $\frac{7!}{2!(7-2)!}$ = 21 patterns.

The energy consumption of combinational circuits measured by the real chip is shown in Fig 8(a). From the result, it is apparent that the energy consumption depends on the pipeline structure in the real chip measurement. The energy consumption calculated by the proposed model is presented in Fig 8(b). The values contain the error of 3.327 % on average compared to the real chip measurement, but it well estimates the real chip results. On the contrary, the power from the post-layout-simulation result shown in Fig8(c) includes a large error up to 71.1%. It is because the characterization of the SOTB process is not well matched to the real chip, and the error will be reduced with other common processes. However, in general, a certain error is included in the estimation of the power consumption with post-layout simulations.

B. Accuracy of the model

The error of the proposed model was analyzed for all application programs and all possible pipeline structures, and shown in Table IV. The relative mean error (ME) is caluculated as follows.

$$ME = \frac{1}{N}\sum_{k}^{N}\frac{|E_{model,k} - E_{real,k}|}{E_{real,k}} \tag{9}$$

where N is the number of possible pipeline structure, $E_{model,k}$ is the energy calculated by the proposed model in case of k-th pipeline structure, and $E_{real,k}$ is that measured from the real chip.

TABLE IV: Error of the proposed model

	gray	sepia	af	sf	dct
ME (%)	24.78	34.99	4.72	5.11	7.08
Max (%)	31.35	46.76	9.40	28.91	26.47
Min (%)	21.13	23.03	3.06	0.15	0.12

The error of the model is enough small in *af*, *sf* and *dct*, but relatively large in *sepia* and *gray*. These application programs include a large number of arithmetic operations, and the output data of each PE is likely to contain a lot of zero bits. Since the random input data are used for obtaining the switching counts for a PE, the energy was estimated higher than the case of using real application programs. That is, the model becomes slightly conservative, if we used random input data for characterize.

C. Computation time

By using the proposed model, the energy of the pipeline structure can be obtained with a spread sheet software based on the mapping results and parameters in the previous section. Here, we developed a dedicated C program to compute it. It takes about 0.05 second to evaluate 128 patterns.

On the other hand, for obtaining the dynamic power including glitch propagation, the SAIF(Switching Activity Interchange Format) data must be obtained by the post-layout simulation, then the power analysis tool must be used. When Cadence NC-Verilog and Synopsys PrimeTime are used for the analysis, the total time for obtaining the power for a pipeline structure takes about 6 minutes with a high-end server. Thus, it is difficult to evaluate all pipeline structures, since it requires more than 10 hours.

That is, the proposed model is quite useful to estimate the power of various type of pipeline structures quickly.

D. Finding the suitable pipeline structures

To check the reliability of the proposed search method, five application programs described above were executed for several performance requirements. For comparison, a fixed pitch pipeline structures (1,2,4,and 8-stage) are also evaluated.

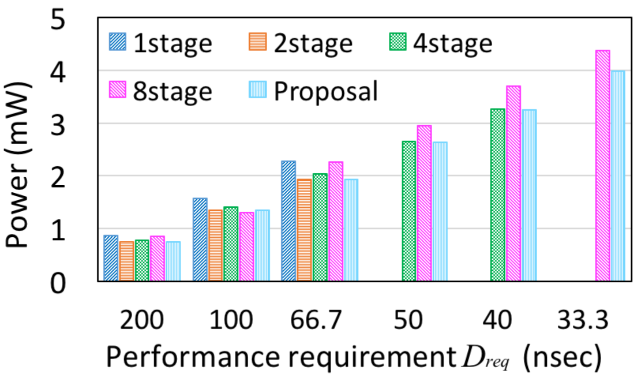


Fig. 9: Total Power of pipeline structures (gray)

TABLE V: Energy Comparisons of each structure (pJ)

	1-stage	2-stage	4-stage	8-stage	Proposal
gray	160.86	137.20	138.97	148.64	134.39
sf	153.89	121.42	122.32	134.18	117.34
dct	212.85	171.71	146.05	153.97	150.06
	1-stage	2-stage	4-stage	7-stage	Proposal
af	138.07	127.90	130.70	139.90	126.45
	1-stage	2-stage	3-stage	6-stage	Proposal
sepia	107.12	100.21	99.21	109.60	98.66

Examples of the power of executing *gray* for various required frequencies are shown in Fig 9. Here, ”Proposal” means the pipeline structure recommended by the model. In all cases, the static power was 0.126 mW. Both 1-stage pipeline and 2-stage pipeline could not fulfill D_{req} = 50 ns. Likewise, 4-stage pipeline could not satisfy D_{req} = 33.3 ns. The result shows that the proposed method can find the appropriate pipeline structure while satisfying the performance requirements.

For each application program and the target structure, the average energy consumption among several required frequencies is calculated as Table V. In case of dct, the pipeline structure recommended by the proposed method is slightly worse than the fixed 4-stage pipeline.

This comes from the following two reasons. First, the proposed method used the worst case delay time to make sure the operations, while the fixed pipeline was measured as its best result on the real chip. The proposed method removed 4-stage pipeline from candidates and uses more pipeline registers for conservative design. Second, an error of the dynamic power model can influence the result. As shown in Table IV, in case of *dct*, the model contains 7.01% of error on average. When the number of pipeline stage is small, the model can include more error, and miss more appropriate pipeline structure.

VI. Related work

The trade-off between inserting registers and unifying PEs has been examined in various type of CGRAs. [9] proposed a high level description for CGRAs, and carried out design exploration considering the trade-off. ADRES applied a dynamic operation fusion techniques for its processing elements,[3] and EGRA also proposed a scheduler to exploit the time slack of PEs[4]. However, all of them did not model nor take into account the influence of glitches explicitly.

The tool for analyzing the glitch power was proposed for FPGAs in [10]. It reduces glitch power by activating unused flip-flops. Although the approach is similar, it is not for CGRAs with the pipelined structure.

VII. Conclusion

We propose a dynamic power model considering glitches and an optimization technique using it for CGRAs. The model aims to estimate the energy consumption from the switching counts of a PE array approximately. Compared with the energy estimation with a post-layout simulation, the model could estimate it with more than 10000 times faster with smaller error from the results of the real chip measurement than a post-layout simulation. The optimized pipeline structure using the proposed method achieved better energy consumption compared to fixed pitch pipeline structures in most cases.

This paper assumed a fixed application mapping on the PE array. However, the energy efficiency can be increased the mapping is changed considering the glitch propagation. Developing such a mapping tool integrating the proposed model is our future work. Besides, although this paper focused on the optimizing only the dynamic power, the static power should be taken into consideration by applying the body bias control to the VPCMA.

References

[1] N. Ando, K. Masuyama, H. Okuhara, and H. Amano, “Variable pipeline structure for coarse grained reconfigurable array cma,” in *2016 INTERNATIONAL CONFERENCE ON FIELD-PROGRAMMABLE TECHNOLOGY*, 2016, pp. 231–238.

[2] N. Ozaki, Y. Yoshihiro, Y. Saito, D. Ikebuchi, M. Kimura, H. Amano, H. Nakamura, K. Usami, M. Namiki, and M. Kondo, “Cool mega-array: A highly energy efficient reconfigurable accelerator,” in *Field-Programmable Technology (FPT), 2011 International Conference on*. IEEE, 2011, pp. 1–8.

[3] Y. Park, H. Park, and S. Mahlke, “Cgra express: accelerating execution using dynamic operation fusion,” in *Proceedings of the 2009 international conference on Compilers, architecture, and synthesis for embedded systems*. ACM, 2009, pp. 271–280.

[4] G. Ansaloni, P. Bonzini, and L. 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.

[5] H. Schmit, D. Whelihan, A. Tsai, M. Moe, B. Levine, and R. R. Taylor, “Piperench: A virtualized programmable datapath in 0.18 micron technology,” in *Custom Integrated Circuits Conference, 2002. Proceedings of the IEEE 2002*. IEEE, 2002, pp. 63–66.

[6] B. Levine, “Kilocore: Scalable, High Performance and Power Efficient Coarse Grained Reconfigurable Fabrics,” in Proc. of International Symposium on Advanced Reconfigurable Systems, 2005, pp. 129–158.

[7] J. M. Arnold, “S5: the architecture and development flow of a software configurable processor,” in *Proceedings. 2005 IEEE International Conference on Field-Programmable Technology, 2005*. IEEE, 2005, pp. 121–128.

[8] Y. Morita, R. Tsuchiya, T. Ishigaki, N. Sugii, T. Iwamatsu, T. Ipposhi, H. Oda, Y. Inoue, K. Torii, and S. Kimura, “Smallest Vth variability achieved by intrinsic silicon on thin BOX (SOTB) CMOS with single metal gate,” in *2008 Symposium on VLSI Technology*, June 2008, pp. 166–167.

[9] A. Chattopadhyay, X. Chen, H. Ishebabi, R. Leupers, G. Ascheid, and H. Meyr, “High-level modelling and exploration of coarse-grained reconfigurable architectures,” in *Proceedings of the conference on Design, automation and test in Europe*. ACM, 2008, pp. 1334–1339.

[10] C.-T. Hsieh, J. Cong, Z. Zhang, and S.-C. Chang, “Behavioral synthesis with activating unused flip-flops for reducing glitch power in FPGA,” in *Proceedings of the 2008 Asia and South Pacific Design Automation Conference*. IEEE Computer Society Press, 2008, pp. 10–15.