The Potential of Temperature-Aware Configurable Cache on Energy Reduction

Noori, Hamid
Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University

Goudarzi, Maziar
System LSI Research Center, Kyushu University

Inoue, Koji
Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University

Murakami, Kazuaki
Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University

出版情報：情報処理学会研究報告, 2007-ARC-173. 2007 (55), pp.13-18, 2007-05. 情報処理学会
バージョン：

The Potential of Temperature-Aware Configurable Cache on Energy Reduction

Hamid Noori[†] Maziar Goudarzi[††] Koji Inoue[†] and Kazuaki Murakami[†]

[†]Department of Informatics, Graduate School of Information Science and Electrical Engineering, Kyushu University,
[††] System LSI Research Center, Kyushu University

E-mail: noori@c.csce.kyushu-u.ac.jp, goudarzi@slrc.kyushu-u.ac.jp, {inoue, murakami}@i.kyushu-u.ac.jp,

Abstract Active power used to be the primary contributor to total power dissipation of CMOS designs, but with the technology scaling, the share of leakage in total power consumption of digital systems continues to grow. Moreover, temperature is another factor that exponentially increases the leakage current. In this paper, we show the effects of temperature and technology nodes on the optimal (minimum-energy-consuming) cache configuration for low energy embedded systems. We show that a temperature-aware configurable cache is an effective way to save energy in finer technologies when the embedded system may be used in different temperatures. Our results show that using a temperature-aware configurable cache, up to 66% energy can be saved with only 1% performance penalty for instruction cache and 74% energy saving with 4.7% performance loss for data cache.

Keyword Low Energy, Embedded Systems, Leakage Current, Temperature-Aware Design.

1. Introduction

Leakage (also known as static) power is becoming the dominant contributor to the power consumption of CMOS integrated circuit. Unlike dynamic power, which depends on the activity in the circuit, leakage exists as long as power supply is applied to the circuit even if there is no activity. Consequently, in general, bigger circuits dissipate higher leakage.

Figs. 1 and 2 show the dynamic energy and leakage power for various cache sizes in three different technologies: 180nm, 100nm and 70nm. The values shown in these figures are results of running CACTI tool [1] (for temperature 110°C), which is widely used for cache power-estimation, and correspond to different sizes of a 4-way set-associative cache with 16-byte block size. The figures clearly state that dynamic and static powers are exchanging their roles. While the dynamic power is decreasing in finer technologies, the static power is increasing. It is important to note that the rate at which static power increases is much bigger than the rate at which dynamic power decreases. For the given example, the dynamic energy in 180nm is almost four times of 100nm and 9 times of 70nm, however the static power in 70nm is 400 times of 180nm and five times of 100nm.

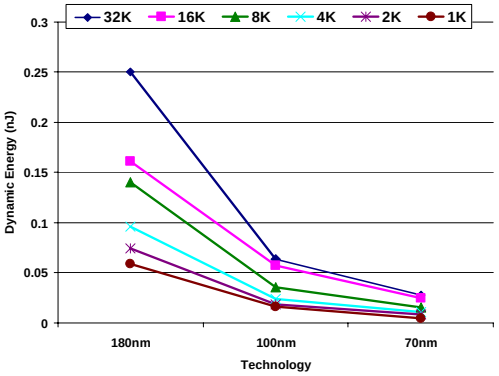

Fig. 1. Dynamic energy per access for different

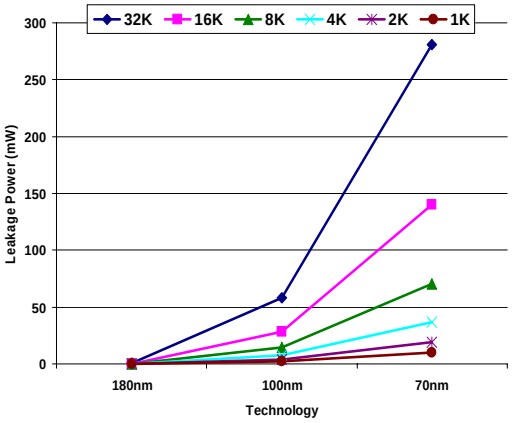

Fig. 2. Leakage power for different cache sizes

Energy consumption is an important issue for battery powered embedded systems. On-chip cache consumes almost half of a microprocessor total energy [2][3]. Energy efficient cache architecture design is thus a critical issue in the design of processor-based embedded systems. Larger cache size may improve the hit rate for some programs, at the expense of more power consumed to fetch (dynamic power) and to hold (static power) the data. Performance-oriented applications benefit from a large cache size. Energy-oriented applications however want the cache size such that the energy savings from higher hit rate outweigh the energy increase from more leakage.

Temperature is a parameter that changes for different reasons. The sources of temperature variation can be the environment (the location or time) or the system itself (the heat produced when executing applications). Temperature has an exponential effect on the leakage power [1]. Fig. 3 shows the leakage power of a 32KB cache for six different temperatures in 180nm, 100nm and 70nm technologies. The data were obtained using CACTI 4.1 [1]. As it is seen, the leakage current for 180nm is negligible compared to 100nm and 70nm, however in finer processes, not only it is higher than before, but also it increases significantly with temperature at an increasingly higher rate. Consequently, it is quite expected that at some point in future, an embedded system (e.g. a cell phone) performance and power would be much different in a ventilated room versus inside a car parked in direct sunshine in a hot summer.

In the given example (Fig. 3), although for 180nm the leakage power of the cache in 100°C is 400.13 times of 0°C, since the leakage absolute value in 0°C is very small, the effect of temperature is not considerable. In 100nm and 70nm technologies, the leakage power of the cache in 100°C is 31.4 and 19.86 times bigger than 0°C leakage power respectively, however since the leakage power is much bigger for 100nm and 70nm compared to 180nm (1025.23 times and 7581.59 times for 100nm and 70nm, respectively) the effect of temperature becomes much more significant in finer technologies.

In this paper, we show that when the temperature changes, a new cache size (or configuration) may need to be selected, due to the increase of leakage current, if the minimum energy consumption is a must [4][5]. Our results show that using a temperature-aware configurable cache, up to 66% energy can be saved with only 1% performance penalty for instruction cache and 74% energy saving with 4.7% performance loss for data cache. As mentioned, the effect of temperature is more considerable in finer technologies, while it used to be trivial in older technologies. Therefore for new technologies, we show that a cache configuration selected for minimum energy–consumption at a cold environment may not work efficiently when used in a warmer environment (up to 74% higher energy than the best choice at the latter temperature) and vice versa. We conduct experiments at six different temperatures and three different technologies: 180nm, 100nm and 70nm. Thus, at the same time we also study the effect on nanometer-scale technologies on the cache configuration selection.

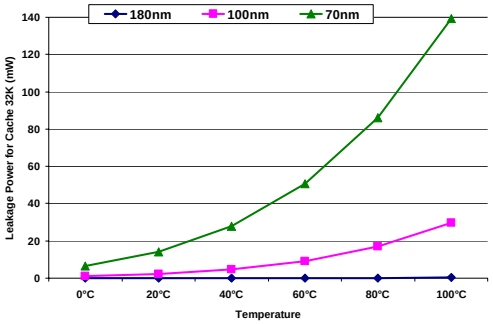

Fig. 3 The impact of temperature on the leakage

2. Problem Formulation

2.1. Energy Evaluation

The power consumption of CMOS circuits includes dynamic, static, and short circuit power. The short circuit power consumption is much smaller than dynamic and static power, and is negligible. Dynamic power in cache memories attributes to signal transitions in activated bit- and word-lines, sense-amplifiers, comparators, and selectors during reads/writes, while static power is due to the total amount of leakage current through inactive or OFF transistors. Energy equals power times time. The dynamic energy consumed per access is a sum of the energy spent on searching within the cache, an extra energy required for handling the writes and energy consumed by block replacement on cache miss.

Dynamic energy per cache access equals the dynamic power of all the circuits in the cache multiplied by the time per cache access; while the static energy of a cache equals static power multiplied by the time.

Dynamic energy consumption causes most of the total energy dissipation in micrometer-scale technologies and lower temperatures, but static energy dissipation will contribute an increasingly larger portion of total energy dissipation in nanometer-scale technologies and higher temperatures. We consider both types of energies.

Furthermore, we should not disregard energy consumption due to accessing off-chip memory, since fetching instructions and data from off-chip memory is energy expensive because of the high off-chip capacitance and large off-chip memory storage. Additionally, when accessing the off-chip memory, the processor may stall while waiting for the instruction and/or data, and such waiting still consumes some energy. Thus, we calculate the total energy due to memory accesses using Equation 1. In our evaluation the energy is a function of configuration

of cache (*C*), temperature (*Temp*) and technology (*Tech*). Other parameters are assumed to be constant.

$$energy\_memory(C, Temp, Tech) = energy\_dynamic(C, Tech) + energy\_static(C, Temp, Tech) \quad (1)$$

where

$$energy\_dynamic(C, Tech) = cache\_accesses(C) * energy\_cache\_access(C, Tech) + cache\_misses(C) * energy\_miss(C,Tech) \quad (2)$$

$$energy\_miss(C, Tech) = energy\_off\_chip\_access + energy\_cache\_block\_refill(C, Tech) \quad (3)$$

$$energy\_static(C, Temp, Tech) = executed\_clock\_cycles(C) * clock\_period * leakage\_power(C, Temp, Tech) \quad (4)$$

The values of *cache_accesses*, *cache_misses* and *executed_clock_cycles* are computed by running SimpleScalar [6] for the given applications with the desired cache configuration.

The *energy_cache_access*, *energy_cache_block_refill*, and *leakage_power* are respectively the energy for accessing the cache, cache block (line) refilling after a cache miss, and the leakage power (static power) of a given cache and are computed using CACTI.

The *energy_off_chip_access* is the energy of accessing off-chip memory when there is a miss, and the *energy_uP_stall* is the energy consumed when the processor is stalled waiting for the memory system to provide the missed instruction or data. According to the explanations and experiments done in [2], we assumed:

$$energy\_off\_chip\_access + energy\_uP\_stall(Tech) = 20\ nJ \quad (5)$$

2.2. Problem Definition

In the study that we report in this paper, we assume that the embedded application, the processor and the target technology used are all invariant and only the temperature and configuration of the cache is changing. This reflects a scenario where a processor-based embedded system is used in different environments with various temperatures. So in our experiments, the optimization problem is defined as follows:

"For a given application, processor architecture, and technology find the cache configuration that results in minimum energy consumption in different temperatures (i.e. minimizes Equation 1 for a given temperature) over the entire application run."

3. Experimental Results

We use applications from Mibench [7] benchmark suite. As mentioned, SimpleScalar and CACTI (version 4.1) are used as our simulation tool and power modeling tool, respectively. The cache hit is assumed to take one clock cycle and cache miss 100 cycles. The clock frequency of the single-issue base processor is assumed to be 200 MHz. Our experiments are done for six different temperatures: 0°C, 20°C, 40°C, 60°C, 80°C and 100°C, and three different technologies: 180nm, 100nm and 70nm.

In this section it is assumed that a temperature-aware configurable cache is available with configurability in size, associativity (number of ways) and line size. We want to study the effectiveness of such a configurable cache in reducing the energy impact of the changes in the temperature. In other words, we want to show how a temperature-aware configurable cache can help to reduce energy consumption. The cache size can vary from 64KB, to 1KB, the associativity can be 4, 2 and 1, and finally the line size can be 32, 16 and 8 bytes.

In these experiments, for each application we ran Simplescalar for all the configurations once for the instruction cache and once again for the data cache (7x3x3x2 = 126 simulations per application). Then we extracted the data for dynamic energy (note that the dynamic energy is the same for different temperatures) and static energy for all the 63 cache configurations for six different temperatures (0°C, 20°C, 40°C, 60°C, 80°C, 100°C) and three different technologies (180nm, 100nm and 70nm) using CACTI (63 x 6 x 3 = 1134 cases). Using these data and equations of Section 3.1, for each application we find the cache configurations that minimize Equation 1 for different temperatures and technologies.

Table 1 and 2 respectively show the optimal instruction and data cache configurations which result in minimum energy for six different temperatures in three technologies. For the configurable instruction cache, the data cache is fixed and assumed to be a 4-way, 32KB with 32 bytes line size and for the configurable data cache, the instruction cache is assumed to be 2-way, 32KB with 32 bytes line size. In the triple used for presenting the cache configuration in the tables, the first number shows the cache size in kilo bytes, the second number shows the line size in bytes and the last one specifies the number of ways.

According to the results of Table 1 for 180nm, the temperature does not have much effect on the energy consumption of the instruction cache and the cache configuration need not to be changed. In Table 2 only for two examined applications (*basicmath* and *qsort* at 100°C) the cache need to be reconfigured. However as the technology is getting finer, more reconfigurations are required to have a cache with minimum energy. Since the accesses to data cache is less than instruction cache, the leakage has more important role in the total energy, thus more reconfigurations are required as the temperature is changing. According to the results, as the technology is becoming finer and the temperature is increasing (larger leakage power) the optimal cache is moving to smaller sizes.

Fig 4 and 5 show how a temperature-aware configurable instruction cache can help to reduce the energy consumption for 100nm and 70nm, respectively. These figures illustrate the percentage of energy saving obtained by reconfiguring the cache to the new optimal configuration at the new temperature compared to the energy of the cache with minimum energy at 0°C now operating at the new temperature. For example, in Fig. 4 for *basicmath*, by reconfiguring the instruction cache from (32KB, 16, 1) (optimal configuration at 0°C, Table 1) to (8KB, 16, 4) (optimal configuration at 100°C, Table 1) 66.65% energy is saved. The saved energy for each application in different temperatures is calculated using Equation 6 below. In this equation, the *energy_cache0_tempN* is the energy consumption of the cache configuration with minimum energy consumption for 0°C when the temperature is *N*°C; the *energy_cacheN* is the energy of the cache configuration with minimum energy for temperature *N*°C.

Energy Saving =

$$\frac{energy\_cache0\_tempN - energy\_cacheN}{energy\_cache0\_tempN} * 100 \quad (6)$$

The configurable cache seems to be more effective in 100nm than 70nm. The reason is large leakage power of 70nm. Since the leakage power is much higher in 70nm the initial cache configuration with minimum energy is already small. Therefore when the temperature increases and new configurations are selected for the cache, the effect is smaller compared to 100nm.

Fig. 6 and 7 show the *performance penalty* (the percentage of increase in execution time) when the instruction cache is reconfigured for the lowest energy. The *performance penalty* is calculated using the following equation (Equation 7):

Performance Penalty =

$$\frac{exec\_time\_cacheN - exec\_time\_cache0}{exec\_time\_cache0} * 100 \quad (7)$$

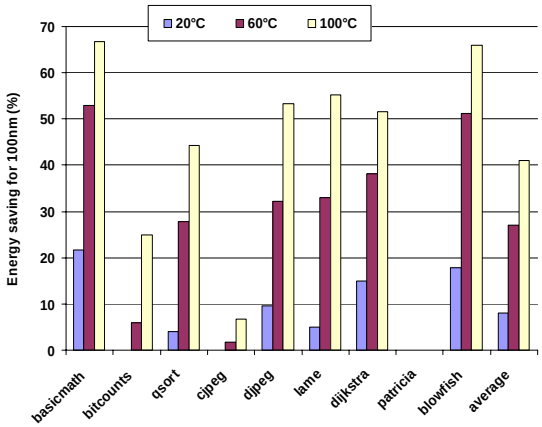

Fig. 4. The effect of configurable cache on energy saving in 100nm (Instruction Cache)

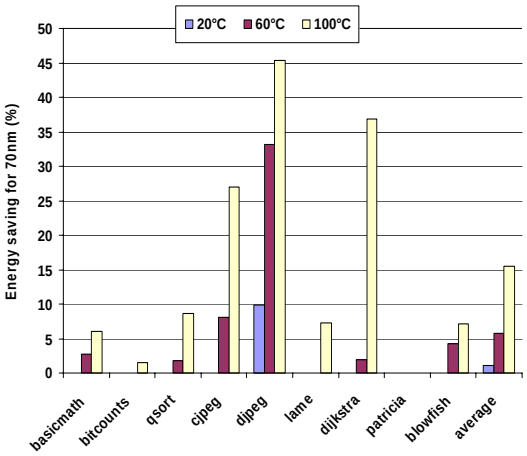

Fig. 5. The effect of configurable cache on energy saving in 70nm (Instruction Cache)

where the *exec_time_cache0* is the execution time of cache configuration with minimum energy for 0°C and *exec_time_cacheN* is the execution time of cache configuration with minimum energy for *N*°C. For applications such as *cjpeg* and *dijkstra*, the new cache configurations not only help to reduce the energy consumption, but also improve the performance. For 100nm by 2% increase in execution time 41% energy is saved, in average, however in 70nm by 16% increase in execution time 16% energy is saved, in average. This shows that as the technology is becoming finer it will be

much harder to find the optimal cache size in terms of both performance and energy consumption.

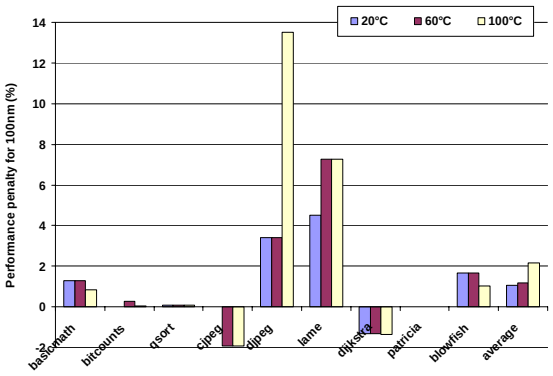

Fig. 6. The effect of configurable instruction cache on execution time in 100nm

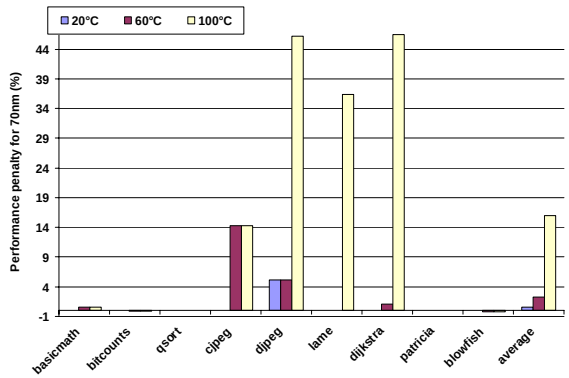

Fig. 7. The effect of configurable instruction cache on execution time in 70nm

Fig. 8 and 9 show the *energy saving* when the data cache is a temperature-aware configurable cache and Fig. 10 and 11 show their corresponding *performance penalty* for 100nm and 70nm respectively. As mentioned before, since the accesses to data cache is less than instruction cache the leakage power has more contribution in the total energy of the data cache. According to the results using a configurable cache in 100nm, 47% energy can be saved in average when temperature changes from 0°C to 100°C. The penalty in the performance is around 8% in average. For 70nm the average energy saving is around 33% while the performance penalty is around 20%. These results show that the temperature-aware configurable cache is more effective for data cache compared to instruction cache. The results again confirm the idea that in new technologies finding the optimal cache size in terms of both performance and energy consumption will be more difficult.

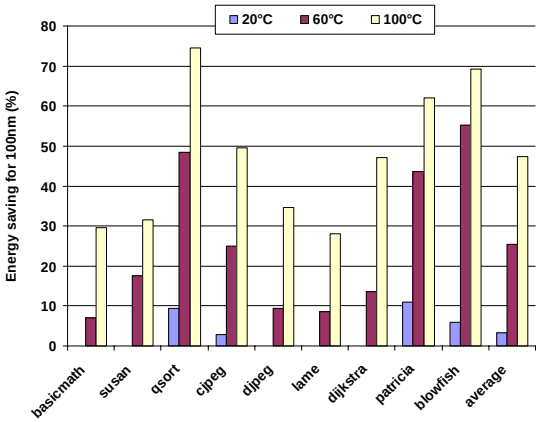

Fig. 8. The effect of configurable data cache on energy saving in 100nm

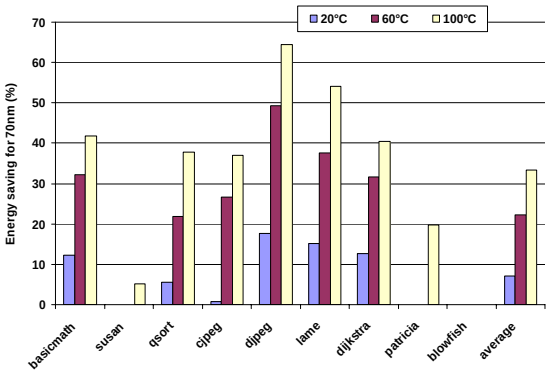

Fig. 9. The effect of configurable data cache on energy saving in 70nm

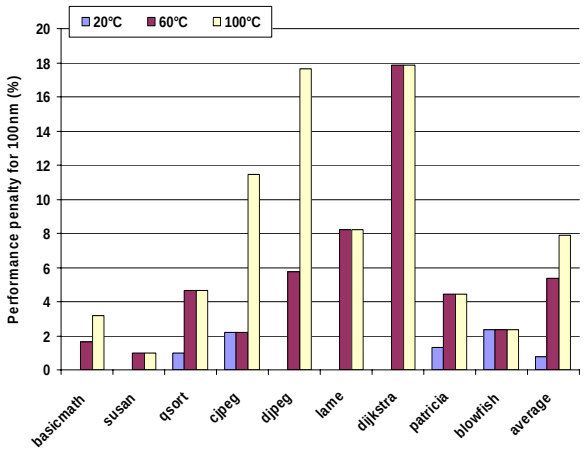

Fig. 10. The effect of configurable data cache on execution time in 100nm

4. Conclusions

Due to the increase of leakage effect in finer technologies and higher temperatures the smaller caches will be more energy efficient for future low energy systems

Since the smaller caches are more suitable for low energy systems in finer technologies and higher temperatures, finding an optimal cache configuration that simultaneously optimizes performance and energy is

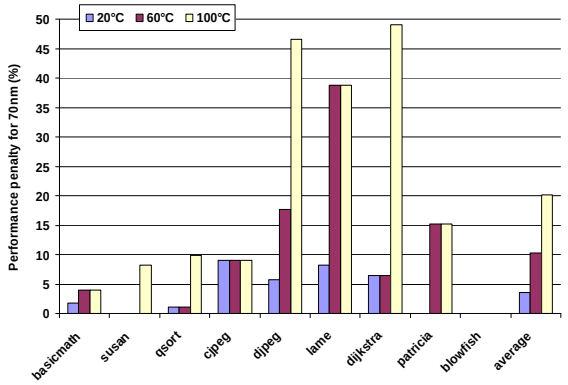

Fig. 11. The effect of configurable data cache on execution time in 70nm

increasingly more difficult in future, specially at high temperatures.

Acknowledgement

This research was supported in part by the Grant-in-Aid for Encouragement of Young Scientists (A), 17680005 and the 21st Century COE Program.

Reference

[1] D. Tarjan, Sh. Thoziyoor, N. P. Jouppi, Cacti 4.0, HP Laboratories, Technical Report, 2006.

[2] C. Zhang, F. Vahid, and W. Najjar, "A Highly Configurable Cache Architecture for Embedded Systems," ACM Transactions on Embedded Computing Systems, Vol. 4, No. 2, May 2005.

[3] V. G. Moshnyaga and K. Inoue, "Low-Power Cache Design", in Low-Power Electronics Design, CRC Press, 2005.

[4] H. Noori, M. Goudarzi, K. Inoue, and K. Murakami, "The Effect of Temperature on Cache Size Tuning for Low Energy Embedded Systems", The 17th edition of ACM Great Lakes Symposium on VLSI (GLSVLSI), 2007.

[5] H. Noori, M. Goudarzi, K. Inoue, and K. Murakami, "The Effect of Nanometer-Scale Technologies on the Cache Size Selection for Low Energy Embedded Systems", International Conference on Embedded Systems & Applications (ESA'07), 2007.

[6] SimpleScalar http://www.simplescalar.com/

[7] Mibench http://www.eecs.umich.edu/mibench/

Table 1: Instruction cache configurations with minimum energy. Values shown are respectively cache size, line-size in bytes, and the number of ways.

Application		basimath	bitcounts	qsort	cjpeg	djpeg	lame	dijkstra	patricia	blowfish
180nm	0°C	32K, 16, 1	2K, 16, 1	32K, 8, 2	8K, 16, 2	16K, 16, 1	16K, 32, 2	16K, 16, 1	32K, 8, 4	32K, 16, 1
	20°C	32K, 16, 1	2K, 16, 1	32K, 8, 2	8K, 16, 2	16K, 16, 1	16K, 32, 2	16K, 16, 1	32K, 8, 4	32K, 16, 1
	40°C	32K, 16, 1	2K, 16, 1	32K, 8, 2	8K, 16, 2	16K, 16, 1	16K, 32, 2	16K, 16, 1	32K, 8, 4	32K, 16, 1
	60°C	32K, 16, 1	2K, 16, 1	32K, 8, 2	8K, 16, 2	16K, 16, 1	16K, 32, 2	16K, 16, 1	32K, 8, 4	32K, 16, 1
	80°C	32K, 16, 1	2K, 16, 1	32K, 8, 2	8K, 16, 2	16K, 16, 1	16K, 32, 2	16K, 16, 1	32K, 8, 4	32K, 16, 1
	100°C	32K, 16, 1	2K, 16, 1	32K, 8, 2	8K, 16, 2	16K, 16, 1	16K, 32, 2	16K, 16, 1	32K, 8, 4	32K, 16, 1
100nm	0°C	32K, 16, 1	2K, 16, 1	32K, 16, 2	8K, 16, 2	32K, 32, 1	32K, 32, 4	16K, 8, 1	32K, 32, 4	32K, 16, 1
	20°C	8K, 8, 4	2K, 16, 1	16K, 8, 4	8K, 16, 2	16K, 32, 1	16K, 32, 2	8K, 8, 4	32K, 32, 4	8K, 8, 4
	40°C	8K, 8, 4	2K, 32, 1	16K, 8, 4	8K, 16, 2	16K, 32, 1	8K, 32, 4	8K, 8, 4	32K, 32, 4	8K, 8, 4
	60°C	8K, 8, 4	1K, 8, 2	16K, 32, 4	8K, 32, 2	16K, 32, 1	8K, 32, 4	8K, 8, 4	32K, 32, 4	8K, 8, 4
	80°C	8K, 16, 4	1K, 16, 2	16K, 32, 4	8K, 32, 2	8K, 32, 2	8K, 32, 4	8K, 16, 4	32K, 32, 4	8K, 16, 4
	100°C	8K, 16, 4	1K, 16, 2	16K, 32, 4	8K, 32, 2	8K, 32, 2	8K, 32, 4	8K, 16, 4	32K, 32, 4	8K, 16, 4
70nm	0°C	8K, 16, 4	1K, 16, 2	16K, 16, 4	8K, 32, 2	16K, 32, 1	8K, 32, 4	8K, 16, 4	32K, 32, 4	8K, 16, 4
	20°C	8K, 16, 4	1K, 16, 2	16K, 16, 4	8K, 32, 2	8K, 32, 4	8K, 32, 4	8K, 16, 4	32K, 32, 4	8K, 16, 4
	40°C	8K, 16, 4	1K, 16, 2	16K, 16, 4	8K, 32, 2	8K, 32, 4	8K, 32, 4	8K, 16, 4	32K, 32, 4	8K, 32, 4
	60°C	8K, 32, 4	1K, 32, 2	16K, 32, 4	4K, 32, 4	8K, 32, 4	8K, 32, 4	8K, 32, 4	32K, 32, 4	8K, 32, 4
	80°C	8K, 32, 4	1K, 32, 2	16K, 32, 4	4K, 32, 4	8K, 32, 4	8K, 32, 4	2K, 32, 2	32K, 32, 4	8K, 32, 4
	100°C	8K, 32, 4	1K, 32, 2	16K, 32, 4	4K, 32, 4	4K, 32, 1	4K, 32, 2	2K, 32, 2	32K, 32, 4	8K, 32, 4

Table 2: Data cache configurations with minimum energy

Application		basimath	susan	qsort	cjpeg	djpeg	lame	dijkstra	patricia	blowfish
180nm	0°C	8K, 8, 1	2K, 16, 4	64K, 32, 1	32K, 32, 1	32K, 16, 2	32K, 32, 1	32K, 32, 1	16K, 32, 2	32K, 8, 2
	20°C	8K, 8, 1	2K, 16, 4	64K, 32, 1	32K, 32, 1	32K, 16, 2	32K, 32, 1	32K, 32, 1	16K, 32, 2	32K, 8, 2
	40°C	8K, 8, 1	2K, 16, 4	64K, 32, 1	32K, 32, 1	32K, 16, 2	32K, 32, 1	32K, 32, 1	16K, 32, 2	32K, 8, 2
	60°C	8K, 8, 1	2K, 16, 4	64K, 32, 1	32K, 32, 1	32K, 16, 2	32K, 32, 1	32K, 32, 1	16K, 32, 2	32K, 8, 2
	80°C	8K, 8, 1	2K, 16, 4	64K, 32, 1	32K, 32, 1	32K, 16, 2	32K, 32, 1	32K, 32, 1	16K, 32, 2	32K, 8, 2
	100°C	4K, 32, 1	2K, 16, 4	32K, 32, 1	32K, 32, 1	32K, 16, 2	32K, 32, 1	32K, 32, 1	16K, 32, 2	32K, 8, 2
100nm	0°C	4K, 16, 2	4K, 32, 4	32K, 32, 2	32K, 32, 2	32K, 32, 4	32K, 32, 4	32K, 32, 2	32K, 32, 4	32K, 16, 2
	20°C	4K, 16, 2	4K, 32, 4	16K, 32, 2	16K, 32, 2	32K, 32, 4	32K, 32, 4	32K, 32, 2	16K, 32, 4	8K, 32, 4
	40°C	4K, 16, 2	2K, 32, 4	4K, 32, 4	16K, 32, 2	32K, 32, 4	32K, 32, 4	32K, 32, 2	8K, 32, 4	8K, 32, 4
	60°C	2K, 8, 2	2K, 32, 4	2K, 32, 4	16K, 32, 2	16K, 32, 4	16K, 32, 4	8K, 32, 4	8K, 32, 4	8K, 32, 4
	80°C	2K, 8, 2	2K, 32, 4	2K, 32, 4	8K, 32, 2	16K, 32, 4	16K, 32, 4	8K, 32, 4	8K, 32, 4	8K, 32, 4
	100°C	2K, 16, 1	2K, 32, 4	2K, 32, 4	8K, 32, 2	8K, 32, 4	16K, 32, 4	8K, 32, 4	8K, 32, 4	8K, 32, 4
70nm	0°C	4K, 32, 2	2K, 32, 4	4K, 32, 4	16K, 32, 2	32K, 32, 4	32K, 32, 4	16K, 32, 2	8K, 32, 4	8K, 32, 4
	20°C	2K, 16, 4	2K, 32, 4	2K, 32, 4	8K, 32, 2	16K, 32, 4	16K, 32, 4	8K, 32, 4	8K, 32, 4	8K, 32, 4
	40°C	2K, 16, 4	2K, 32, 4	2K, 32, 4	8K, 32, 2	8K, 32, 4	16K, 32, 2	8K, 32, 4	8K, 32, 4	8K, 32, 4
	60°C	2K, 32, 1	2K, 32, 4	2K, 32, 4	8K, 32, 8	8K, 32, 4	8K, 32, 4	8K, 32, 4	4K, 16, 4	8K, 32, 4
	80°C	2K, 32, 1	2K, 32, 4	2K, 32, 4	8K, 32, 2	8K, 32, 4	8K, 32, 4	8K, 32, 4	4K, 16, 4	8K, 32, 4
	100°C	2K, 32, 1	1K, 8, 2	1K, 32, 4	8K, 32, 2	4K, 32, 4	8K, 32, 4	4K, 32, 2	4K, 16, 4	8K, 32, 4