Hot Peripheral Thermal Management to Mitigate Cache Temperature Variation

Houman Homayoun[1], Mehryar Rahmatian[2], Vasileios Kontorinis[1], Shahin Golshan[2], Dean M. Tullsen[1]
[1]Dept. of Computer Science and Engineering, University of California, San Diego
[2]Dept. of Computer Science, University of California, Irvine
[1]Email: {hhomayou, vkontori, tullsen}@cs.ucsd.edu,
[2]Email:{mrahmati, golshans}@uci.edu

Abstract
Modern microprocessor caches are often regarded as cool chip components that dissipate power uniformly. This research demonstrates that this uniformity is a misconception. Memory cell peripherals dissipate considerably higher power than the actual memory cell and this can result in up to 30°C of temperature difference between the warmest and the coolest part of the cache. To be effective and accurate, cache temperature and power modeling and management must take this effect into account. Further, this paper focuses on the surrounding logic of the memory cell and applies two novel techniques, peripheral bit swapping (PBS) and peripheral monitor and shutdown (PMSD), to reduce the thermal variation as well as reduce the corresponding steady-state temperature and leakage power of the cache. Overall, these techniques decrease temperature by 8°C for the L1 Data Cache and 5°C for the shared L2 cache and reduce their thermal gradient by more than 75%, on average.

Keywords
Peripheral Circuits, Temperature, Cache, Thermal Management, Processor

1. Introduction

Cache memory structures tend to expend less power per total area than other structures in high-performance processor designs at high utilization levels. As a result, it has been suggested in many studies that these structures can be placed in such a way that they act as heat sinks for hotter structures that are nearby [24, 31]. In proposals for 3D stacked architectures, it is often assumed that by placing cache memory layers next to active logic layers [25, 26, 27], many of the thermal challenges of these architectures can be avoided. Most of these proposals, however, assume that power is dissipated evenly over the large cache structure. This research demonstrates that this assumption of uniform thermal variation in cache memories is an invalid assumption, in general. In fact, the temperature can vary as much as 30 degrees across the cache. Such a large thermal variation can significantly reduce the lifetime reliability of the cache [28]. In addition a placement and floor planning methodology that ignores this variation can in fact significantly exacerbate hot spots within the cache by placing them adjacent to other hot regions of the design.

The variation in thermal behavior across the cache comes from four sources: (1) the difference in activity factors for peripheral logic (global address routing drivers, local address routing drivers, global data in/out drivers, row predecoder drivers, and wordline drivers) versus the memory cells, (2) the different transistor types used for peripheral logic versus the memory cells, (3) the uneven distribution of 1's in cache (e.g., most 1's appear in the low bits of 64-bit integers), and (4) the uneven distribution of bit flips in the routing and buffer logic.

The peripheral logic of the cache represents a relatively small area of a large cache, but accounts for a majority of the activity and power, while memory cells dominate the area but are individually accessed rarely and designed to minimize static power dissipation. A program whose primary data type is 64-bit integers (for example) will typically find that the high bits are typically zero and as a result those same bits in the routing and peripheral logic flip much less often, thereby dissipating less dynamic energy than those lines which read and transmit the low bits.

This paper addresses this problem of the high thermal variance of cache designs and makes three major contributions as follows: First, it demonstrates a large thermal variation in cache memories between the peripheral logic and the memory banks, indicating the importance of more accurately modeling the thermal behavior of cache designs. This can be used to enable current modeling tools to better study the impact of block placement on the actual cache hot spots. Second, it presents design and architectural solutions that reduce the thermal variation of caches, enhancing the lifetime reliability of the caches, making them appear closer to what is typically modeled, and enhancing their ability to absorb heat from neighboring structures without adverse effects. Third, it presents architectural solutions to reduce the steady-state temperature of the cache, by reducing the temperature of the hotspot peripherals within the cache — this results in significant reduction of cache leakage power (due to interdependence of leakage and temperature [30]) and reduction of SRAM cache failure probability [29, 33].

We demonstrate two techniques that reduce thermal variation and reduce steady-state temperature in caches: peripheral bit swapping and peripheral monitoring and shutdown. Peripheral bit swapping migrates switching activity from the less significant bits to the more significant bits by periodically multiplexing them. This results in significant temperature reductions. Peripheral monitoring and shutdown uses a zig-zag circuit to disable the periphery when periods of inactivity are detected and thus achieves big leakage power decrease.

This paper is organized as follows. Section 2 provides a brief background on modern cache organization and design. Section 3 quantifies power dissipation and temperature variation on different cache components. In section 4 we present our power and temperature reduction techniques and section 5 gives our results. Finally, section 6 describes related work and section 7 concludes.

2. Background on cache architecture

In this section we provide an overview of the standard cache architecture layout. The cache array is composed of multiple identical banks [9]. Each bank is composed of multiple identical sub banks. Only one sub bank can be active at a time. A sub bank is also composed of multiple identical mats. The mat itself is a memory structure with a memory cell array, a row decoder, a wordline driver, a pre-charge circuit, local input and output drivers, a sense amplifier, and a bitline multiplexer. The area of the cache memory (data array part) is estimated based on the area of a single bank (WBank x HBank) and the area occupied in routing the address and the data to/from the banks [9]. Address and data are typically routed in an H-Tree distribution network to the mats [9]. H-tree distribution networks are used to route address and data and provide uniform access to all the mats in a large memory. To facilitate pipelining multiple accesses to the memory structure, separate request and reply networks are assumed. The request network carries address and data-in from the edge of the array to the mats while the reply network carries data-out from the mats to the edge of the array. Figure 1 shows the layout of the request H-tree network between the array edge and the banks. Address and data-in drivers are routed to each bank on this H-tree network and enter each bank at the middle from one of its sides. The H-tree network from the edge of the bank to the mats is further divided into two 1-dimensional horizontal and vertical H-tree networks. We refer to all these H-tree local/global horizontal/vertical routing networks as peripheral. These includes address H-tree drivers, data in/out H-tree drivers, predecoder drivers and global wordline drivers. Figure 1(b) shows the layout configuration of horizontal routing of the address and the data driver to each memory cell array. The drivers are typically placed at each of the nodes in the H-tree (V and H node in the figure).

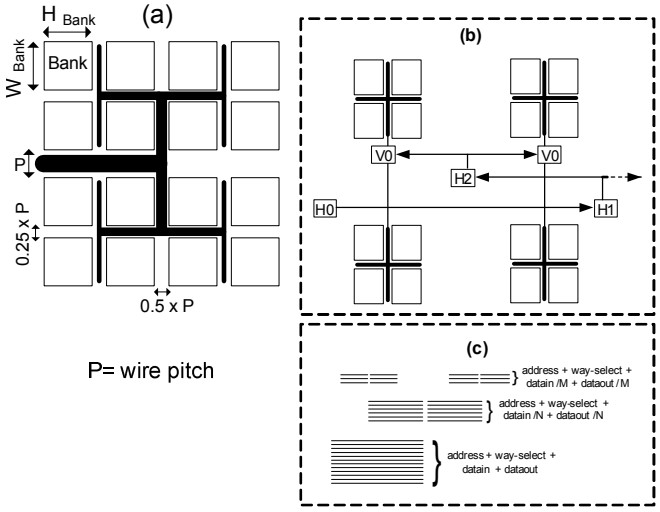


Figure 1. (a) Layout of H-tree network from edge of array to banks (b) Layout of horizontal H-tree address and data routed to the memory cell array, and (c) Layout of wires for horizontal H-tree.

Figure 1(c) shows the layout of wires for the horizontal H-tree assuming that the wires are laid out using a single layer of metal.

3. The need for power and temperature reduction in SRAM peripheral circuits

This section explains the importance of power and temperature management in the peripheral circuits of cache memories.

3.1. Peripheral circuit power dissipation

Recent studies [8, 13, 17, 19] have shown that peripheral circuits are responsible for a large portion of overall memory power dissipation. They dissipate significant dynamic power due to their high activity (every access to any memory bank has to go through peripheral circuits including address, data in, and data out, which make their activity factor large). Also, recent studies demonstrate that the peripheral circuits dissipate a large portion of total cache leakage power [8], [12], [13], [17], [19], [20]. Figure 2 shows the leakage and dynamic power for different components of a modern 128KB L1 Data cache (DL1) and a 4MB L2 cache for 45nm technology, based on CACTI 6.5 [9]. CACTI uses the characteristics of transistors modeled by the ITRS [35]. It includes data for the tree device types that the ITRS defines - High Performance (HP), Low Standby Power (LSTP) and Low Operating Power (LOP). The HP transistors are fast transistors with short gate lengths, thin gate oxides, low Vth, and low VDD. The LSTP transistors, on the other hand, have longer gate lengths, thicker gate oxides, higher Vth, and higher VDD. The LOP transistors use the lowest VDD to control the operating power and have performance between the HP and LSTP transistors. Similar to [8, 9, 19, 20] we assume that L2 cache memory cells use low standby power (LSTP) transistors while L2 peripherals use the high performance (HP) transistors. For DL1, due to timing constraints, the memory cells use faster, higher-leakage transistors [8, 9, 19, 20]. We assume ITRS-HP transistors for the DL1 peripherals and the ITRS-LOP transistors for the DL1 cells. Figure 2 demonstrates that the peripheral circuits – data drivers, address driver, decoder, and wordline drivers – account for over 80% of the overall cache power. In brief, two main reasons explain this difference in leakage:

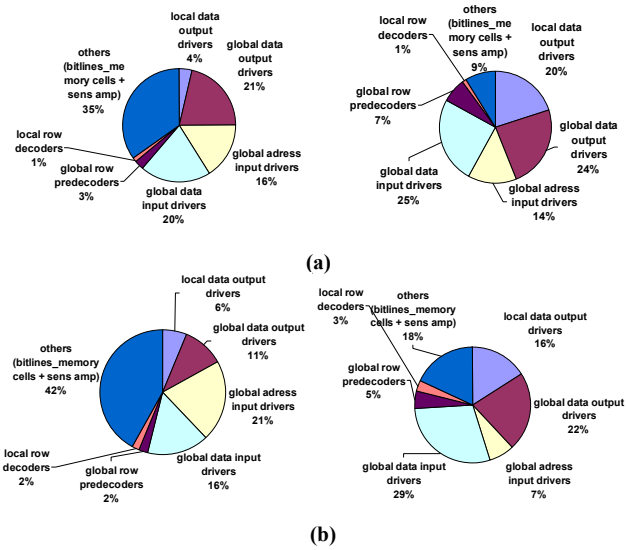


Figure 2. Dynamic power (left) and leakage power (right) of (a) 4MB and (b) 128KB cache.

1. Memory cells are designed with minimum-sized transistors mainly for area considerations. Unlike memory cells, peripheral circuits use larger, faster and accordingly, more leaky transistors so as to satisfy timing requirements [8, 13, 17].

2. Memory cells use high threshold voltage transistors, which have significantly lower leakage compared with typical threshold voltage transistors used in peripheral circuits [8], [13], [17], [19].

In summary, SRAM memory cells are optimized for low power (specifically leakage power) and area without a significant impact on the cell performance [13, 19, 20, 21]. In addition, proposed circuit techniques such as gated-Vdd and drowsy cache further reduce the memory cell leakage and only widen the gap between cell array and peripheral leakage power dissipation.

3.2.Peripheral circuit temperature

In this section we present the results of a detailed thermal modeling and temperature estimation of on-chip cache memories. In figure 3 we show the thermal image of a large 4MB L2 cache and a small 128KB L1 Data cache. We integrated CACTI 6.5 with Hotspot 5.0 to generate this thermal image. The area data for memory block height/width, bank height/width, mat height/width, etc. have been extracted from CACTI and used to generate the detailed floorplan of the SRAM cache. The leakage and dynamic power of the cache components, including the data in/out horizontal and vertical h-tree, address horizontal/vertical h-tree, and predecoder driver were also extracted from CACTI. As reported in figure 3, peripheral circuits occupy a relatively small part of the cache area. Most of the area is occupied by memory banks.

Using Hotspot the steady state temperature of various blocks of the cache is calculated. To acquire the steady state temperature we assume that the accesses are randomly distributed among memory banks. The thermal image shows a large thermal gradient across the entire cache block. The hottest units indicated are the peripheral circuits both in DL1 and L2 cache. To better explain such a large thermal difference across L2, we refer to the power breakdown in figure 2 and the area breakdown of the DL1 and L2 caches in figure 3. In fact, a large thermal difference between peripherals and memory banks is due to their large power density difference. Peripherals have a noticeably higher power density than the memory banks and as a result they have higher temperature. In addition, any access to DL1 and L2 caches has to go through peripherals while accesses to memory cells are distributed across all banks. In other words, the activity factor of peripherals are higher than the activity factor of a single memory bank. The thermal variation as shown in the figure is as large as 12 degrees (comparing P0 temperature vs. B0) for both DL1 and L2 caches.

3.3.Sensitivity analysis: thermal variation w.r.t cache organization

In this section we study thermal variation across different blocks of an DL1 and L2 cache as a function of cache size (number of bytes), I/O width (number of data read out/write bits) and cache associativity.

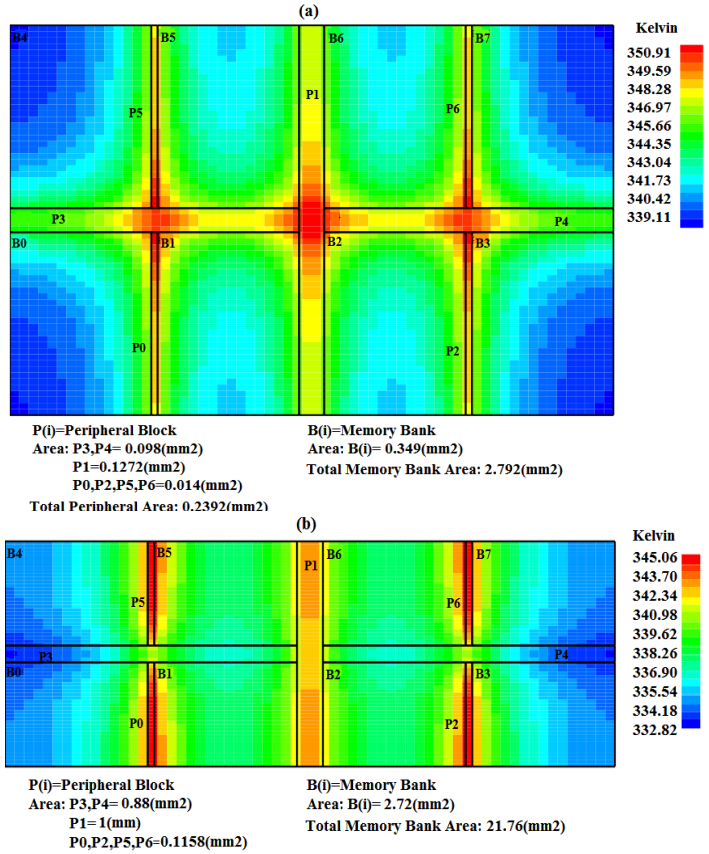


Figure 3. Thermal variation within a (a) 128 KB 4-way set associative, 64 readout bits DL1 cache and (b) 4MB 16-way set associative 256 readout bits L2 cache.

The sensitivity analysis results are reported in figure 4. We report maximum thermal variation between banks and peripherals and for the worst case scenario where the memory (DL1 and L2 caches) is accessed every cycle.

The results reported in figure 4 show that for various memory configurations the thermal variation between bank and peripheral is large. The largest variation observed is close to 30 degrees (Kelvin) for 1 MB L2 cache with associativity of 4. For larger cache size and wider read-out lines, the thermal variation is reduced. In both cases, this spreads the peripheral activity over a larger area. Also, increasing the associativity reduces the thermal variation between banks and peripherals. Increasing the associativity results in increasing leakage power dissipation in the banks which reduces the thermal difference between banks and peripherals.

For DL1 cache a smaller thermal variation is observed for various cache configurations. This is in fact due to the smaller power/density difference between peripherals and banks. Note that, compare to L2, DL1 caches typically use higher performance transistors (with lower threshold voltage) both in their peripherals and memory cells to decrease the latency of the data access.

Also as shown in the figure, increasing the size of the DL1 cache does not noticeably change the thermal variation between bank and peripherals.

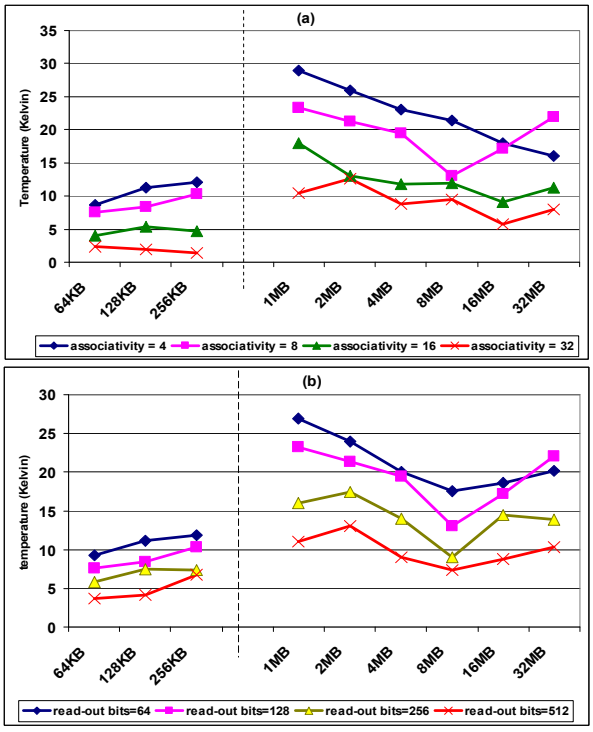


Figure 4. Thermal variation for various DL1 and L2 caches configurations for (a) different associativities (b) different number of output bits.

These results indicate dramatic variation in thermal activity across cache structures. Existing tools for modeling onchip processor thermals typically assume uniform thermal behavior. These results indicate that such assumption can be highly inaccurate and lead to potential design errors, for example when it is assumed that a uniformly cool cache structure can be used as a heat sink for a nearby hot structure.

4. Temperature reduction in peripherals

In this section we present two architectural techniques, peripheral bit swapping (PBS) and peripheral monitoring and shut down (PMSD) to reduce temperature in the peripheral circuits and reduce temperature gradients.

4.1.Peripheral bit swapping (PBS)

A conventional approach to reduce the temperature of a unit is to periodically migrate its activity to a cooler unit. We observe heavy activity in the lower order bits of data I/Os. Note that the majority portion of on-chip memory peripherals is composed of data I/O routing rather than the address routing. For instance for a 2MB L2 cache the size of address I/O is 44 which is noticeably smaller than the 512 data I/O (256 bits data in and 256 bits data out). We present the breakdown of activity factor for individual bits of data read/written from/to DL1 and L2 caches in figure 5 (for 64 bits data width). The reported results are average activity factors for individual bits of data read from DL1 and L2 caches, and across all SPEC2K benchmarks. To find the activity factor we measure the number of bit flip for individual bits when reading data from DL1 and L2 caches. For L2 cache we partition the 256 bits of read data (one line of cache data) into 8 smaller data sets of 64 bits each, for activity factor study. The activity factor for individual bits is

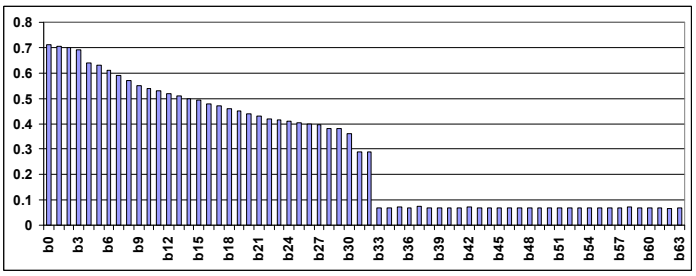


Figure 5. Activity Factor per bit of data read/write from/to Cache. The MSBs beyond 32 has very low activity factor.

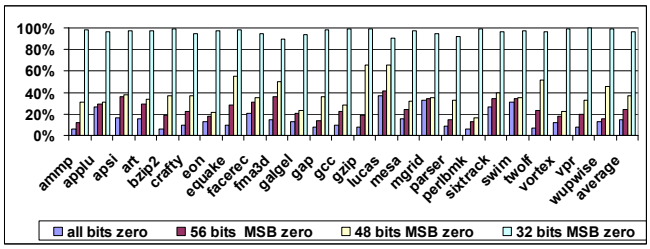


Figure 6. Presence of zero in the data read from DL1 and L2 Cache.

acquired by dividing the total number of bit flips (for individual bits) over the total number of times accessing DL1 and L2 caches. The activity factor is quite large for LSBs and drops gradually for higher order MSBs. The activity factor drops noticeably to below 10% for bits order 32 and above. Thus, not only is there significant imbalance between the peripheral circuitry and the memory cells, but here we see that even the stress on the peripheral circuitry is imbalanced, further exacerbating thermal gradients.

We also report the presence of zeros in the data read/written from/to DL1 and L2 caches (figure 6). Almost 15% of data is zero. 25% of all data has 56 MSBs zero and only has activity in the lower 8 bits. Interestingly more than 95% of accesses to DL1 and L2 caches generate data that has zero in its higher order 32 MSBs. This unbalanced activity (fewer zeroes and more bit flips) results in thermal build up in the LSBs. We propose peripheral bit swapping (PBS) to migrate the activity between high and low bits periodically to balance the activity and reduce temperature. In figure 7 we present the logical implementation of PBS. Multiplexers are inserted after the sense amplifier stage to swap the 32 LSBs with the 32 MSBs (step 1 in the figure).

After the data is routed to the output buffers, the MSBs and LSBs have to be multiplexed again to reorder bits back to their original order (step 2). The major overhead of this swapping scheme is multiplexing power dissipation. While smaller migration period results in higher temperature reduction, it increases the activity of the multiplexer and therefore it increases the power overhead. To reduce the power overhead of multiplexers, migration is done only every 10K cycles (Note that the activity factor of a multiplexer is decided by the switching factor of the inputs and the select signal. By multiplexing every 10K cycles we can reduce the average switching activity of the select signal). Using CACTI power and area model we estimate the multiplexers incur 3% power overhead and less than 2% area overhead for L2 cache. The power and area overhead for DL1 cache is 3% and 3% respectively.

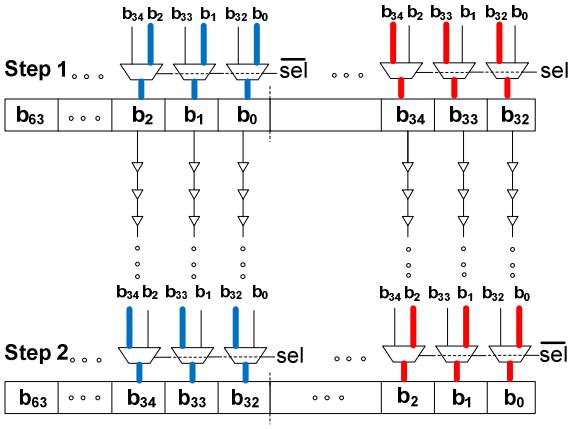


Figure 7. Peripheral Bit Swapping logic implementation.

4.2.Peripheral monitoring and shut down (pmsd)

To control the power of peripheral circuits, we would like to shut them down when they are idle. We propose to deploy MZZ-HVS (multiple sleep mode zigzag sleep transistor insertion technique), a circuit design technique recently proposed to reduce peripheral circuit leakage power in SRAM memories [8, 15]. In this section we first briefly elaborate on the MZZ-HVS technique. We then explain in detail a novel monitoring scheme we propose in this work to control the peripheral circuitry dynamically and shut them down when they are idle.

4.2.1.MZZ-HVS: multi modes zig-zag horizontal and vertical sleep transistor sharing

In MZZ-HVS sleep transistors are inserted in a zig-zag fashion [8] to minimize the impact of sleep transistor on peripheral circuit timing component such as rise time, fall time, and propagation delay. To further minimize the impact on circuit timing and improve both leakage reduction and area-efficiency of the zig-zag scheme, one set of sleep transistors is being shared between multiple stages of peripherals [8]. They further explore the design space of sleep transistor insertion in SRAM peripheral circuitry and show the effect of sleep transistor size and number of horizontal and vertical level sharing on the trade off between the leakage power savings and the impact on instability, area, dynamic power, propagation delay, and rise time and fall time delay increases on the peripheral circuit of SRAM. Results [8, 15] show that using zz-hvs reduces the leakage power significantly, by up to 100X. Such noticeable savings comes at negligible impact on memory access time, dynamic power and area increase. The maximum switching power of the sleep transistors in the zz-hvs scheme is shown to be ~3% of total inverter chain power dissipation. Sharing the sleep transistor across multiple stages of inverter chain, both horizontally and vertically, combined with infrequent switching of sleep transistors makes the additional power dissipation much smaller. Also the memory access delay is shown to be increased by up to 1% in a non-pipelined memory [8]. Pipelined memories such as DL1 and L2 caches can hide this small increase in peripheral circuit latency.

While sleep transistor sharing is effective in reducing leakage, it has a drawback of impacting circuit wakeup latency which occurs when the circuit is transitioning from sleep mode to active mode and requires the voltage of virtual ground to reach ground voltage [15]. To reduce the wakeup delay of the sleep transistor, they increase the bias voltage of the NMOS footer sleep transistor in a zig-zag share circuit (and decrease it for the PMOS header transistor). This gate bias increase reduces the leakage power benefit of the sleep transistor. By controlling the gate voltage of footer and header transistors we can thus define different sleep modes where each mode has a different wakeup delay overhead and a different amount of leakage power reduction. They also propose a stable and robust voltage reference generator to ensure the performance of multiple sleep mode operation over process, voltage, and temperature variations. The bias generator is shown to be fairly small and uses a compact layout with overall area that is not more than 30x30 nm^2.

4.4.2.Peripheral circuits sleep mode control

MZZ-HVS circuit techniques lead to power saving when the underlying circuit is idle. Therefore, it is crucial to identify the idle patterns of peripheral circuitry during the course of program execution. One approach is to monitor individual data/address I/O bits to detect their idle pattern and turn off their associated driver using MZZ-HVS. While per bit control of peripheral circuits could potentially deliver the highest power savings, it is difficult to implement and could significantly impact performance (we will explain this later). In addition, the large power overhead incurred due to frequent peripheral circuit wakeup could potentially surpass the benefit. Therefore, we adapt a coarse-grain monitoring approach; i.e. monitor the entire peripheral logic and shut it down when we predict it to be idle. Such a coarse-grain approach results in less frequent switching of peripheral circuit power mode and reduced wakeup power overhead.

We propose an integrated architectural approach, referred to as coarse-grain peripheral monitoring and shut down (or in brief, PMSD) to control the multiple sleep mode zig-zag share circuits in DL1 and L2 caches. As explained above, there is a latency associated with waking up the SRAM peripheral circuitry. The overall time delay for transition to/from the standby mode, STL, is the sum of the sleep transistors wakeup delay and the propagation delay of the sleep signal. Both of these delays increase as the memory area increases, especially for the latter delay, because the sleep signal needs to be transmitted over a greater distance. Accordingly, depending on the memory size and configuration, there is a different wakeup delay overhead for a specific MZZ-HVS bias voltage. To find the STL delay for an SRAM array, SPICE and CACTI are used to measure the wakeup delay of a sleep transistor and the propagation delay of the sleep signal, respectively. To estimate the propagation delay we assume that the sleep signal has to be transmitted across the SRAM peripherals. Based on these experimental results different sleep modes were defined for the DL1 and L2 caches (see Table 1). The first sleep mode is the basic-lp mode in which the STL delay is dominated by the sleep signal propagation delay. Aggr-mode is the second sleep mode with a larger power reduction compared to the basic-lp but at a higher wakeup delay cost; 2 cycles (1 cycle sleep transistor wakeup + 1 cycle sleep signal propagation delay) for the DL1 cache and 3 cycles (1+2) for the L2 cache. The highest saving power mode is ultra-lp, which shuts down the peripheral circuitry completely through Vss and Vdd bias

selection for NMOS and PMOS sleep transistors, respectively. More precisely, the proposed scheme achieves 79% leakage power reduction relative to peripheral leakage (with no leakage mechanism control) in the DL1 cache and 93% power reduction in the L2 cache (with no leakage mechanism control).

Table 1. DL1 and L2 cache peripherals multiple sleep modes normalized leakage power savings and wakeup delay overhead.

power mode	DL1 Cache			L2 Cache		
	STL delay (cycle)	leakage reduction relative to peripheral leakage (%)	leakage Reduction relative to total cache leakage (%)	STL delay (cycle)	leakage reduction relative to peripheral leakage (%)	leakage reduction relative to total cache leakage (%)
basic-lp	1	42%	38%	2	46%	42%
aggr-lp	2	58%	52%	3	54%	49%
ultra-lp	7	79%	72%	9	93%	84%

Given these wakeup delays, the key issue is how to avoid loss of processor performance while benefiting the most from the largest power saving modes. During periods of frequent access peripheral circuits need to be kept in the basic-lp mode (to limit performance loss) and when their access frequency is low they can be kept in aggr-lp or ultra-lp modes. One period of infrequent access to any of the relevant units is when an L2 cache miss occurs which can result in a processor stall. Thus one approach is to turn off the peripheral circuitry once the processor becomes idle. Such idle periods can occur frequently during program execution. As shown in recent work, when an L2 cache miss occurs the processor executes a number of miss-independent instructions and then stalls [10, 22]. We refer to the interval between an L2 cache miss occurring and the processor stalling as the pre-stall period. The processor stays idle until the L2 cache miss is serviced. This may take hundreds of cycle (300 cycles for our processor model which is similar to the Intel® CoreTM 2 Duo processor). It should be noted that during the pre-stall period the processor still executes some instructions until its resources fill up (such as reorder buffer, instruction queue, and load/store queue). As a result, the processor typically stalls for a large fraction of the L2 cache miss service time. Since processor performance decreases noticeably during the pre-stall period, and the number of accesses to DL1 and L2 reduces significantly [10] we thus propose to put DL1 and L2 cache peripheral into aggr-lp mode during that period. PMSD detects the stall period as follows: The instruction queue and functional units of the processor are monitored after an L2 miss. The sleep signal is asserted to the cache peripheral circuits if the instruction queue has not issued any instructions and functional units have not executed any instructions for K consecutive cycles (K=10). Given a 7 to 9 cycle wakeup latency for the DL1 and L2 cache peripherals, the sleep signal is de-asserted 7 and 9 cycles before the miss service is completed. Note that the L2 cache miss penalty is assumed to be known a-priori. In a non-deterministic memory latency model the cache miss return will trigger the wakeup signal. This would contribute a small delay of 7 to 9 cycles to the overall delay of the L2 cache miss service time. To further improve the leakage reduction, we propose to always put L2 cache peripherals in the basic-lp low power mode. This mode adds 2 cycles to the L2 cache access latency. Considering the delay of accessing L2 is already 20 cycles, the additional 2 cycles have a small impact on performance (see results below). The DL1 cache is accessed more frequently than the L2 and thus its peripherals cannot be kept in a low power mode as this may degrade the performance noticeably. However, the DL1 cache read ports are accessed more frequently than its write ports. Results in Table 2 show that on average a DL1 write port is accessed once every 30 cycles, while a read port is accessed every 8 cycles. Such different access patterns require a different control mechanism for reducing leakage. Our evaluation shows that making a write port one cycle slower (the wakeup delay of basic-lp mode) has a very small impact on performance. But one extra delay cycle on every DL1 read (port) leads to a noticeable performance degradation in some of the benchmarks (e.g. gzip, twolf, and vpr). Therefore, the DL1 write port is always kept in the basic-lp mode and is awakened only when it is accessed.

Table 2. Average time between accesses to DL1 read and write ports (cycles).

	DL1 read	DL1 write		DL1 read	DL1 write
ammp	15.8	73.8	**lucas**	22.7	58.1
applu	10.5	23.6	**mcf**	28.9	203.8
apsi	5.4	11.1	**mesa**	3.7	9.8
art	6.3	29.5	**mgrid**	7.0	52.9
bzip2	4.2	14.3	**parser**	6.4	18.2
crafty	3.2	17.8	**perlbmk**	5.9	11.5
eon	3.9	6.02	**sixtrack**	3.7	10.3
equake	5.7	14.1	**swim**	18.1	44.8
facerec	6.1	10.8	**twolf**	6.6	22.2
galgel	3.5	33.6	**vortex**	4.0	7.6
gap	7.8	15.5	**vpr**	6.6	20.8
gcc	5.8	8.9	**wupwise**	8.9	18.2
gzip	5.3	19.7	**average**	8.2	30.4

During the pre-stall period (after a miss) the DL1 peripherals are put into aggr-lp mode. Both read and write ports are put into the ultra-lp mode once the processor is stalled. Figure 8 shows the general state machine of PMSD to put DL1 and L2 cache peripherals into different low power modes.

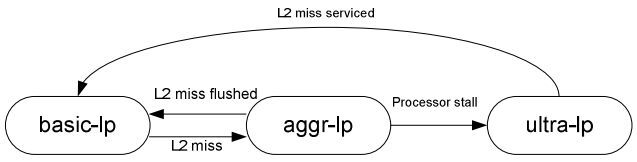


Figure 8. State machine of PMSD.

5. Experimental methodology and results

In this section we present the experimental methodology and results.

5.1. Experimental methodology

We evaluate PBS and PMSD by simulating the processor described in Table 3. The architecture is simulated using an extensively modified version of SimpleScalar 4.0 [11] using the SPEC2K benchmarks. Benchmarks are compiled with the -O4 flag using the Compaq compiler targeting the Alpha 21264 processor. The benchmarks are fast–forwarded for 1 billion instructions, then fully simulated for 5 billion instructions using the reference data sets.

We integrated CACTI 6.5 [9] with Hotspot 5.3 [34] to estimate power, area and temperature of various blocks within the DL1 and L2 caches. The thermal model is described in Table 3. A detailed memory floorplan (shown in figure 3) was created using the area information reported by CACTI. The power trace is obtained every 10K cycles. Once the temperature is calculated it is reported back to the simulator for next-interval leakage power computation. Since the leakage power is a function of temperature, the power simulator includes a lookup table for leakage power dissipation as a function of temperature in the range from 45°C to 120°C in increments of 5°C.

Table 3. Processor organization and thermal model.

Parameter	Value
L1 Instruction	128KB, 2 cycles
L1 Data	128KB, 2 cycles
L2 cache	2MB, 8 way, 20 cycles
Fetch, dispatch	4 wide
Issue	4 way out of order
Memory	300 cycles
Reorder buffer	96 entry
Instruction queue	32 entry
Register file	128 integer and 125 floating point
Load/store queue	32 entry
Branch predictor	64KB entry g-share
Arithmetic unit	4 integer, 4 floating point units
Complex unit	2 INT, 2 FP multiply/divide units
Pipeline	15 cycles
Clock Frequency	3.0 GHz
Die thickness (μm)	150
Ambient temperature	30°C
Convection capacitance	140 J/K
Convection resistance	0.1 K/W
Heat sink side	0.076 m
Heat spreader side	0.035 m

5.2.Results

Figure 9 shows the power reduction for DL1 and L2 caches when applying PMSD. The leakage in DL1 and L2 caches is reduced by up to 72% (applu) and 70% (ammp) respectively. On average, PMSD reduces the leakage power by 40% and 54% for the DL1 and L2 caches, respectively. An interesting observation is that the reduction due to the ultra-lp mode is very significant and is the highest of all low-leakage modes in many cases. This is because DL1 and L2 transition to this mode when the processor stalls. The large number of stalls [10], combined with the large leakage savings associated with the ultra-lp mode make this mode the major source of leakage reduction. Exceptions are crafty, eon, gzip and sixtrack with almost no leakage savings in ultra-lp mode. The reason is that in these benchmark the L2 cache miss rate is almost negligible (see [10]). We also report the PMSD total power reduction in figure 9. On average, PMSD reduces total power by 15% for DL1 cache and 46% for L2 cache. Such a large total power reduction in L2 cache is in fact due to the large contribution of leakage power to total power. Figure 10 shows the PMSD performance degradation in terms of IPC (committed instructions per cycle). The performance reduction is minimal, far below 1%, across most of the benchmarks. The exceptions are art, mcf and swim with 1.6, 0.92 and 0.88% IPC drop, respectively. In fact art, mcf and swim have large DL1 cache miss rates and hence L2 is accessed very frequently (almost one out of every two DL1 accesses is a miss in art). Recall that L2 is always kept in a basic-lp mode, which incurs a 2 cycle wakeup delay before access, explaining the performance degradation in these benchmarks.

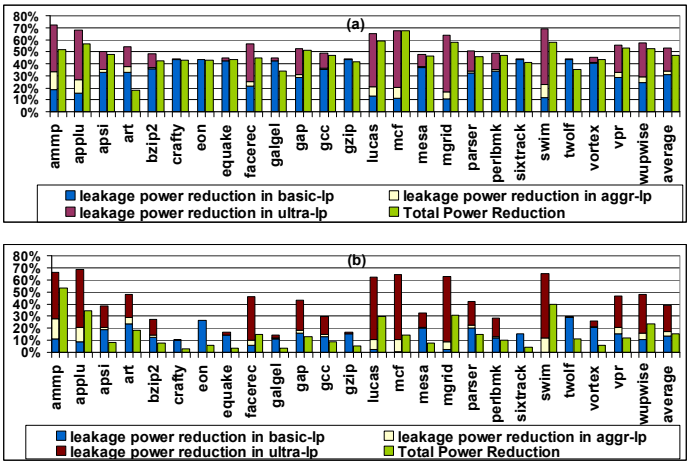


Figure 9. Leakage and total power reduction for (a) L2 and (b) DL1 cache for PMSD.

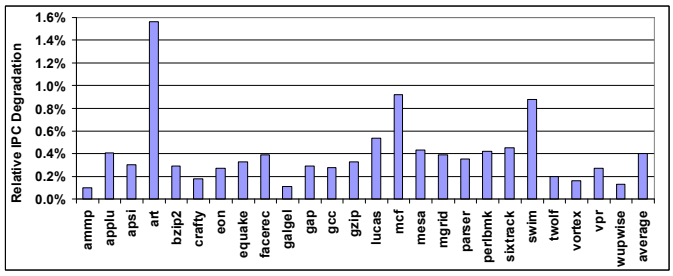


Figure 10. Performance Degradation of the peripheral monitoring and shut down technique.

Additionally, we propose PBS as a technique to reduce thermal imbalance among the memory cells and between bits of the peripheral logic. In figure 11 we present the activity factor per bit for data read from the cache after applying PBS. Comparing figure 5 and figure 11, we see that PBS fairly balanced the activity across the 64 bits of data read from caches. The coefficient of variation (standard deviation to mean) of activity across all bits reduces from 23% to 6% after applying PBS. Having more balanced activity results in a more uniform thermal distribution as well as larger thermal reduction.

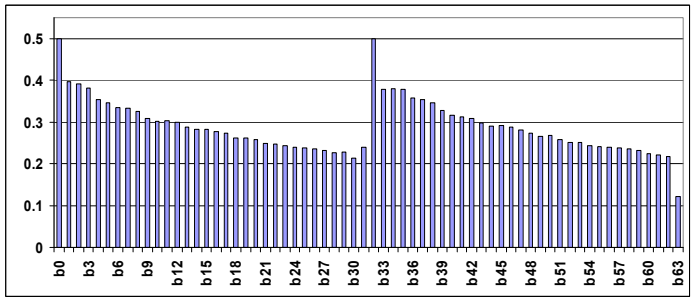


Figure 11. Activity Factor per bit of data read from caches after applying PBS.

In figure 12 we report average temperature reduction when PBS and PMSD are applied together for thermal reduction (we did not present the result for temperature for each technique separately due to space limitation). The absolute temperature for DL1 and L2 cache and for

individual benchmarks is also shown in the figure. A noticeable temperature reduction is achieved for both DL1 and L2 when applying PBS and PMSD. The L2 cache benefits more from the two techniques. This is due to the larger power reduction achieved in L2 when applying the PMSD technique. The largest temperature reduction is 10.5 degrees for mcf and mgrid. In fact these two benchmarks have the highest power reduction (more than 50% power reduction) when applying PMSD. Also, the base temperature of L2 cache is relatively high when running these benchmarks. For DL1 the highest temperature reduction is for ammp. Applying PMSD resulted in a noticeable power reduction for this benchmark (54% as reported in figure 9).

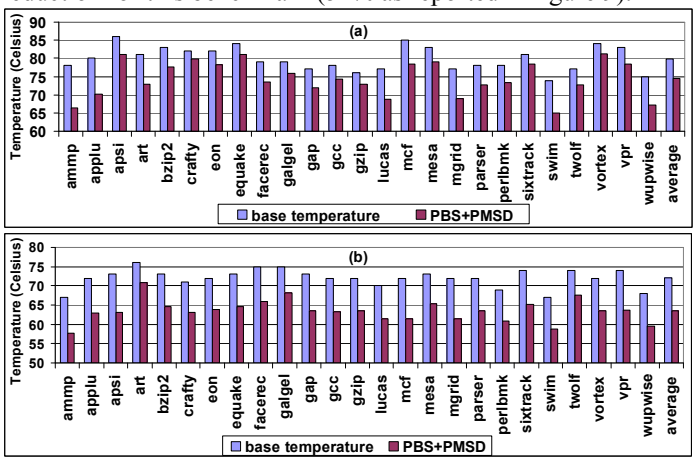
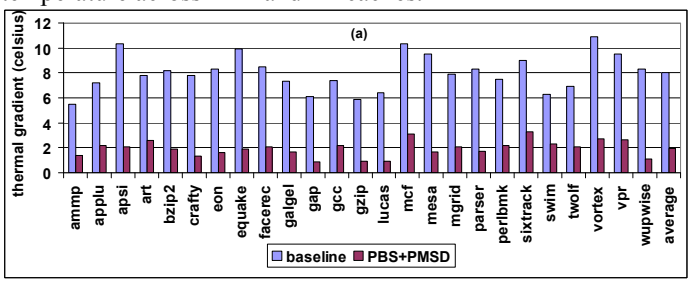


Figure 12. Temperature reduction when applying PBS and PMSD for (a) DL1 and (b) L2 caches.

In figure 13 we present the maximum thermal gradient in DL1 and L2 caches for different benchmarks before (baseline) and after applying the PBS+PMSD techniques. On average there is an 8°C temperature difference between the hottest and the coldest parts of the DL1 cache before applying PBS+PMSD. For L2 this is higher and on average the maximum thermal difference is 11.2°C. These results are consistent with the worst case scenario results presented in figure 4. After applying PBS+PMSD techniques the thermal gradient reduces on average to 1.9°C (76% reduction) for DL1 and 2.3°C (79%) for L2 cache. The largest thermal gradient reduction occurs in wupwise for DL1 cache (85% reduction) and in perlbmk for L2 cache (87% reduction). The results indicate that applying PBS+PMSD leads to significant reduction in thermal gradient and more uniformly distributing temperature across DL1 and L2 caches.



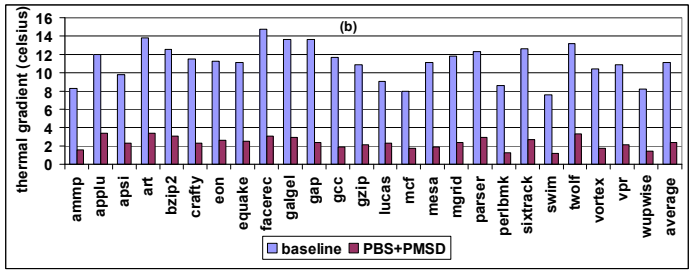


Figure 13. Thermal gradient for (a)basline and (b) after applying PBS+PMSD

6. Related work

Significant prior research has targeted on-chip cache power and thermal characteristics [1, 2, 3, 4, 5, 16, 23, 32]. All these techniques treat caches uniformly and focus on reducing power and temperature in the memory cell (bank), ignoring the surrounding logic. Specifically, way-based techniques will clock-gate or power-gate whole ways, while drowsy-caches and cache decay disable or set in low-power mode only the memory cell. This paper, as discussed in section 1.1, emphasizes for the first time the peripheral of the cache, which has become increasingly hotter for modern cache organizations. Heo, et al. [5] introduce activity migration as an effective solution for hot spot mitigation. They study activity migration across several microprocessor structures at much coarser granularity and add additional resources to generate explicit redundancy. Our PBS technique also employs activity migration; however it does not introduce high area overheads and works at finer time granularity. Ku et. al. [4] propose two thermal-aware cache architectures, namely the power density minimized architecture (PMA) and block permutation schemes (BPS), for cache line level thermal management in set-associative caches. PMA spatially spreads activity by multiplexing accesses to ways within sets, while BPS multiplexes blocks within sets. According to our analysis in section III these techniques will affect temperature and leakage in the cooler portions of the cache, the lower levels of cache topology, without addressing the higher levels of data input and output networks which are the main thermal contributors. Our PBS technique was partially inspired by [7] and their narrow operand technique. They explicitly detect narrow operands and clock-gate unused portions of the ALU, or pack more data through the same bitlines. We simply stress a similar phenomenon for the data part of on-chip caches and leverage it with datapath activity migration.

7. Conclusion

This research focuses on the peripheral circuitry of SRAM memory, which is shown to be a major contributor of power dissipation for on-chip caches. In this work we highlight the large thermal variation in on-chip cache and indicate the importance of more accurately modeling the thermal behavior of cache designs. Targeting peripheral circuitry, we propose two novel architectural techniques – peripheral bit swapping and peripheral monitor and shutdown – that successfully reduce power and average temperature in Data L1 and shared L2 caches.

Our first technique, peripheral bit swapping, migrates switching activity from the less significant bits to the more significant bits of data read from cache by periodically multiplexing them. Our second technique, peripheral monitor and shutdown, detects periods of inactivity and uses a zig-zag circuit to disable cache periphery. The two techniques together reduce leakage power by 40% and 54% on average for the L1 Data and the L2 cache respectively. They result in a temperature reduction of up to 13°C for some benchmarks in L1 Data cache and up to 10°C in the shared L2. 10°C temperature reduction in large caches translates to an additional 7% leakage reduction and 5-10x reduction in probability of failure [18]. Applying the two techniques also results in a large reduction in maximum temperature difference in L1 and L2 caches. All these come with a performance reduction that is negligible (less than 0.5%).

8. Acknowledgments

This research was supported in part by the National Science Foundation Computing Innovation Fellow Program and under the grant NSF 1019343/CRA, Sub Award CIF-B-68, and by grant 2086.001 from the Semiconductor Research Corporation.

9. References

[1] D. H. Albonesi. "Selective cache-ways: On demand cache resource allocation." In Proc. of MICRO, 1999.

[2] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge. "Drowsy caches: simple techniques for reducing leakage power". In Proc. of ISCA, 2002.

[3] S. Kaxiras, Z. Hu, and M. Martonosi. Cache decay:exploiting generational behavior to reduce cache leakage power. In Proc. of ISCA, 2001.

[4] J. C. Ku, S. Ozdemir, G. Memik, Y. Ismail, "Thermal Management of On-Chip Caches Through Power Density Minimization", In Proc. of MICRO, 2005.

[5] S. Heo, K. Barr, K. Asanovi´c, "Reducing Power Density through Activity Migration", In Proc. of ISLPED, 2003.

[6] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, D. Tarjan, "Temperature-aware microarchitecture," In Proc. of ISCA, 2003.

[7] D. Brooks, M. Martonosi. "Dynamically Exploiting Narrow Width Operands to Improve Processor Power and Performance". In Proc. of HPCA, 1999.

[8] H. Homayoun, M. Makhzan, A. Veidenbaum, "ZZ-HVS: Zig-Zag Horizontal and Vertical Sleep Transistor Sharing to Reduce Leakage Power in On Chip SRAM Peripheral Circuits", In Proc. of ICCD, 2008.

[9] Cacti 6.5, http://quid.hpl.hp.com:9082/cacti/.

[10] Arun Kejariwal et al., "Comparative Architectural Characterization of SPEC CPU2000 and CPU2006 Benchmarks on the Intel Core 2 Duo Processor," In Proc. of SAMOS VIII , 2008.

[11] SimpleScalar4 tutorial, http://www.simplescalar.com/tutorial.html.

[12] G. Gerosa et al., "A Sub-lW to 2W Low-Power IA Processor for Mobile Internet Devices and Ultra-Mobile PCs in 45nm Hi-K Metal Gate CMOS," in ISSCC-2008.

[13] Y. Nakagome et al., "Review and future prospects of low-voltage RAM circuits," IBM Journal of R&D, Sept, 2003.

[14] H. Li, C.-Y. Cher, T. Vijaykumar, and K. Roy, "VSV: L2-miss-driven variable supply-voltage scaling for low power," Proc. of MICRO, December, 2003.

[15] K. Agarwal, H. Deogun, D. Sylvester, K. Nowka, "Power gating with multiple sleep modes," In ISQED 2006.

[16] H. Homayoun, S. Pasricha, M.A. Makhzan, A. Veidenbaum: Dynamic register file resizing and frequency scaling to improve embedded processor performance and energy-delay efficiency, Proc. 45th ACM/IEEE Design Automation Conference DAC 2008, 2008.

[17] M. Mamidipaka, K. S. Khouri, N. Dutt and M. S. Abadir, "Analytical models for leakage power estimation of memory array structures," in CODES+ISSS 2004.

[18] Amin Khajeh, Aseem Gupta, Ahmed Eltawil, Fadi Kurdahi, Nikil Dutt, "TRAM: A Tool for Temperature and Reliability Aware Memory Design", Proc. of DATE, 2009.

[19] Y. Takeyama et al., "A Low Leakage SRAM Macro with Replica Cell Biasing Scheme," IEEE Journal Of Solid- State Circuits, April, 2006.

[20] K. Nii et al., "A 90-nm low-power 32 KByte embedded SRAM with gate leakage suppression circuit for mobile applications," IEEE J. Solid-State Circuits, vol. 39, April, 2004.

[21] E. Grossara et al., "Statistically Aware SRAM Memory Array Design," in International Symposium on Quality Electronic Design, In Proc. of ISQED, 2006.

[22] D. Marculescu, "On the use of microarchitecture-driven dynamic voltage scaling," in Workshop on Complexity-Effective Design, 2000.

[23] H. Homayoun and A. Baniasadi, "Reducing execution unit leakage power in embedded processors," in Proceedings of SAMOS 2006, pp. 299-308, 2006.

[24] Kevin Skadron et al. "Temperature-aware microarchitecture." In Proc. of ISCA, 2003

[25] Bryan Black, et al. "Die Stacking (3D) Microarchitecture". In Proc. of MICRO, 2006.

[26] Gabriel H. Loh."3D-Stacked Memory Architectures for Multi-core Processors" In Proc. of ISCA, 2008.

[27] Ayse K. Coskun, et al. "Dynamic thermal management in 3D multicore architectures" In Proc. of DATE, 2009.

[28] W. Wu, J. Yang, S. X.-D. Tan, and S.-L. Lu. Improving the reliability of on-chip data caches under process variations. In Proc. of ICCD, 2007.

[29] K. Banerjee et al., "Analysis of Non-Uniform Temperature-Dependent Interconnect Performance in High Performance ICs," Proc. of DAC, 2001.

[30] A. Gupta et al., "A System Level Leakage-Aware Floorplanner for SoCs," Proc. of ASP-DAC, 2007.

[31] W-L. Hung, et al. "Thermal-Aware Floorplanning Using Genetic Algorithms." In Proc. of ISQED, 2005.

[32] H. Homayoun, Mohammad Makhzan, Alex Veidenbaum, "Multiple sleep mode leakage control for cache peripheral circuits in embedded processors", in Proc. CASES 2008

[33] A. Sasan, H. Homayoun, et al., "A fault tolerant cache architecture for sub 500mV operation: resizable data composer cache (RDC-cache)," in Proc. CASES 2009

[34] K. Skadron et al., "Temperature-aware microarchitecture," in Proc. of ISCA, pp. 2–13, Jun. 2003.

[35] Semiconductor Industries Association, "International Technology Roadmap for Semiconductors," 2005, http://www.itrs.net/