To appear in the **Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)**, 2000.

Gated-V_{dd}: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories

Michael Powell, Se-Hyun Yang, Babak Falsafi, Kaushik Roy, and T. N. Vijaykumar

School of Electrical and Computer Engineering
Purdue University
1285 EE Building
West Lafayette, IN 47907

icalp@ecn.purdue.edu, http://www.ece.purdue.edu/~icalp

Abstract

Deep-submicron CMOS designs have resulted in large leakage energy dissipation in microprocessors. While SRAM cells in on-chip cache memories always contribute to this leakage, there is a large variability in active cell usage both *within* and *across* applications. This paper explores an integrated architectural and circuit-level approach to reducing leakage energy dissipation in instruction caches. We propose, *gated-V_{dd}*, a circuit-level technique to gate the supply voltage and reduce leakage in unused SRAM cells. Our results indicate that gated-V_{dd} together with a novel resizable cache architecture reduces energy-delay by 62% with minimal impact on performance.

1 INTRODUCTION

The ever-increasing levels of on-chip integration in the recent decade have enabled phenomenal increases in computer system performance. Unfortunately, the performance improvement has been also accompanied by an increase in a chip's power and energy dissipation. Higher power and energy dissipation require more expensive packaging and cooling technology, increase cost, decrease product reliability in all segments of computing market, and significantly reduce battery life in portable systems.

Historically, chip designers have relied on scaling down the transistor supply voltage in subsequent generations to reduce the *dynamic energy* dissipation due to a much larger number of on-chip transistors. Maintaining high transistor switching speeds, however, requires a commensurate down-scaling of the transistor threshold voltage giving rise to a significant amount of *leakage energy* dissipation even when the transistor is not switching. Borkar [3] estimates a factor of 7.5 increase in leakage current and a five-fold increase in total leakage energy dissipation in every chip generation.

State-of-the-art microprocessor designs devote a large fraction of the chip area to memory structures — e.g., multiple levels of instruction (i-cache) caches and data (d-cache) caches, TLBs, and prediction tables. For instance, 30% of Alpha 21264 and 60% of StrongARM are devoted to cache and memory structures [8]. Unlike dynamic energy which depends on the number of actively switching transistors, leakage energy is a function of the number of on-chip transistors, independent of their switching activity. As such, caches account for a large (if not dominant) component of leakage energy dissipation in recent designs, and will continue to do so in the future. Unfortunately, current proposals for energy-efficient cache architectures [7,2,5,1] only target reducing dynamic energy and do not impact leakage energy.

There are a myriad of circuit techniques to reduce leakage energy dissipation in transistors/circuits (e.g., multi-threshold or multi-supply voltage design, dynamic threshold or dynamic supply voltage design, transistor stacking, and cooling). These techniques, however, either impact circuit performance and are only applicable to circuit sections that are not performance-critical, or may require sophisticated fabrication process and increase cost.

Modern cache hierarchies are designed to satisfy the demands of the most memory-intensive application phases. The actual cache utilization, however, varies widely both *within* and *across* applications. We have recently proposed the Dynamically ResIzable instruction-cache (DRI i-cache) [11], a novel cache architecture that exploits this variability in utilization.

Our cache design presents the first fully-integrated architectural and circuit-level approach to reducing energy dissipation in deep-submicron cache memories. A DRI i-cache identifies an application's i-cache requirements dynamically, and uses a circuit-level mechanism, *gated-V_{dd}*, to gate the supply voltage to the SRAM cells of the cache's unused sections and reduce leakage.

While voltage gating effectively eliminates the leakage in SRAM cells, it may adversely impact cell performance and prohibitively increase cell area. This paper evaluates in detail the design space for gated-V_{dd} with respect to performance, energy, and area trade-offs. Our results indicate that: (i) a PMOS gated-V_{dd} transistor incurs negligible impact on cell performance and area but only reduces leakage by an order of magnitude, (ii) an NMOS dual-V_t gated-V_{dd} transistor virtually eliminates leakage with minimal impact on the cell area but increases cell read time by 35%, (iii) a wide NMOS dual-V_t gated-V_{dd} transistor with a charge pump offers the best configuration and virtually eliminates leakage with minimal impact on cell speed and area, and (iv) using gated-V_{dd} a DRI i-cache reduces the overall energy-delay in applications by 62%.

ISLPED'00, Rapallo, Italy.

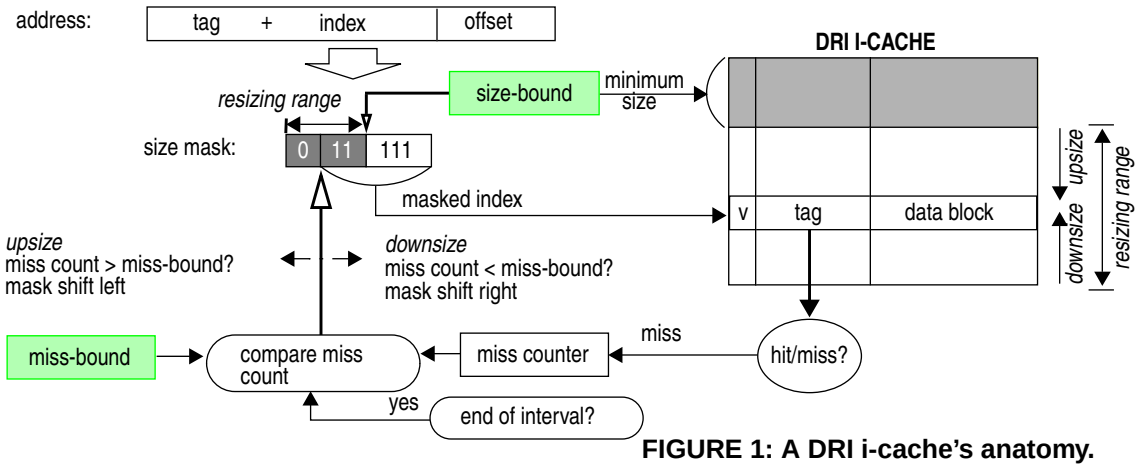

FIGURE 1: A DRI i-cache's anatomy.

The rest of the paper is organized as follows. In Section 2, we present an overview of a DRI i-cache. In Section 3, we describe the circuit-level gated-V_{dd} mechanism to reduce leakage in SRAM cells. In Section 4, we present experimental results. Finally, we conclude the paper in Section 5.

2 DRI I-CACHE OVERVIEW

The key observation behind a DRI i-cache design is that there is a large variability in i-cache utilization both *within* and *across* programs leading to large energy inefficiency in conventional caches; while the memory cells in the cache's unused sections are not actively referenced, they leak current and dissipate energy. Our approach to resizing the cache increases or decreases the number of sets used in the cache. In this section, we present an overview of a DRI i-cache's anatomy. For a more detailed description of a DRI i-cache, please refer to [11].

2.1 DRI i-cache design

Much like conventional adaptive computing frameworks, our cache uses a set of parameters to monitor, react, and adapt to changes in application behavior and system requirements dynamically. Figure 1 depicts the anatomy of a direct-mapped DRI i-cache (the same design applies to set-associative caches). The cache monitors itself in fixed-length *sense intervals*, measured in number of dynamic instructions (e.g., one million instructions). A miss counter counts the number of cache misses in each sense interval. At the end of each sense interval, the cache upsizes/downsizes, depending on whether the miss counter is lower/higher than a preset *miss-bound* value. The factor by which the cache resizes (up or down) is called the *divisibility*. To avoid thrashing, a DRI i-cache never downsizes beyond a preset *size-bound* value. The cache's adaptive parameters are all set at the start of execution.

Among these parameters, the key parameters that control the i-cache's size and performance are the miss-bound and size-bound. The combination of these two key parameters provides accurate and tight control over the cache's performance. Miss-bound allows the cache to react and adapt to an application's instruction working set by "bounding" the cache's miss rate in each monitoring interval. Thus, the miss-bound provides a "fine-grain" resizing control between any two intervals independent of the cache size. Applications typically require a specific minimum cache capacity beyond which they incur a large number of capacity misses and thrash. Size-bound provides a "coarse-grain" resizing control by preventing the cache from thrashing due to a small size.

The other two parameters, the sense interval length and divisibility, are less critical to a DRI i-cache's performance. Intuitively, the sense interval allows selecting an interval length that best matches an application's phase transition times, and the divisibility determines the amount by which the working set size changes.

Resizing the cache requires that we dynamically change the cache block lookup and placement function. Conventional (direct-mapped or set-associative) i-caches use a fixed set of index bits from a memory reference to locate the set to which a block maps. Resizing the cache either reduces or increases the total number of cache sets thereby requiring a larger or smaller number of index bits to look up a set. Our design uses a mask to find the right number of index bits used for a given cache size (Figure 1). Every time the cache downsizes, the mask shifts to the right to use a smaller number of index bits and vice versa. Therefore, downsizing removes the highest-numbered sets in the cache in groups of powers of two. The mask can be folded into the address decoder trees of the data and tag arrays, so as to minimize the impact on the lookup time.

Because smaller caches use a small number of index bits, they require a larger number of tag bits to distinguish data in block frames. Because a DRI i-cache dynamically changes its size, it requires a different number of tag bits for each of the different sizes. To satisfy this requirement, our design maintains as many tag bits as required by the smallest size to which the cache may downsize itself. Thus, we maintain more tag bits than conventional caches of equal size. We define the extra tag bits to be the *resizing tag bits*. The size-bound dictates the smallest allowed size and, hence, the corresponding number of resizing bits. For instance, for a 64K DRI i-cache with a size-bound of 1K, the tag array uses 16 (regular) tag bits and 6 resizing tag bits for a total of 22 tag bits to support downsizing to 1K.

2.2 Impact on Energy and Performance

Cache resizing helps reduce leakage energy by allowing a DRI i-cache to turn off the cache's unused sections. Resizing, however, may adversely impact the miss rate (as compared to a conventional i-cache) and the access frequency to the lower-level (L2) cache. The increase in L2 accesses may impact both execution time and the dynamic energy dissipated in L2. While the impact on execution time depends on an application's sensitivity to i-cache performance, the higher miss rate may significantly impact the dynamic energy dissipated due to the growing size of on-chip L2 caches [1]. A DRI i-cache may also increase the dynamic energy dissipated as compared to a conventional cache due to the extra resizing tag bits in the tag RAM. The combined effect of the above may offset the gains in leakage energy. In Section 4.2, we will present results that indicate that the leakage reduction in a DRI i-cache significantly offsets the increase in the dynamic energy dissipated.

3 GATED-V_{DD}: GATING THE SUPPLY VOLTAGE

Subthreshold leakage current and leakage energy dissipation increase exponentially with decreasing threshold voltage. To prevent leakage energy dissipation in a DRI i-cache from limiting aggressive threshold-voltage scaling, we propose a novel circuit-level mechanism, called *gated-V_{dd}*. Gated-V_{dd} enables a DRI i-cache to "turn off" the supply voltage and eliminate virtually all the leakage energy dissipation in the cache's unused sections. The key idea is to introduce an extra transistor in the supply voltage (V_{dd}) or the ground path (Gnd) of the cache's SRAM cells; the extra transistor is turned on in the used sections and turned off in the unused sections. Thus, the cell's supply voltage is "gated."

Gated-V_{dd} maintains the performance advantages of lower supply and threshold voltages while reducing leakage and leakage energy dissipation. The fundamental reason for the reduction in leakage is the stacking effect of self reverse-biasing series-connected transistors [12]. Gated-V_{dd}'s extra transistor produces the stacking effect in conjunction with the SRAM cell transistors when the gated-V_{dd} transistor is turned off.

3.1 SRAM cell with gated-V_{dd}

Cache data arrays are usually organized in banks; each bank contains SRAM cell rows, with each row containing one or more cache blocks. In this paper, we assume conventional 6-T SRAM cells with dual-bitline architecture. Figure 2 shows a DRI i-cache SRAM cell using an NMOS gated-V_{dd} transistor; PMOS gated-V_{dd} is achieved by connecting the gated-V_{dd} transistor between V_{dd} and the SRAM PMOS transistors. The gated-V_{dd} transistor is turned on for the cell to be in "active" mode and turned off for the cell to be in "standby" mode.

Much as conventional gating techniques, the gated-V_{dd} transistor can be shared among multiple circuit blocks to amortize the overhead. To reduce the impact on SRAM cell speed and to ensure stability of the SRAM, the gated-V_{dd} transistor must be carefully sized with respect to the SRAM cell transistors it is gating. While a gated-V_{dd} transistor must be made large enough to sink the current flowing through the SRAM cells during a read/write operation in the active mode, too large a gated-V_{dd} transistor may reduce the stacking effect, thereby diminishing the energy savings. Moreover, large transistors also increase the area overhead.

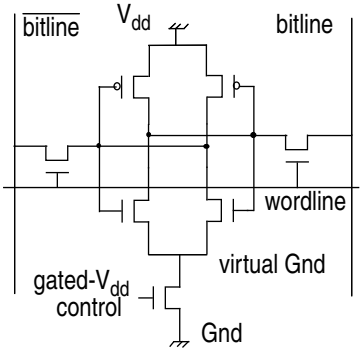

FIGURE 2: SRAM with an NMOS gated-V_{dd}.

3.2 Gated-V_{dd} with NMOS or PMOS transistors

Using a PMOS or an NMOS gated-V_{dd} transistor presents a trade-off between area overhead, leakage reduction, and impact on performance.

To maintain stability and high SRAM cell speed, an NMOS gated-V_{dd} transistor needs to be sufficiently wide. One estimate is to use the sum of the widths of all the transistors that could simultaneously switch in the SRAM cells. If an entire cache block is connected to a single NMOS gated-V_{dd} transistor, the desired width of the transistor may be determined as the product of the width of one of the SRAM cell's NMOS transistors (because only one of the two is "on" during a read) and the number of cells in the cache block. Such a wide NMOS gated-V_{dd} transistor may incur a high area overhead. Using NMOS gated-V_{dd} transistors, however, substantially reduces standby energy dissipation through the stacking effect of three NMOS transistors between the bitlines and ground.

Alternatively, using a PMOS gated-V_{dd} transistor significantly reduces the required transistor width. Dual-bitline architectures typically precharge the bitlines before read operations, so the PMOS transistors simply help in holding the cell value intact and do not contribute to read operations. It reduces the required gated-V_{dd} transistor width, resulting in a negligible area overhead. A PMOS gated-V_{dd} transistor, however, does not create the isolation between the bitlines and the ground as does an NMOS transistor, reducing the amount of energy saving.

The switching speed of a gated-V_{dd} transistor does not impact the SRAM cell speed because it switches only when the DRI i-cache resizes (which is at most every hundreds of thousands of processor cycles). A gated-V_{dd} transistor, however, impacts the switching speed of the cell in the active mode. This impact is mainly due to a non-zero voltage drop across the gated-V_{dd} transistor between the supply rails and the "virtual Gnd" for NMOS gated-V_{dd} (Figure 2) or the "virtual V_{dd}" for PMOS gated-V_{dd}.

When an SRAM cell with an NMOS gated-V_{dd} is read, the discharging of the precharged bitlines takes longer due to the non-Gnd voltage at the virtual Gnd. In contrast, because the PMOS cell transistors do not contribute to read operations, a PMOS gated-V_{dd} transistor does not significantly impact the cell performance. Small

degradation in the cell performance is acceptable because reading a value from the SRAM cell to the bitlines constitutes only a small portion of the total DRI i-cache access time. We use CACTI [10] to model the access time for a 64K DRI i-cache using a 0.18µ process. This model indicates that reading data onto the bitlines is only 6% of the total data access time. The majority of the access time is in decoding the address (40%) and activating the wordline (30%) which are not affected in a DRI i-cache. Because reading data onto the bitlines is such a small portion of the total access time, small changes in SRAM cell performance will not significantly affect overall cache access time.

3.3 Gated-V_{dd} circuit techniques

There is a design spectrum of gated-V_{dd} techniques with various area, energy, and speed trade-offs. The gated-V_{dd} transistor can be made wider to lower the virtual Gnd, allowing more discharging current to flow through the gated-V_{dd} transistor during a cell read. Moreover, forward-biasing the gated-V_{dd} transistor in the active mode increases the current flow. Alternatively, using a charge pump to raise the gate voltage of the gated-V_{dd} transistor would increase the current flow in the active mode.

Gated-V_{dd} can be coupled with a dual-threshold voltage (dual-V_t) process technology to achieve even larger reductions in leakage [9]. SRAM cells use low-V_t transistors to maintain high speed and the gated-V_{dd} transistors use high-V_t to achieve additional leakage reduction. There is an energy-performance trade-off between high- and low-V_t gated-V_{dd} transistors. Raising the threshold voltage for the gated-V_{dd} transistor increases the stacking effect and further reduces leakage current. However, this will impact the read time of the SRAM cells and may have to be offset with other techniques.

4 Results

The reduction in leakage, overall energy savings, and SRAM cell performance depend on the circuit technology used for the gated-V_{dd} transistor (Section 3). Moreover, depending on the circuit technology used for gated-V_{dd}, there is a fundamental trade-off between reduction in leakage, transistor switching speeds, and area overhead of a gated-V_{dd} transistor.

In this section, we first present the methodology used in our circuit evaluation. Then we present empirical results on the performance, energy, and area trade-offs of gated-V_{dd}. Finally, we present results on reducing energy-delay in applications using a DRI i-cache.

4.1 Circuit evaluation

To perform circuit simulations on a DRI i-cache, we determine the cache geometry, use that geometry to lay out a portion of the cache, and extract cell parameters from the layout to estimate energy dissipation and access time. We use CACTI [10] to determine the SRAM layout and geometry of a 64K direct-mapped cache. CACTI estimates the cache's optimal geometry and area utilization. With 32-byte blocks, the cache's data array is divided into 256 by 256 bit banks. All our circuit and layout measurements work with a single cache block of 256 bits and a single cell. Using Mentor Graphics IC-Station, we lay out the SRAM cells of the 256-bit cache block and the gated-V_{dd} transistor and extract netlists and area estimates. We modify the netlist to include our simulation parameters. All simulations use a 0.18µ process and supply voltage of 1.0V. To simulate read time accurately, we model the capacitance of a full bitline. To estimate the SRAM speed and energy dissipation, we vary the spice model's threshold voltage of the SRAM and gated-V_{dd} transistors.

SRAM Cell V_t (V)	Gated-V_{dd} V_t (V)	Active Leakage Energy (nJ)	Standby Leakage Energy (nJ)
0.40	0.40	12	10
0.30	0.40	143	49
0.20	0.40	1700	50
0.40	0.20	12	11
0.30	0.20	143	76
0.20	0.20	1700	165

Table 1: Impact of changing SRAM and gated-V_{dd} threshold voltages.

We estimate cell access time and energy dissipation using Hspice for transient analog analysis. We compute standby and active mode energy dissipation by measuring average energy dissipated by a steady state cache block with the gated-V_{dd} transistor.

4.1.1 SRAM Cell Area

Figure 3 shows a layout from Mentor Graphics ICstation of 64 SRAM cells on the left and an adjoining NMOS gated-V_{dd} transistor connected to them. In the layout, the gated-V_{dd} transistor is actually made up of eight parallel transistors that are each one-eighth of the total desired width. The total increase in data array area due to the addition of the NMOS gated-V_{dd} transistor is about 3% for the layout in the figure. The total width of the gated-V_{dd} control lines is close to that of a single SRAM cell and is negligible. Area increase is negligible for PMOS gated-V_{dd} because the transistor is the size of one of the block's 512 PMOS transistors.

4.1.2 Impact of Lowering Threshold Voltage

Table 1 shows leakage energy with varying SRAM threshold voltages using two NMOS gated-V_{dd} threshold voltages. From the first three rows, decreasing the SRAM cell threshold voltages increases active leakage energy by several orders of magnitude. The standby column shows the standby mode leakage energy using gated-V_{dd}, which is orders of magnitude smaller than active energy. Comparing the first three rows with the last three indicates that decreasing

FIGURE 3: Layout of 64 SRAM cells connected to a single gated-V_{dd} NMOS transistor.

Area Increase (%) of NMOS Gated-V_{dd}	Relative Read Time	Active Leakage Energy (nJ)	Standby Leakage Energy (nJ)
2	1.00	1700	166
4	0.90	1710	245
8	0.85	1720	371

Table 2: Widening the gated-V_{dd} transistor.

the threshold voltage of the gated-V_{dd} transistors significantly increases standby leakage energy dissipation.

4.1.3 Impact of Widening Gated-V_{dd} Transistor

Increasing the width of the gated-V_{dd} transistor improves SRAM cell read times but decreases energy savings and worsens the impact of gated-V_{dd} on SRAM area. Table 2 shows energy, area, and relative speed as the width of the gated-V_{dd} transistor is increased. In the first row, the gated-V_{dd} transistor is sized as described in Section 3.2 and increased in the second and third rows. The cell and the gated-V_{dd} transistors threshold voltage is 0.20V for these simulations. There is a clear trade-off in cell read time against area and standby energy, though the standby energy is low in all cases.

4.1.4 Gated-V_{dd} Techniques Combined

Table 3 depicts four circuit-level gated-V_{dd} techniques we evaluate. The table depicts the percentage of leakage energy saved in the standby mode, the cell read times, and the area overhead of each technique relative to a standard low-V_t SRAM cell with no gated-V_{dd}. The techniques can be grouped into two categories as indicated. The first category (the first three rows) has lower performance and higher energy savings. In contrast, the second has higher performance but potentially lower energy savings.

From the first two rows, we see that in spite of decreasing the cell threshold voltage from 0.40V to 0.20V, gated-V_{dd} manages to reduce the standby mode energy. The second and third rows illustrate the trade-off between energy and speed depending on the threshold voltage of the gated-V_{dd} transistor. If we are willing to sacrifice energy savings for better performance, we may use PMOS gated-V_{dd} transistors. The fifth row indicates a slightly faster read time for gated-V_{dd} because the PMOS gated-V_{dd} transistor creates a virtual V_{dd} for the SRAM cells slightly lower than the supply voltage.

To mitigate the negative impact on SRAM cell speed due to an NMOS gated-V_{dd} transistor, we can use a wider transistor with a charge pump. To offset a wider transistor's increased leakage current, we further raise the gated-V_{dd} transistor's threshold voltage. The last row shows results for increasing the gated-V_{dd} transistor width by a factor of four and adding a charge pump that raises the active mode gate voltage to 1.35V. The resulting SRAM speed overhead is only around 8% compared to the low threshold voltage SRAM cells without gated-V_{dd}. Moreover, the reduction in standby mode energy is 97%.

4.2 DRI i-cache performance

We use SimpleScalar-2.0 [4] and SPEC95 to model an L1 DRI i-cache in an out-of-order microprocessor. The DRI i-cache is configured with a sense interval of one million instructions, a divisibility of two, and a miss-bound and size-bound for each benchmark chosen to keep execution time degradation within 4%. For each benchmark, we measure the relative execution time of a system with a DRI i-cache compared to a conventional cache and the effective DRI i-cache leakage and size as a percentage of a conventional 64K direct-mapped i-cache.

While a DRI i-cache reduces the average required cache size, it incurs overhead due to resizing and may affect execution time. Figure 4 shows relative energy-delay products comparing the leakage energy-delay of a DRI i-cache using the wide NMOS gated-V_{dd}, dual-V_t technique of Table 3 to that of a conventional i-cache. We use the analytical models developed by Kamble and Ghose [6] to estimate the extra L1 and L2 dynamic energy dissipation [11].

The figure also shows the average cache size for each of the benchmarks as a percentage of a conventional 64K cache. The figure indicates that a DRI i-cache decreases the average cache size significantly. A DRI i-cache reduces the cache size by an average of 62% while increasing execution time by less than 4%.

The benchmarks are grouped into three classes. The first class, ranging from *applu* through *swim*, primarily requires a small i-cache throughout execution. A DRI i-cache reduces the effective

Technique	Gated-V_{dd} V_t (V)	SRAM V_t (V)	Active Leakage Energy (nJ)	Standby Leakage Energy (nJ)	Energy Savings (%)	Relative Read Time	Area Increase (%)
no gated-V_{dd}, high V_t	N/A	0.40	50	N/A	N/A	2.22	N/A
NMOS gated-V_{dd}, dual V_t	0.40	0.20	1690	50	97	1.30	2%
NMOS gated-V_{dd}, dual V_t	0.50	0.20	1740	49	97	1.35	2%
no gated-V_{dd}, low V_t	N/A	0.20	1740	N/A	0	1.00	N/A
PMOS gated-V_{dd}	0.20	0.20	1740	235	86	1.00	0%
NMOS gated-V_{dd}, dual V_t wide, charge pump	0.40	0.20	1740	53	97	1.08	5%

Table 3: Energy, speed, and area of gated-V_{dd} techniques for one cell.

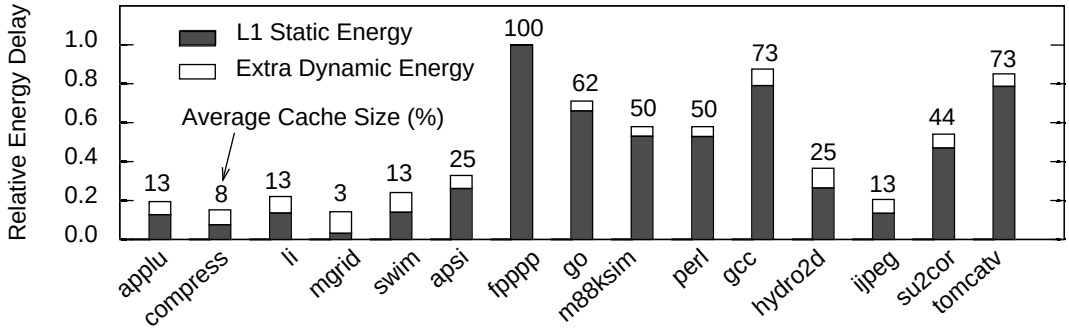

FIGURE 4: Relative energy-delay and average cache size in a DRI i-cache.

cache size to as low as 3% for *mgrid*. The second class of benchmarks are those that primarily require the full 64K i-cache throughout their execution and prevent a DRI i-cache from downsizing; they range from *apsi* to *perl*. *Fpppp* is an extreme example which cannot downsize at all without a large performance degradation. The last class of benchmarks exhibit distinct phases with diverse i-cache size requirements. Benchmarks from *gcc* to *tomcatv* are in this group, with average cache sizes from 73% to 13%.

Although we show the relative energy-delay products for the base DRI i-cache, a different energy-performance trade-off point can be chosen by adjusting the DRI i-cache parameters [11]. For example, a more aggressive miss-bound setting would significantly decrease the overall leakage energy but would have a larger impact on execution time.

5 Conclusions

This paper explored an integrated architectural and circuit-level approach to reducing leakage energy dissipation while maintaining high performance in deep-submicron cache memories. At the architectural level, a dynamically resizable cache resizes and adapts to an application's i-cache requirements during execution. We proposed a circuit-level technique, gated-V_{dd}, to gate the supply voltage to, and reduce leakage in, the SRAM cells in the unused sections of a dynamically resizable instruction cache.

We evaluated and presented simulation results from running the SPEC95 applications on a SimpleScalar model of a DRI i-cache. The results indicated that a 64K DRI i-cache reduces the energy-delay at best by 87% and on average by 62% with less than 4% impact on execution time. We evaluated and presented results on a spectrum of circuit techniques to implement supply voltage gating with varying leakage reduction, performance, and area overhead trade-offs. The results indicated that a wide NMOS dual-V_t gated-V_{dd} with a charge pump reduces leakage most with minimal impact on cell speed and area.

Acknowledgements

This research is supported in part by SRC under contract 2000-HJ-768, and DARPA under contract DAAH04-96-1-0222. This material is also based upon work supported under a National Science Foundation Graduate Fellowship.

References

[1] D. H. Alboensi. Selective cache ways: On-demand cache resource allocation. In *Proceedings of the 32nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 32)*, Nov. 1999.

[2] N. Bellas, I. Hajj, and C. Polychronopoulos. Using dynamic management techniques to reduce energy in high-performance processors. In *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, Aug. 1999.

[3] S. Borkar. Design challenges of technology scaling. *IEEE Micro*, 19(4):23–29, July 1999.

[4] D. Burger and T. M. Austin. The SimpleScalar tool set, version 2.0. Technical Report 1342, Computer Sciences Department, University of Wisconsin–Madison, June 1997.

[5] K. Inoue, T. Ishihara, and K. Murakami. Way-predicting set-associative cache for high performance and low energy consumption. In *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, pages 273–275, Aug. 1999.

[6] M. B. Kamble and K. Ghose. Analytical energy dissipation models for low power caches. In *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, Aug. 1997.

[7] J. Kin, M. Gupta, and W. H. Mangione-Smith. The filter cache: An energy efficient memory structure. In *Proceedings of the 30th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 30)*, pages 184–193, Dec. 1997.

[8] S. Manne, A. Klauser, and D. Grunwald. Pipline gating: Speculation control for energy reduction. In *Proceedings of the 25th Annual International Symposium on Computer Architecture*, pages 132–141, June 1998.

[9] L. Wei, Z. Chen, M. C. Johnson, K. Roy, and V. De. Design and optimization of low voltage high performance dual threshold CMOS circuits. In *Proceedings of the 35th Design Automation Conference*, pages 489–494, 1998.

[10] S. J. E. Wilson and N. P. Jouppi. An enhanced access and cycle time model for on-chip caches. Technical Report 93/5, Digital Equipment Corporation, Western Research Laboratory, July 1994.

[11] S.-H. Yang, M. D. Powell, B. Falsafi, K. Roy, and T. N. Vijaykumar. Dynamically resizable instruction cache: An energy-efficient and high-performance deep-submicron instruction cache. Technical Report ECE-007, School of Electrical and Computer Engineering, Purdue University, 2000.

[12] Y. Ye, S. Borkar, and V. De. A new technique for standby leakage reduction in high performance circuits. In *IEEE Symposium on VLSI Circuits*, pages 40–41, 1998.