Electronics **2014**, *3*, 661-674; doi:10.3390/electronics3040661

OPEN ACCESS

electronics

ISSN 2079-9292

www.mdpi.com/journal/electronics

Article

Optimizing Power Heterogeneous Functional Units for Dynamic and Static Power Reduction

Toshinori Sato * and Yoshimi Shibata †

Department of Electronics Engineering and Computer Science/8-19-1 Nanakuma, Fukuoka University, Jonan-ku, Fukuoka 814-0180, Japan; E-Mail: TL081297@cis.fukuoka-u.ac.jp

† The author is currently with Hitachi Solutions, Ltd.

* Author to whom correspondence should be addressed; E-Mail: toshinori.sato@computer.org; Tel.: +81-92-871-6631 (ext. 6388); Fax: +81-92-865-6031.

External Editor: Bimal K. Bose

Received: 2 June 2014; in revised form: 9 October 2014 / Accepted: 18 November 2014 / Published: 10 December 2014

Abstract: Power consumption is the major constraint for modern microprocessor designs. In particular, static power consumption becomes a serious problem as the transistor size shrinks via semiconductor technology improvement. This paper proposes a technique that reduces the static power consumed by functional units. It exploits the activity rate of functional units and utilizes the power heterogeneous functional units. From detailed simulations, we investigate the conditions in which the proposed technique works effectively for simultaneous dynamic and static power reduction and find that we can reduce the total power by 11.2% if two out of four leaky functional units are replaced by leakless ones in the situation where the static power occupies half of the total power.

Keywords: power consumption; leakage power; out-of-order processors; functional units

1. Introduction

As semiconductor technologies have improved aggressively, power consumption due to leakage current has become one of the serious problems in microprocessor designs [1,2] and will continue to be so [3]. The static power is always consumed regardless of the circuit activity, and thus it tremendously increases as the number of transistors on a single chip increases. Several techniques are proposed to eliminate or reduce the static power consumption [4], such as power gating [4,5], MRAM-based logic [6], and leakage current control techniques [4], which utilize multiple threshold voltages and adaptive body bias.

Power gating [4,5] is a virtual power switch inserted into the target circuit in series. When the sleep transistor is off, the power supply is virtually cut off, thus static power is almost eliminated. The issue to be considered in the adoption of a sleep transistor is break-even time (BET) [5]. This is because the power switch consumes dynamic power to discharge and charge the capacitance of the target circuit. It also requires the additional circuit that controls the power switch, which also consumes additional power. BET is a time period in which the power saved by cutting the power supply is equal to the total overhead mentioned above. The period when the power supply is cut off should be longer than BET to reduce power and it could be considerably long. Ikebuchi *et al.* [5] proposed to predict the time when the functional units in an in-order processor are inactive and turn off the power supply only when the period is predicted to be longer than BET.

While it is easy for in-order processors to predict which functional unit will be active, it is almost impossible for out-of-order processors. This is because the functional unit is selected just before the instruction is ready for execution, therefore there is not sufficient time to make a prediction. Recently, even embedded processors have adopted out-of-order execution. For example, Apple's A7 [7] and ARM's Cortex-A15 [8], which are integrated in iPhone 5S and in NVIDIA's Tegra-K1 [9], respectively, are out-of-order processors. These systems on a chip (SoCs) are widely used in smartphones and tablet PCs, which require high performance and low power consumption. Furthermore, the high-performance computing (HPC) market also expects high performance and low-power processors. For example, ARM Cortex-A57 [10], which is the latest out-of-order embedded processor, is scheduled to be used in HPC and mobile computing. Hence, a new technique to reduce static power consumed by out-of-order processors is strongly desired.

This paper is organized as follows. The next section explains how CMOS circuits consume power. Section 3 proposes to utilize power heterogeneous functional units to simultaneously reduce dynamic and static power consumptions. Section 4 introduces our evaluation environment, and Section 5 presents the simulation results. Section 6 surveys previous studies, and Section 7 concludes this paper.

2. CMOS Power Consumption

CMOS power consumption is controlled by the equation:

$$P = P_{dynamic} + P_{static} \tag{1}$$

where $P_{dynamic}$ is the dynamic power and P_{static} is the static power. $P_{dynamic}$ and gate delay t_{pd} are given by:

$$P_{dynamic} \propto f \cdot C_{load} \cdot V_{dd}^2 \tag{2}$$

$$t_{pd} \propto \frac{V_{dd}}{(V_{dd} - V_t)^{\alpha}} \tag{3}$$

where f is the clock frequency, C_{load} is the load capacitance, V_{dd} is the supply voltage, and V_t is the threshold voltage of the device. α is a factor dependent on the carrier velocity saturation and is ~1.3–1.5. It is clear that a power supply reduction is the most effective way to lower the power consumption. However, reduction in the supply voltage increases gate delay, resulting in lower performance in the microprocessor. To maintain high transistor switching speeds, the threshold voltage needs to be proportionally scaled down with the supply voltage.

On the other hand, the static power can be given by:

$$P_{static} = I_{static} \cdot V_{dd} \tag{4}$$

where I_{static} is the leakage current. The major part of the leakage current is the subthreshold leakage current and is dominated by threshold voltage V_t in the following equation:

$$I_{static} \propto 10^{-V_t/S} \tag{5}$$

where S is the subthreshold swing parameter.

Therefore, lower threshold voltage leads to increased subthreshold leakage current and thus an increase in the static power. Maintaining high transistor switching speed via low threshold voltage gives rise to a significant amount of static power consumption. In other words, it is very difficult to reduce both dynamic and static power while maintaining high performance.

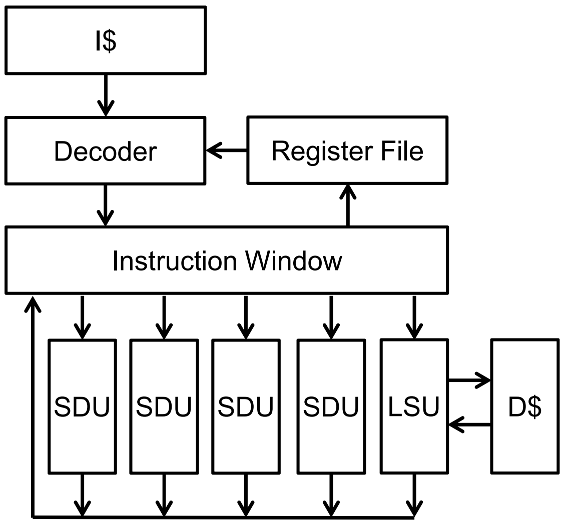
3. Power Heterogeneous Functional Units

In this section, we propose a technique that simultaneously reduces dynamic and static power consumption by optimizing the utilization of functional units.

Figure 1 shows a superscalar processor core. Every instruction is fetched from the instruction cache (I$ in the figure) and is decoded in the decoder (Decoder in the figure) and then dispatched into the instruction window (Instruction Window in the figure), where the instruction waits until its operands are ready. When the instruction is ready to execute, it is issued into one of the arithmetic and logical units (SDUs in the figure) or into the load store unit (LSU in the figure) in an out-of-order fashion. In this paper, we use the term *functional unit* to refer to the arithmetic and logical unit and we call the functional unit made of leaky transistors the *static power dominated unit* (SDU). SDU works for arithmetic and logical operations, while LSU works for load and store operations, which refer to the data cache (D$ in the figure). From the discussions in Section 2, we already know that high-performance processor cores should use low threshold voltage transistors to obtain high performance, though these transistors have higher leakage current. Even when the instruction finishes, it stays in the Instruction Window until it moves into the head of the window. Finally, its execution result is written back into the register file (Register File in the figure) in an in-order fashion.

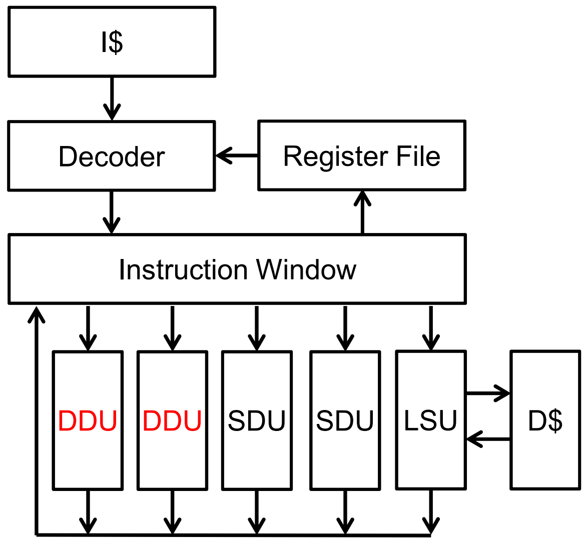
Modern superscalar processor cores have multiple SDUs and each SDU consumes a considerable amount of static power regardless of its activity. A naïve technique to reduce the static power is to replace the leaky transistors by leakless ones, which have a high threshold voltage. Unfortunately, this increases the dynamic power consumed by the functional units if we expect the performance to be maintained, as explained in Section 2. A technique that considers the trade-off between dynamic and static power consumptions is required. Here, we can exploit the activity rate of SDUs. It is widely known that instruction-level parallelism is limited, thus every SDU is not always active. In most periods of program execution, the instruction-level parallelism is around two even when there are sufficient functional units. This means that about half of the SDUs are inactive if the processor core has four SDUs, as shown in Figure 1. When the number of instructions to be executed in parallel is less than that of the SDUs implemented in the core, the selection logic in the instruction window determines that into which SDU one of the instructions is issued. Since this selection happens in a static priority order [11], the highest-priority SDU will always be active. On the other hand, the lowest-priority SDU will be active only in the rare case in which the full issue width is used. This policy results in the highest-priority SDU being accessed frequently and the lowest-priority ALU being rarely accessed. Hence, we classify SDUs into two classes: SDUs with high activity rate and those with low activity rate. Power consumed by the former class is dominated by dynamic power, and that consumed by the latter ones is overwhelmingly the static power.

Figure 1. Superscalar processor core.



Considering the above observations, we propose to replace only the SDUs with low activity rate with functional units made of leakless transistors. This reduces the static power consumed by the less-active functional units. Unfortunately, the threshold voltage of the leakless transistor is high, thus the functional unit becomes slow. This affects the dynamic sequence of the instructions and might degrade the processor performance. To compensate for this effect on the speed, we follow [12] and increase the power supply voltage delivered to the slow functional units. In other words, we combine the multiple threshold voltage technique with the multiple supply voltage technique [13] so that the critical path delay does not increase. Though this increases the dynamic power consumed by the units, it might not increase power consumption seriously because their activity rates are low. Here, we call the functional unit that is leakless transistors and that works at high supply voltage *dynamic power dominated units* (DDU). Note that the supply voltage for SDU does not change. Figure 2 shows an example of the superscalar processor core, two of whose SDUs are replaced by DDUs. Its instruction scheduler uses SDUs whenever available. We can see heterogeneity in power among the functional units. This technique simultaneously optimizes the dynamic and static power consumption, thus reducing the total power consumption. It reduces the dynamic power of the processor core, all of whose functional units are DDU, and the static power of the processor core, all of whose functional units are SDUs. In addition, there is no negative impact on performance. One of the possible drawbacks of this heterogeneous design is that the specific SDU might wear out faster than the other units because it works more frequently than the others, which reduces the lifetime of the processor. However, this static selection policy of functional units is commonly used in conventional processors [11], thus we believe that lifetime reduction will not occur in our heterogeneous design.

Figure 2. Processor core with power heterogeneous functional units.



The goal of this study is not to propose the solution to power problems but to propose a choice to relieve the problem, which can be combined with other techniques.

4. Evaluation Methodology

We used the SimpleScalar tool set [14] for cycle-by-cycle simulation. We selected six benchmark programs from the SPEC2000 suite: 164.gzip, 175.vpr, 176.gcc, 197.parser, 255.vortex, and 256.bzip2. For each program, 100M instructions were executed in detail. Table 1 summarizes the processor core configuration.

The configurations of the heterogeneous functional units are as follows. Assumption 1: the processor has four functional units, *i.e.* the sum of the numbers of SDUs and DDUs is four; Assumption 2: the static power consumed by each DDU is negligible because dynamic power is dominant; Assumption 3: in three scenarios for SDUs, the ratio of the static power over the total power is 30%, 40%, and 50%, respectively, when all functional units are SDU, which means that it is a high-performance processor core.

We do not use any simulators such as Wattch [15] and McPAT [16] for power analysis because of the following. We are not interested in the power optimization of a specific process technology, which is the goal of the simulators. Instead, we would like to determine how to arrange future process technology in order to make the proposed technique work best. In other words, we have to find power parameters for both dynamic and static power consumption by evaluations.

Considering the above three assumptions, energy consumption is calculated as follows. First, from the cycle-by-cycle simulation results, average dynamic power through all benchmark programs is obtained by counting the performed activities. Second, according to the above assumption 3, static power is determined. Therefore, for example, static energy is the same among the programs if their execution clock cycles are the same. In contrast, even if their execution cycles are the same, dynamic energy among the programs differs, because the activity rates are different among the programs.

Table 1. Processor core configuration.

Fetch width	8 instructions
L1 instruction cache	16 KB, 2-way, 1-cycle latency
Branch predictor	Gshare + bimodal
Gshare predictor	4 K entries, 12 histories
Bimodal predictor	4 K entries
Branch target buffer	1 K set, 4-way
Dispatch width	4 instructions
Instruction queue size	32 instructions
Issue width	4 instructions
Integer ALU's (SDU/DDU)	4 units
Integer multipliers	2 units
Floating-point ALU	1 unit
Floating-point multiplier	1 unit
Load and store queue	16 entries
L1 data cache	16 KB, 4-way, 2-cycle latency
Unified L2 cache	8 MB, 8-way, 10-cycle latency
Memory	Infinite, 100-cycle latency
Commit width	4 instructions

To calibrate the ratio of static power to the total power according to the assumption, the ratio between the dynamic and static power of an SDU should be determined. Based on the cycle-by-cycle simulation results through all programs, the dynamic power of SDU is calculated as seven, 4.5, and three times more than the static power of SDU, respectively.

Here, we would like to mention how realistic the above assumption 3 is. While process engineers have tried to reduce the static power for nearly a decade after the static power problem became famous, it is still serious, and a ratio of 50% is a possible scenario [17].

Under these assumptions, we investigate how the dynamic power of DDU can be predicted. We use an arbitrary unit for measuring power consumption.

5. Experimental Results

First, we show the baseline performance. Figure 3 shows the instructions per cycle (IPC). The horizontal axis indicates the benchmark programs, and the vertical axis indicates the numbers of IPC. They vary between 0.79 and 2.50, and the average is 1.40. This variation is translated into the utilization of the functional units. Table 2 summarizes how the four functional units are utilized, and clearly, the utilization is very low. Only 15% of the total execution cycles utilize all four functional units on an average. This result is encouraging for our proposed technique, which reduces the static power consumed by idle units.

Figure 3. Baseline performance in IPC.

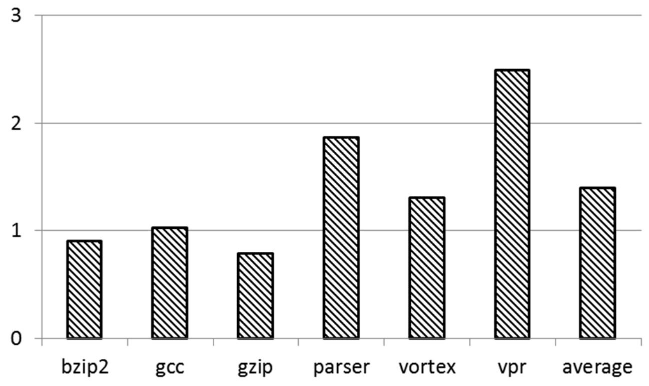


Table 2. Functional units utilization.

Program Name	Average Utilization of Functional Units
164.gzip	0.85
175.vpr	2.52
176.gcc	1.12
197.parser	2.03
255.vortex	1.29
256.bzip2	0.89

Next, we present the break-even point of the proposed technique in Figure 4. The horizontal axis indicates the configurations of the heterogeneous functional units. For example, "DDU:1 SDU:3" means that the processor core has one DDU and three SDUs. The vertical axis indicates the active power of DDU. Each of the three lines corresponds to one of the three scenarios and presents the border where the total power consumption is reduced. If the active power of DDU is under the line, the total power has been reduced successfully. Therefore, the lines indicate the break-even power (BEP) of DDU. For example, in the 50% scenario, the dynamic power of DDU should be less than 11.15, which is 3.72 times larger than that of SDU.

Figure 4. Break-even point for DDU dynamic power.

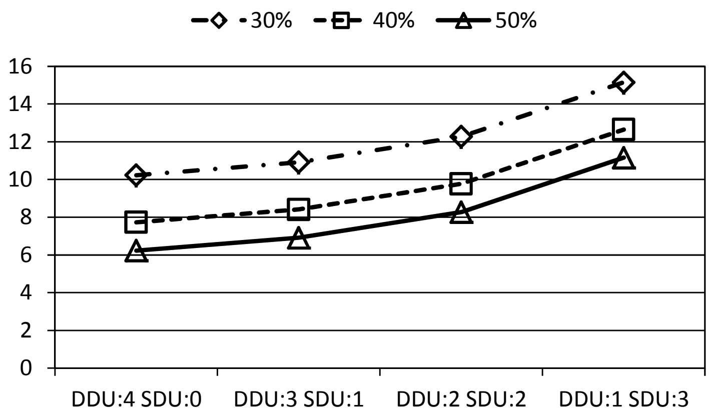


Figure 5 presents the power consumption in the case where the ratio of the static power over the total power is 50% and the dynamic power of DDU is twice as large as that of SDU. The horizontal axis indicates the benchmark programs and the vertical axis indicates relative power compared with the baseline power consumption. Hence, if the bar is lower than 100%, the power consumption is reduced. For each group, four bars from the left to right indicate the result when four, three, two and one SDU(s), respectively, are replaced by DDUs. We can see that the bathtub curve and benchmark programs are categorized into three types. In the first type, which consists of bzip2 and gzip, power consumption monotonously increases as the number of SDUs increases. This means that the static power dominates. The second type consists of parser and vpr. In this type, in contrast, power consumption decreases as the number of SDUs increases. This means that the dynamic power is dominant.

Figure 5. Relative power consumption (50%-scenario, x2.0).

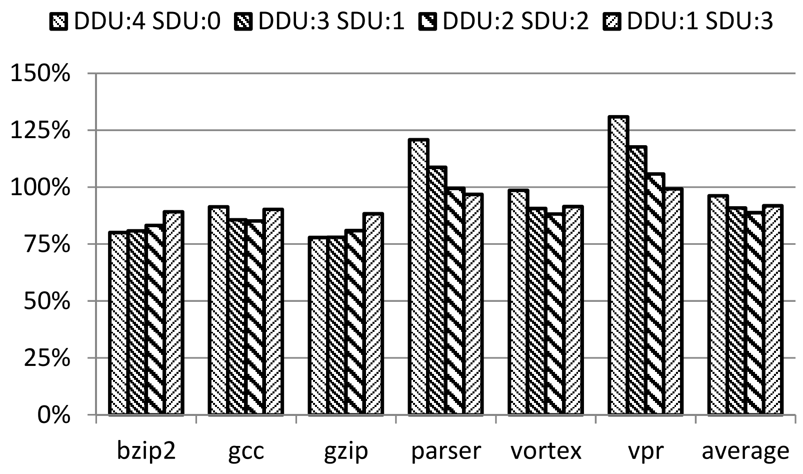
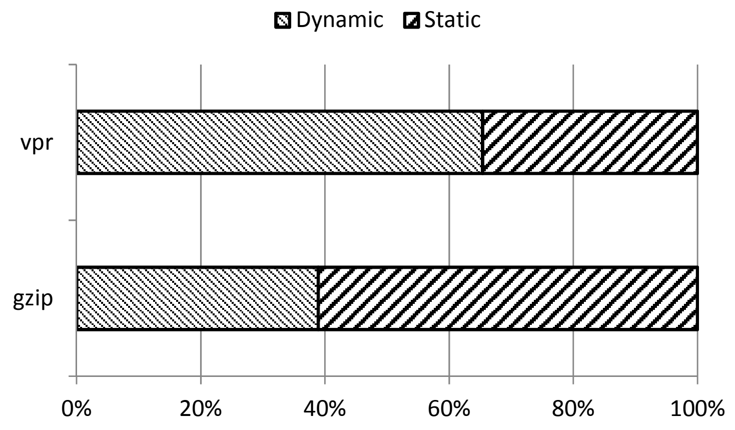


Figure 6 presents the breakdown of the baseline power consumption. The left part indicates the dynamic power and the right one indicates the static power. The results confirm the above discussions. Static power dominates in the first type and dynamic power dominates in the second type. The third type, which consists of gcc and vortex, is located between the two types. Unfortunately, in the cases of parser and vpr, the proposed technique does not achieve power reduction. However, on average, the total power consumption is reduced by 3.76%, 9.13%, 11.2%, and 8.20%, respectively, for the cases where four, three, two and one SDU(s) are replaced by DDUs.

Figure 6. Breakdown of power consumption.



Results shown in Figures 7 and 8 assume the same scenario used in Figure 5, except that the ratios of the dynamic power of DDU over SDU are 1.5 and 3.0, respectively. As we have already seen in Figure 4, a smaller ratio of dynamic power is preferable. When the dynamic power of DDU is 1.5 times larger than that of SDU, the total power consumption is always reduced, regardless of the configuration of the functional units. On the contrary, when the dynamic power of DDU is three times larger than that of SDU, the total power consumption is reduced only in the case of DDU:1 SDU:3.

Figure 7. Relative power consumption (50%-scenario, ×1.5).

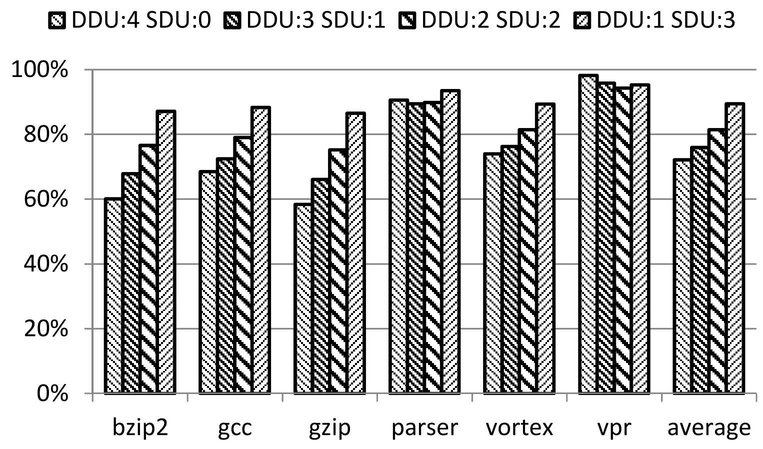


Figure 8. Relative power consumption (50%-scenario, ×3.0).

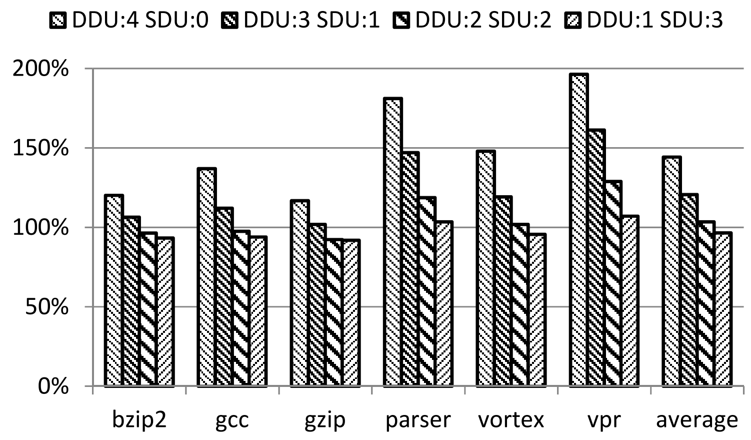
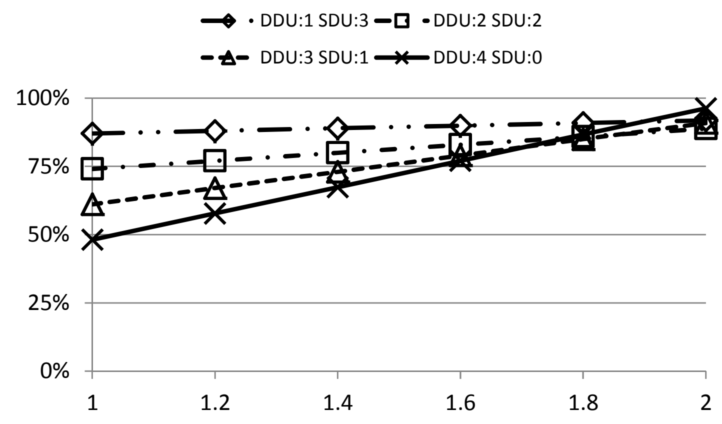


Figure 9 presents the relative power consumption when we vary the ratio of the dynamic power of DDU over that of the SDU between one and two in the case of the 50% scenario. The horizontal axis indicates the ratio, and the vertical axis indicates the relative power consumption. The four lines from top to bottom (see the leftmost) present the cases where one, two, three and four SDU(s) are replaced by DDUs, as shown in the figure legend. As expected, the smaller the ratio, the larger is the power reduction.

Figure 9. Sensitivity to dynamic power of DDU (50%-scenario).



Figures 10 and 11 present the relative power consumption in the case where the dynamic power of DDU is twice as large as that of SDU and the ratio of the static power over the total power is 40% and 30%, respectively. As the ratio of the static power over the total power of the baseline becomes smaller, almost all benchmarks are categorized in the second type and the benefit from the proposed technique is reduced. This is as expected because the dynamic power dominates in these cases. To accommodate such cases, we propose the use of heterogeneous functional units in order to reduce the static power consumption.

Figure 10. Relative power consumption (40%-scenario, ×2.0).

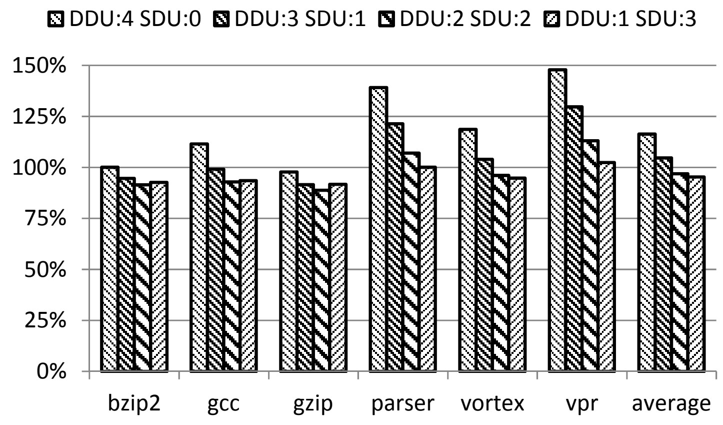
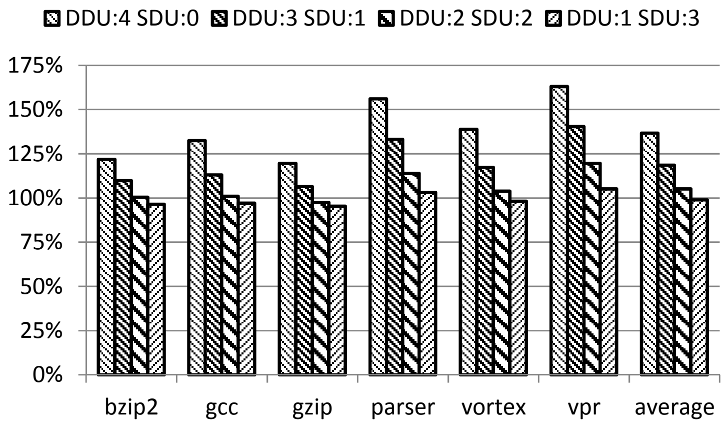


Figure 11. Relative power consumption (30%-scenario, ×2.0).



Figures 12 and 13 present the relative power consumption when the ratio of the dynamic power of DDU is varied over that of the SDU between one and two in the cases of 40% and 30% scenarios, respectively. The layouts of these figures are the same as that of Figure 9. Even the trends are almost the same as that shown in Figure 9. As expected, the smaller the ratio, the larger is the power reduction. The difference is that the effectiveness is diminished as the contribution of static power to total power is reduced. Because it is predicted that the static contribution will become larger for upcoming processors, our proposal will become more effective in the near future.

Figure 12. Sensitivity to dynamic power of DDU (40%-scenario).

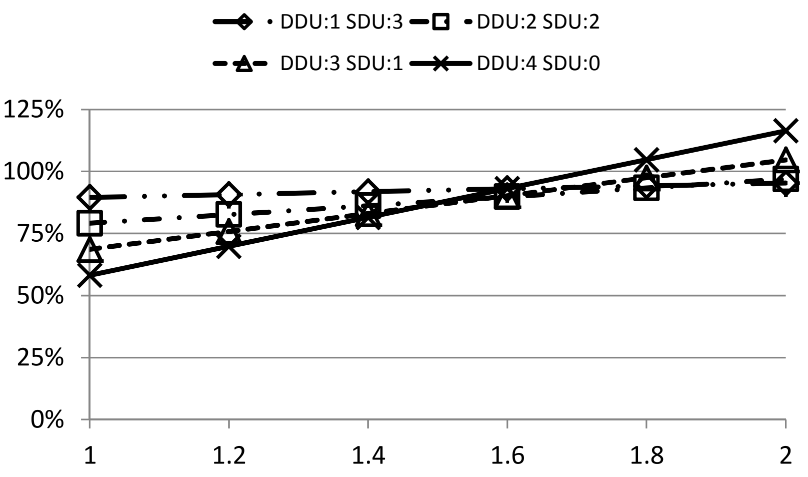
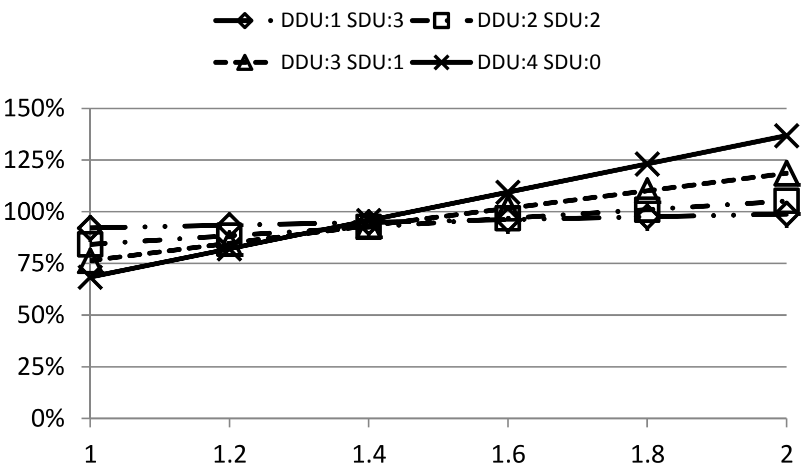


Figure 13. Sensitivity to dynamic power of DDU (30%-scenario).



6. Related Works

Dropsho *et al.* [18] claimed that the static power consumption in functional units will become serious in the near future. Following the study, we are studying how to reduce static power consumption in functional units.

Optimizing the threshold voltage is a popular technique to reduce static power consumption [4]. Utilizing two kinds of threshold voltage is typical in this technique. The cost incurred by the conventional technique is in performance degradation, because gate delay is increased as the threshold voltage is increased for static power reduction. In contrast, our proposal maintains performance.

This can be possible at the cost of an increase in dynamic power consumption. However, the total power consumption can be reduced by utilizing power heterogeneous functional units.

Ikebuchi *et al.* [5] investigated the power gating technique to reduce static power. When we adopt this technique, we had to consider BET, which is a time period in which the power saving by cutting the power supply is equal to the total overhead. The period of power gating should be longer than BET, which is considerably long. They propose to predict when the functional units are inactive and turn off the power supply only when the period is predicted to be longer than BET. Our proposal can collaborate with the technique because these two techniques exploit completely different mechanisms.

Big.LITTLE [8,10] and 4-plus-1 [9] exploit the heterogeneity among the processor cores for power reduction. In contrast, our proposal exploits the heterogeneity among the functional units within a single core. Hence, these two power reduction techniques can likewise orchestrate with each other.

Table 3 summarizes the merits and demerits of the above techniques. We do not intend to claim that our proposed design method is superior to the others; instead we propose to combine them to compensate for the shortcomings of each technique.

Table 3. Comparison of low-power techniques.

Techniques	Merit	Demerit
Higher V_{th} [4]	No additional circuit	Performance loss, No adaptability
Power gating [5]	Adaptability, Fine grain	Additional circuit for prediction
Heterogeneous core [8–10]	Adaptability	Coarse grain, Large chip area
Proposed Power Heterogeneous Functional Units	No performance loss, Fine grain	Limited adaptability, Dual power networks

7. Conclusions

As semiconductor technologies have improved, good techniques to reduce the leakage power consumption of functional units are strongly required. This paper proposed to utilize the power heterogeneous functional units in order to reduce the total power consumption of microprocessor cores. The proposal exploits two kinds of functional units: one is made of leaky transistors and the other is made of leakless transistors. While the latter consumes a significant amount of dynamic power, we can reduce the total power consumption by optimizing the heterogeneous functional units. From the detailed simulations, a power reduction of 11.2% is achieved if two out of four leaky functional units are replaced by leakless ones in the situation in which the static power occupies half the total power.

Acknowledgments

The authors thank Takanori Hayashida for his support in building the simulation environment and Shinya Takahashi for his technical advice. This study was supported in part by JSPS Grant-in-Aid for Challenging Exploratory Research (#23650026) and by funding from Central Research Institute of Fukuoka University, and is currently by JSPS Grant-in-Aid for Scientific Research(C) (#26330075).

Author Contributions

Toshinori Sato contributed to the concept and the design of this study, analyzed and interpreted the experimental data, and wrote the manuscript. Yoshimi Shibata built the simulation environment and collected the experimental data.

Conflicts of Interest

The authors declare no conflict of interest.

References and Notes

1. Krishnamurthy, R.K.; Alvandpour, A.; Mathew, S.; Anders, M.; De, V.; Borkar, S. High performance, low-power, and leakage-tolerance challenges for sub-70 nm microprocessor circuits. In Proceedings of European Solid-State Circuits Conference, Florence, Italy, 24–26 September 2002; pp. 315–321.
2. Krishnamurthy, R.K.; Mathew, S.K.; Anders, M.A.; Hsu, S.K.; Kaul, H.; Borkar, S. High-performance and low-voltage challenges for sub-45 nm microprocessor circuits. In Proceedings of IEEE International Conference on ASIC, Shanghai, China, 24–27 October 2005; pp. 24–27.
3. Iwai, H. Roadmap for 22 nm and Beyond. *Microelectro. Eng.* **2009**, *7–9*, 1520–1528.
4. Kao, J.; Narendra, S.; Chandrakasan, A. Subthreshold leakage modeling and reduction techniques. In Proceedings of IEEE/ACM International Conference on Computer-Aided Design, San Jose, CA, USA, 10–14 November 2002; pp. 141–148.
5. Ikebuchi, D.; Seki, N.; Kojima, Y.; Kamata, M.; Zhao, L.; Amano, H.; Shirai, T.; Koyama, S.; Hashida, T.; Umahashi, Y.; *et al*. Geyser-1: A MIPS R3000 CPU core with fine grain runtime power gating. In Proceedings of IEEE Asian Solid-State Circuits Conference, Taipei, Taiwan, 16–18 November 2009; pp. 281–284.
6. Matsunaga, S.; Hayakawa, J.; Ikeda, S.; Miura, K.; Endoh, T.; Ohno, H.; Hanyu, T. MTJ-based nonvolatile logic-in-memory circuit, future prospects and issues. In Proceedings of Design, Automation & Test in Europe Conference & Exhibition, Nice, France, 20–24 April 2009; pp. 433–435.
7. Abthony, S. Apple's A7 Cyclone CPU Detailed: A Desktop Class Chip That Has More in Common with Haswell than Krait. Available online: http://www.extremetech.com (accessed on 31 March 2014).
8. ARM Ltd. *Big.LITTLE Processing with ARM CortexTM-A15 & Cortex-A7*; White paper; ARM Ltd.: San Jose, CA, USA, 2011.
9. NVIDIA Corporation. *NVIDIA Tegra K1—A New Era in Mobile Computing*; White paper; NVIDIA Corporation: Santa Clara, CA, USA, 2011.
10. Jeff, B. *Advances in big.LITTLE Technology for Power and Energy Savings*; White paper; ARM Ltd.: San Jose, CA, USA, 2012.

11. Powell, M.D.; Schuchman, E.; Vijaykumar, T.N. Balancing resource utilization to mitigate power density in processor pipelines. In Proceedings of IEEE/ACM International Symposium on Microarchitecture, Barcelona, Spain, 12–16 November 2005; pp. 294–304.
12. Matsumura, T.; Ishihara, T.; Yasuura, H. Simultaneous optimization of memory configuration and code allocation for low power embedded systems. In Proceedings of ACM Great Lakes Symposium on VLSI, Orlando, FL, USA, 4–6 May 2008; pp. 403–406.
13. Usami, K.; Horowitz, M. Clustered voltage scaling technique for low-power design. In Proceedings of ACM International Symposium on Low Power Design, Dana Pt., CA, USA, 23–26 April 1995; pp. 3–8.
14. Austin, T.; Larson, E.; Ernst, D. SimpleScalar: An infrastructure for computer system modeling. *IEEE Comput*. **2002**, *2*, 59–67.
15. Brooks, D.; Tiwari, V.; Martonosi, M. Wattch: A framework for architectural-level power analysis and optimizations. In Proceedings of IEEE/ACM International Symposium on Computer Architecture, Vancouver, BC, Canada, 14 June 2000; pp. 83–94.
16. Sheng, L.; Ahn, J.H.; Strong, R.D.; Brockman, J.B.; Tullsen, D.M.; Jouppi, N.P. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In Proceedings of IEEE/ACM International Symposium on Microarchitecture, New York, NY, USA, 12–16 December 2009; pp. 469–480.
17. Interviews with LSI designers of high-performance and mobile processors.
18. Dropsho, S.; Kursun, V.; Albonesi, D.H.; Dwarkadas, S.; Friedman, E.G. Managing static leakage energy in microprocessor functional units. In Proceedings of IEEE/ACM International Symposium on Microarchitecture, Istanbul, Turkey, 18–22 November 2002; pp. 321–332.