IOWA STATE UNIVERSITY
Digital Repository

Graduate Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

2017

LowLEAC: Low leakage energy architecture for caches

Rashmi Parisa Girmal
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/etd

 Part of the Computer Engineering Commons

Recommended Citation
Girmal, Rashmi Parisa, "LowLEAC: Low leakage energy architecture for caches" (2017). *Graduate Theses and Dissertations*. 15308.
https://lib.dr.iastate.edu/etd/15308

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

www.manaraa.com

LowLEAC: Low leakage energy architecture for caches

by

Rashmi Girmal

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Engineering

Program of Study Committee:
Arun K. Somani, Major Professor
Zhao Zhang
Steven J. Hoff

The student author and the program of study committee are solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2017

Copyright © Rashmi Girmal, 2017. All rights reserved.

www.manaraa.com

TABLE OF CONTENTS

www.manaraa.com

www.manaraa.com

LIST OF TABLES

www.manaraa.com

LIST OF FIGURES

www.manaraa.com

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this thesis. First and foremost, my adviser, Dr. Arun K. Somani for his guidance, patience and support throughout this research and the writing of this thesis. His insights and words of encouragement have often inspired me and renewed my hopes for completing my graduate education. I would also like to thank my committee members for their support: Dr. Zhao Zhang and Dr Steven J. Hoff. I would like to thank my husband, Sohan Kale, for his support and encouragement at all times. I would also like to thank my mother and sister for their faith in me. Lastly, many thanks to all the friends and lab-mates who were a part of my graduate school journey.

www.manaraa.com

ABSTRACT

With the ever-decreasing feature sizes, static power dissipation has become a concern in computing devices. On-chip memories are a major contributor towards the processors leakage power dissipation due to their large transistor count. We propose a Low Leakage Energy Architecture for Caches, called LowLEAC to minimize the static power dissipation in caches made of CMOS SRAM cells. This technique is based on keeping only k most recently used cache lines powered on other lines powered off to reduce the leakage power dissipation. The control however increases the dynamic power due to re-fetching of data. To overcome that, we deploy CMOS compatible nonvolatile SRAM cell, called cNVSRAM, to implement caches. The cNVSRAM cell works as a conventional SRAM in the regular mode and saves the data in a non-volatile back up when a cache line is turned off or put in the sleep mode. The non-volatile back up mode helps improve the dependability of the cache and avoids the penalty occurred due to loss of data from the inactive cache lines. With a small area penalty, LowLEAC achieves 18% energy savings with insignificant impact on the performance. LowLEAC is a suitable architecture for cache memory in mobile computing devices to minimize battery power consumption and reduce heat.

www.manaraa.com

CHAPTER 1. INTRODUCTION

Increasing power dissipation is a major concern in both mobile computing systems as well as high performance machines. Along with dynamic switching power, leakage power dissipation is becoming a grave issue as the feature size shrinks further. With the decreasing feature size and increasing transistor density, leakage power of CMOS circuits is increasing. Leakage power can contribute up to 25% of the total power consumption of CPUs. Hence power aware computing is the need of the day. Earlier power constraint was mainly applicable to mobile computing devices but in today's times, even desktop and server system designs need to be power aware to minimize the energy and cooling costs involved. A large share of the processor chip's area and transistor count is dedicated to on-chip SRAM memory based structures. The size of these structures like caches, lookup tables and storage buffers has been increasing to achieve better performance. All these SRAM-based arrays have a significant share in the chip's power consumption. Along with the dynamic or switching power, the static or leakage power consumption of SRAM memory is significantly higher in modern day chips. These SRAM based caches are a key target to implement energy efficiency techniques [3].

1.1 The Power Problem

Processor power consumption can be divided into two major categories: dynamic or switching power and static or leakage power. Traditionally, dynamic power consumption has been the major limiting factor in the design of processors. The power consumed in switching the state of transistors has formed the majority of the overall power consumption. But in the modern generations of computing systems, static power is also becoming a major design challenge. Static power or leakage power is directly proportional to the number of transistors on chip and

www.manaraa.com

calculated every cycle [1]. Thus the problem of leakage power is growing with the increasing transistor count and decreasing feature size. While dynamic power can be controlled by reducing the operating frequency and supply voltage, there is a need to control the increasing static or leakage power in modern day systems. The 2013 International Technology Roadmap for Semiconductors (ITRS) poses power management as one of the grand challenges in the near term. With the transistor count doubling every generation, implementation of techniques to contain the leakage power becomes important in order to reduce the energy and cooling costs [4]. Earlier, performance was the major goal behind system design. But recently with the wide spread use of mobile computing, power dissipation also needs to be given priority. Maximizing the battery life for mobile devices and reducing energy and cooling costs for server systems is a known need. Past research like [3], [5], etc. has shown that significant savings can be achieved by reducing leakage power dissipation.

1.1.1 Dynamic Power Consumption

Dynamic power consumption in digital devices is given by the following equation:

$$P_{dynamic} = \alpha \times C \times Voltage \times f \tag{1.1}$$

where, α is the activity factor and f is the operating frequency.

Dynamic power depends upon the switching activity factor, supply voltage and operating frequency. Dynamic power reduction has been achieved by techniques like dynamic voltage and frequency scaling (DVFS) which reduces the supply voltage and frequency to control the power consumption.

1.1.2 Static Power Consumption

Static power is the power dissipated due to the leakage current that keeps flowing even when the transistors are off. The following formula gives the static power consumption:

$$P_{static} = Voltage \times N \times k \times Current_{static} \tag{1.2}$$

where N is the number of transistors, k is a design dependent parameter and $Current_{static}$ is the subthreshold leakage current that flows through the transistor when it is off. It is proportional

www.manaraa.com

to the threshold voltage of the transistor and the operating temperature and given by the following equation:

$$Current_{static} \propto e^{(-Vt/T)} \tag{1.3}$$

where Vt is the threshold voltage of transistors and T is the operating temperature.

1.2 Leakage power in caches

The impact of increasing transistor counts and higher clock frequencies on dynamic power consumption is mitigated by lowering the power supply voltages. In order to match the faster switching speeds, the threshold voltages of transistors also need to be lowered. This results in the increase in leakage current and static power consumption. Due to the exponential dependence, a small decrease in the threshold voltage leads to large increase in the leakage current and power. Thus, the ever decreasing feature size and threshold voltages are further aggravating the leakage power dissipation issue.

Since the leakage power is directly proportional to the number of transistors on chip, it makes sense to target the architectural component taking up the most area on a processor chip. Caches, thus, become a key target to implement leakage power reduction techniques. Methods have been proposed in the past which turn off the unused cache sets or ways to reduce leakage power, but it comes with its own performance penalty.

LowLEAC operates at a finer granularity and turns off individual cache lines. It aims to reduce leakage power consumption without significantly altering the performance or the dynamic power consumption. Turning off cache lines leads to loss of data contained in them resulting in a miss. Not only does this affect the performance by investing cycles in accessing the higher level caches, but also leads to increase in dynamic power consumption. Also, it is equally important to check that, the dynamic power consumption due to a miss does not counter the static power saved by implementing a power reduction technique (like turning off cache lines). Hence, correctly choosing the cache lines to be turned off becomes vital. Earlier studies have given us details on states of data during its lifetime in a cache line [6, 7]. Based on these, we can say that for a considerable amount of time, a cache line contains data that

www.manaraa.com

may not be used in the near future. It is advantageous to turn off such lines which are dead or contain outdated data.

1.3 Our solution

This work presents LowLEAC, a low leakage energy architecture for caches. It explores turning off a portion of the cache by putting it into sleep mode as a way to reduce leakage. The fact that not all cache lines contain data that will be reused in the near future [6] can be exploited to save leakage power. In the proposed technique, only k most recently accessed cache lines are powered on and the rest of the cache powered off or in a sleep mode. The line numbers of cache lines that are on the order in which they have been accessed is maintained in a buffer with the most recently accessed line at the top and the least recently accessed at the bottom. LowLEAC switches the least recently accessed lines off, as they make their way out of the buffer, which are likely to be evicted anyway according to the least recently used (LRU) replacement policy. This keeps only the lines containing active data powered on thus saving leakage power with minimal performance impact.

The data lost due to turning off lines increases the amount of capacity misses. Hence, there is a need to reduce the penalty associated with capacity misses occurring due to the data lost by turning off cache lines. The dynamic power consumption by the cache is kept to a minimum by using CMOS compatible cNVSRAM memory cells [2]. Whenever a line is turned off, the data is saved in the non-volatile part of the cNVSRAM cell and can be retrieved from here, in case it is needed in the near future. This helps us to increase our power savings and reduce our performance penalties further. These cells have ultra low leakage power consumption and incur less penalty in terms of power. These cells work in fast mode when a cache line is turned on and in use. When a line is turned off, the data in those cells is backed up in a non-volatile part of the cell and power supply to it is switched off to eliminate the leakage power dissipation. When the processor tries to access this line again, it can access the data from the non-volatile back up. Thus even if the processor attempts to access data from a cache line turned off recently, it will incur less performance penalty than incurred by trying to access data from the next level in the cache hierarchy. This technique aims at maximizing the power savings yet keeping

www.manaraa.com

the performance and area penalties as low as possible. With the decreasing technology sizes and increasing processor speeds, the overheads become a negligible cost to pay for achieving significant power savings.

1.4 Our contribution

We make the following key contributions.

- To reduce the leakage power consumption in data caches, we keep only 'k' most recently used cache lines turned on and the others stay powered off.
- We develop a queue mechanisms to keep track of the 'k' active cache lines in a queue in the access order. This allows to decide which line to turn off when a new line is brought in the cache.
- Data from the lines that are turned off is lost and needs to be fetched again if needed. This leads to extra dynamic power dissipation which brings down the potential energy savings by about 44%. Thus even with an optimal queue size, just turning off lines only impacts cache power consumption by only 1-2%.
- To regain the loss dues to the extra dynamic power consumption, LowLEAC uses cNVSRAM, a CMOS compatible non-volatile SRAM cell in caches. The data from the powered off lines are backed up in the non-volatile part of the cNVSRAM that are supported by charge pumps to maintain the data.
- LowLEAC with smaller queue sizes exhibits higher transitional energy overhead or higher energy consumed by charge pumps. Thus the optimal queue size becomes an important parameter to save the maximum leakage energy savings. We evalaute the optimal queue sizes for various applications.
- With an optimal queue size, LowLEAC achieves 18% energy savings on average as compared to an L1 data cache with no leakage reduction technique. These savings are achieved at the cost of 25% area overhead of the cNVSRAM memory cells.

www.manaraa.com

1.5 Thesis Organization

The remainder of this thesis is organized into the following chapters.

- Chapter 2 gives background on cache memory, the structure of SRAM memory cell and leakage current in it. It also explains techniques that have been implemented so far to reduce the leakage power consumption in caches.
- Chapter 3 explains the design and architecture of LowLEAC, implemented to reduce leakage power consumption. It also discusses its impact on the performance and the steps taken to minimize it.
- Chapter 4 describes the simulation methodology used to implement LowLEAC. It also states the results obtained by simulation experiments and presents an analysis of the energy efficiency achieved and the costs incurred.
- Chapter 5 summarizes the work described in this thesis.

www.manaraa.com

CHAPTER 2. BACKGROUND AND RELATED WORK

This chapter presents the basic information on cache. Since the introduction of the first microprocessor from Intel in 1971, computers have evolved continuously. With the advances in semiconductor technology, the size of devices as well as computing units has been decreasing. However, the improving performance of the processors was not matched by the performance of the DRAM. As shown in Figure 2.1, processor speeds have increased at a much faster rate than memory access speeds. This is a barrier in achieving higher performance which is also known as the *Memory Wall* [8]. In order to overcome this gap between processor and memory speeds, on-chip cache memories are used to store the data frequently accessed by the processor.

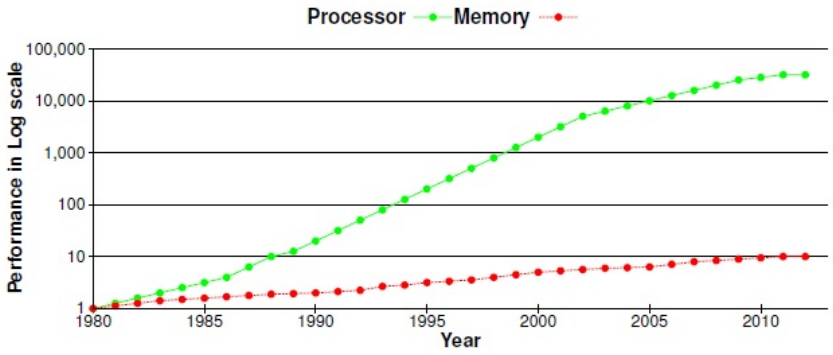

Figure 2.1 Performance gap between processor and memory[1].

2.1 Introduction to Cache Memory

Caches are designed to have a hierarchical structure and placed in between the processor and main memory. The cache closer to the processor is optimized for speed and performance while the caches further away are optimized for area or energy. Cache memory covers a major section of the chip's area and contributes largely to the chip's transistor count. Each bit of data

www.manaraa.com

is stored using a typical CMOS six transistor SRAM memory cell. Due to the high transistor count, caches are also a major contributor towards the static power consumption of the chip. According to Moore's law[9], the transistor count per area doubles every eighteen months. With increasing number of transistors on chip, leakage power or static power has become increasingly significant [3].

2.1.1 Leakage Current in SRAM Memory Cells

The leakage power in CMOS circuits is dissipated because of the leakage current that flows through the transistors. The transistors basically act as imperfect switches and hence, a small amount of current flows through them even when they are turned off.

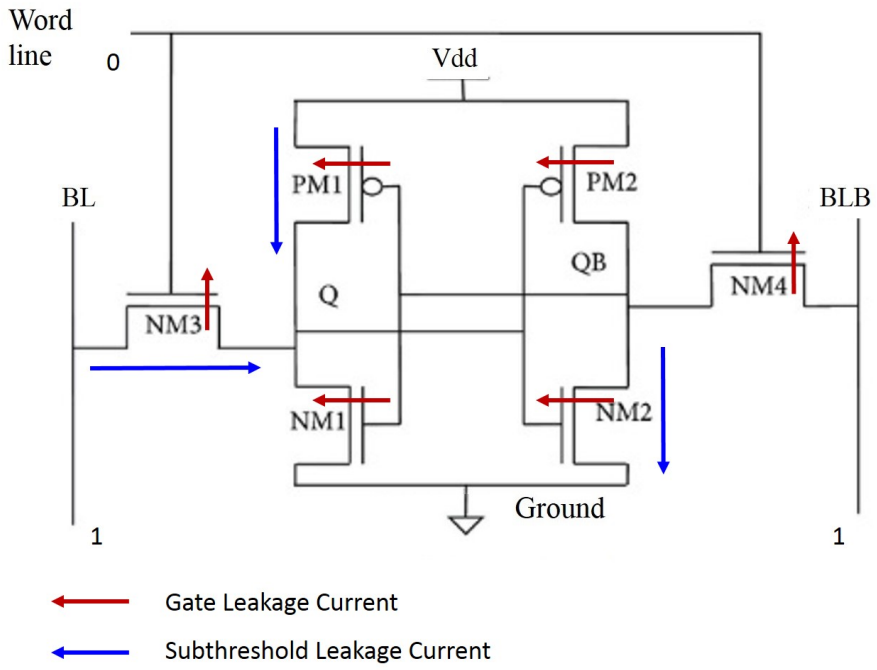

Figure 2.2 Subthreshold and gate leakage currents in an idle 6T SRAM cell.

As shown in Figure 2.2, two types of leakage currents flow through a 6T SRAM cell when it is turned off. The subthreshold leakage current is a small current that passes between the source and drain of the transistor when it is in off state. On the other hand, gate leakage is the current that leaks through the gate terminal. An SRAM memory cell in the idle state

www.manaraa.com

has multiple paths for leakage current to flow. Hence, leakage power consumption has become a serious problem in SRAM memory structures like caches. Subthreshold leakage has been a problem since 90nm technology while gate leakage is also becoming a serious issue below 45nm technology[10]. Designers are lowering the supply voltages to achieve faster transistor switching but lower supply voltages result in increasing the leakage currents further.

2.2 Related Work

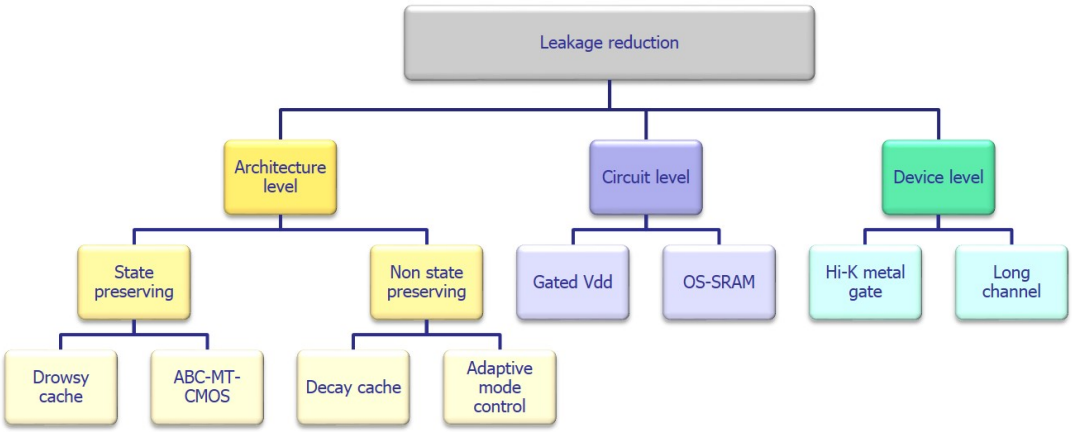

Figure 2.3 Classification of leakage power reduction techniques.

A lot of research has been conducted in reducing the leakage power in SRAM memory structures. Power saving techniques have been implemented at architecture level, circuit level, and device level. Previous techniques can be briefly classified as shown in Figure 2.3. Architectural level techniques like decay cache[3], adaptive mode control[11], and DRI cache[12] put individual or a group of cache lines into a low power or sleep mode such that almost no static power is drawn. These techniques fall in the category of non state-preserving techniques as the data in the cache lines is lost when they are powered off or put into low power mode.

On the other hand, state-preserving techniques proposed in [5], keep data intact and avoid the penalty of next level cache access time in case of an induced miss. These architectural level

www.manaraa.com

techniques use one or more circuit level implementations to power off lines or put them into low power mode. Circuit level techniques like gated-V_{dd}[13] used to turn off lines, insert a sleep transistor that disconnects the memory cells from ground when turned off and avoid the flow of leakage current. Other techniques like dual-Vt[14] and MT-CMOS[15] modify the transistor threshold voltage either statically or dynamically to increase Vt and thus reduce the leakage current, maintaining the state of the lines.

Recently, device level techniques like long channel devices and Hi-K metal gate devices are being used to control the leakage current density and thereby reduce the leakage power consumption. Some of the major techniques are discussed below.

2.2.1 Cache Decay

Kaxiras et.al.[3] have proposed a technique to reduce leakage power consumption by turning off cache lines that contain inactive or dead data. They propose the use of global and local counters to keep track of the number of cycles since the last access of a particular cache line. If the counters exceed a set value, called the decay interval, the particular cache line is considered dead and turned off as the data in it is not likely to be used again. They also implement certain adaptive algorithms to optimize the decay interval to achieve 4 to 5 times reduction in leakage power consumption with minimal performance loss. This method tries to strike a balance between the leakage power saved by turning off lines and the dynamic power and performance overheads due to induced misses. However this technique can prove to be costly because of the area and power consumed by counters for each cache line as well as the complex and computationally intensive adaptive algorithms used.

2.2.2 Adaptive Mode Control

The adaptive mode control cache design[11] proposed by Zhou et.al. is similar to decay cache but it turns off only the data part and keeps the tag part of cache lines on. This helps to find the hypothetical miss rate or the miss rate if all cache lines were kept on. This method uses this miss rate as a bound to turn off lines and save any extra induced misses that would occur due to turning off cache lines. The decay interval or turn-off interval is dynamically adjusted

www.manaraa.com

so as to achieve the hypothetical miss rate bound. This reduces the power and performance penalties incurred in fetching data from the next level of cache. This method also implements local and global counters which contribute to the overhead in terms of chip area and power. The tag part of cache lines, which are kept on, keep dissipating leakage power and the dynamic miss rate calculation further contributes to the overheads.

2.2.3 Drowsy Cache

Drowsy cache is a state-preserving technique proposed by Flautner et. al.[5] where cache lines that are inactive are put into a low power or drowsy mode where they retain the data stored in them. Dynamic voltage scaling (DVS) is used to scale the threshold voltage of the transistors by 1.5 times. This helps in reducing the leakage current, keeping the state of the transistors intact. However, the memory cells must be reinstated to the high power state before the data in it can be accessed again. This results in a slight performance penalty but certainly lesser than that involved in state-losing techniques. Drowsy caches are not able to significantly reduce the leakage power as the lines are not completely shut down and there is still some leakage current flowing, resulting in power dissipation. The changing threshold voltages may also lead to a noisy environment, increasing the error probability and making the data less reliable.

2.2.4 MT-CMOS

The multithreshold CMOS (MT-CMOS) technique[15] reduces leakage current by dynamically raising the transistor threshold voltage thus reducing leakage power consumption. It uses an additional power supply V_{dd+}, higher than the voltage level at the source terminals to bias the p-channel transistors in sleep mode. Increasing the source-substrate voltage potentials leads to higher Vt for pFETs. Also, the nFET source-substrate potential voltage is increased due to diode drop voltages, thus increasing the effective Vt for them. The raised threshold voltage of all transistors results in a decreased leakage current and reduced leakage power. The transistor supply voltage has to be reinstated to V_{dd} in order to access the data in those lines again. The main advantage of this method is that data is retained when the lines are put into

www.manaraa.com

sleep mode. Nii et. al. proposed the auto-backgate-controlled MT-CMOS technique[16] which automatically controls the backgate bias to increase the threshold voltage when put in sleep mode. This technique reduces the leakage power without incurring the overhead of time and energy to fetch data from next level of memory. A major disadvantage is implementing an extra power supply voltage and distributing it throughout the chip. Other cons include the latency penalty of awakening the line from sleep mode and the electric fields across the transistors in sleep mode that may affect the reliability of data.

2.2.5 Long Channel and Hi-K Metal Gate Devices

Traditionally used SiO_2 gate oxide devices are becoming difficult to scale because of the increasing gate oxide leakage as the SiO_2 thickness decreases. As the thickness scales down further, leakage currents due to tunneling increase drastically, leading to high power consumption. Replacing the SiO_2 gate dielectric with a high-K material allows increased gate capacitance without the associated leakage effects. In order to ensure high performance and low leakage power in CMOS technologies below 45nm, high-K gate dielectric and metal gate electrodes are used[10]. Long channel devices have 10% longer channels than normal devices at the same technology node. Hi-K devices reduce the gate leakage by more than 10 times and long channel devices reduce the subthreshold leakage by upto 3 times[17]. Intel's 45nm technology architecture(*Penryn*) uses hi-k metal gate devices to control leakage power maintaining high performance levels[18].

2.2.6 Gated-V_{dd}

An extra transistor called the gated-V_{dd} transistor or sleep transistor is introduced in the path to ground of the cache's SRAM cells[13]. This transistor is turned off for sections that are not in use and turned on for active cache sections. Figure 2.4 shows a basic 6 transistor SRAM cell with sleep transistor. The gated-V_{dd} transistor along with the SRAM transistors produces the stacking effect in transistors[19] thus reducing the leakage current flowing through them. This method reduces leakage power consumption still maintaining the performance advantages of lower supply and threshold voltages. Using a PMOS or an NMOS as a sleep transistor has

www.manaraa.com

its own trade offs. Using an NMOS gives larger energy savings but the width of the NMOS has to be large enough incurring higher area overheads. On the other hand, a PMOS transistor uses less area but cannot provide as much leakage energy saving as an NMOS transistor. The only disadvantage here is that the data in the part of the cache turned off is lost. Thus it incurs in an extra penalty of fetching data from next level memory when a miss occurs due to turning off cells.

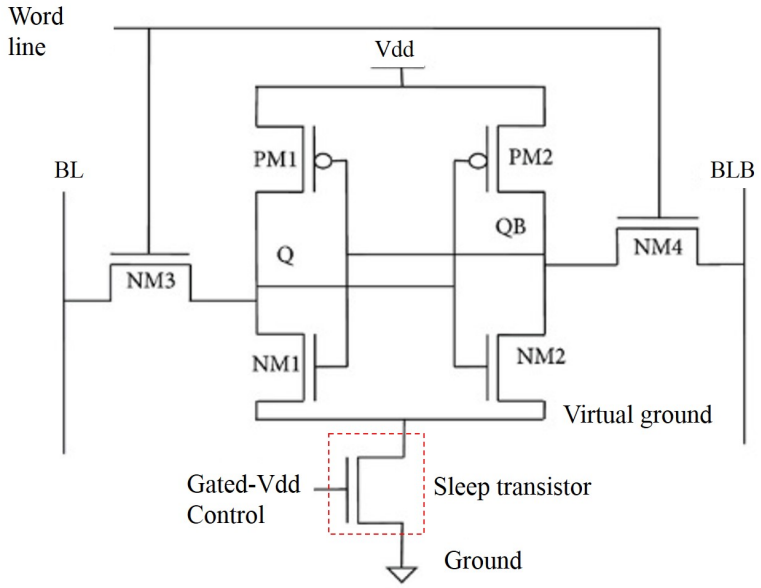

Figure 2.4 Conventional 6T SRAM cell with gated-V_{dd}.

Leakage control at architectural level proves to be beneficial as a group of circuits can be controlled at once. The state-preserving techniques claim to be better because of the low performance overhead [5, 15]. But recent studies prove that the energy savings by non-state-preserving techniques are much higher. The performance overhead can be minimized with faster switching speeds and identifying the proper decay interval using run-time adaptability[20].

This thesis proposes LowLEAC, a non-state-preserving architecture level technique that keeps only selective lines powered on and rest of the cache is powered off. The data from powered off lines is stored in a non-volatile SRAM cell when turning off the line and recovered when the line is turned back on. This helps in reducing the performance penalty to fetch lost data from higher level caches. Details of the LowLEAC design are presented in Chapter 3.

www.manaraa.com

CHAPTER 3. DESIGN AND ARCHITECTURE

In cache memory systems, not all the lines contain active data that would be used in the following cycles. This is the main motivation behind turning off cache lines that have not been used in the recent past. Wood et. al. [7] define the life time of a cache block in terms of *live time* and *dead time*. If the next access of the block is a hit then it is considered *live* else it is considered *dead*. They provide evidence that on average, around 30% of the cache blocks are *dead*. Thus if we could predict which blocks or lines are *dead* and turn them off, we could save a significant amount of static power. Our goal in this research is to identify the cache lines that are in the *live* state and keep only those lines powered on.

Using their oracle predictor, Kaxiras et. al. [3] stated that in an ideal case, on average, a cache line is in the *dead* state for around 65% of its total life span. This inspires us to power off most of the cache lines and keep only a small portion of them powered on. This approach can be used at the block level or the line level granularity. We use the k most recently used (k-MRU) technique to keep an account of the most recently used cache lines. As the time from the last access to a cache line increases, the probability of it becoming dead increases. As more and more lines are accessed, the cache lines containing older obsolete data are not likely to be used in the future. Thus they can be turned off.

Most previous techniques[3, 5] deploy counters to keep track of the access pattern of each cache line. This proves to be costly in terms of chip area especially as the counters are implemented on a per line basis. The higher the number of lines, more area is used for implementing counters. For example, if a cache with 2048 lines is using a counter based technique would need more than 4 kB of space considering a 2 byte counter for each line and additional logic to maintain them. The technique that we propose uses only $k \log N$ bits to store the additional information where N is the number of cache lines. The value of k, (k ¡¡ N), can be selected on

www.manaraa.com

the basis of cache size, number of cache lines, and the acceptable performance penalty. In this chapter, we present our scheme and results. More details of our simulation environment are presented in Chapter 4.

3.1 LowLEAC

Our approach utilizes a fixed number of cache lines to remain active. Our approach differs from previous approaches in such that we only keep the k most recently accessed lines powered on instead of turning off few least recently used lines. In this way, we always have only a fixed number of lines powered on saving a significant amount of leakage power. Also we save a considerable amount of chip area by avoiding counters for each cache line. Instead, we use a fixed size queue to keep track of active lines. Unlike counters in previous approaches that update every cycle, our queue updates only during cycles when an inactive or turned off cache line is accessed. This helps us to reduce performance overhead involved in handling the meta data of the leakage reduction technique.

3.1.1 Active lines queue management

The active cache lines queue to maintains k entries in the order in which the cache lines are accessed. Figure 3.1 shows the process of updating the queue at every cache access. If the line accessed in the current cycle is present in the queue, it is brought to the top of the queue and if it is not present in the queue, it indicates that the line is off. In this case, the newly accessed line is turned on, the line number is added to the top of the queue and all the lines are shifted down by one position. If the queue is full, the line at the bottom of the queue is moved out of it and also turned off. Only the lines present in this queue remain powered on and the rest of the cache lines remain powered off.

3.2 Performance Considerations

The data in a cache line is lost when it is turned off or put into sleep mode. This results into increasing the capacity misses and there is a performance penalty associated with it. This

www.manaraa.com

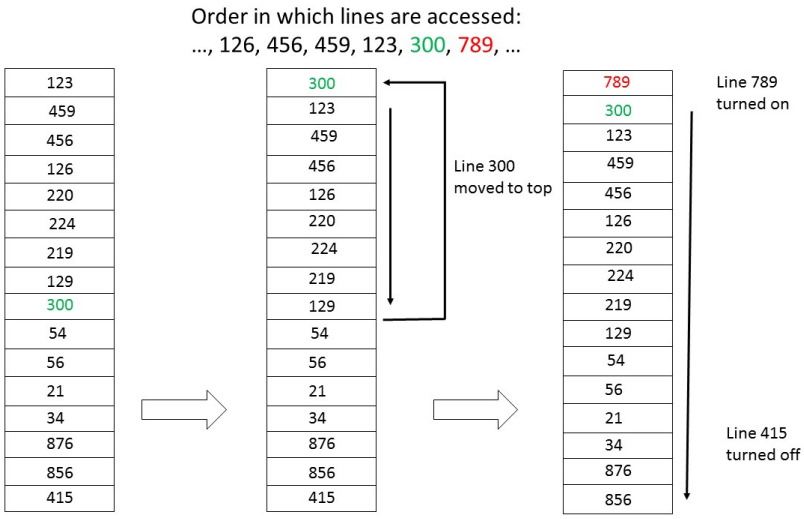

Figure 3.1 Updating the list containing active cache line numbers.

may also result in extra dynamic power consumption to access data from the next level cache. The latter problem is addressed further in the thesis.

Figure 3.2 shows the comparison of miss rates of caches with and without turning off lines. In the first experiment, the cache under study is a L1 data cache of 64 kB size and 32 byte line size. Out of the 2048 cache lines, we keep only 64 most recently used lines ON and turn off the rest of them. The data in the lines turned off is lost or corrupted due to the absence of power. Hence the miss rate increases as a result of our leakage reduction technique. We observe that the cost is more for memory intense applications, like crafty-mem, whereas it is minimal for other applications, like art.

The performance penalty and power trade off can be managed by varying the number of the cache line and the queue containing the active cache line numbers. Figure 3.3 shows the miss rate of various spec 2006 benchmarks when 64, 128 and 256 lines turned on in the L1 data cache consisting of 2048 lines. The miss rate decreases as the number of active lines increases. The miss rate for a cache in which 256 lines are turned on is almost the same as that of keeping the entire cache powered on. Thus we can save a considerable amount of leakage power by keeping 87.5% of the cache turned off, without having a significant impact on the performance. In some applications like mcf, as low as 64 lines out of 2048 are kept on. This results in good

www.manaraa.com

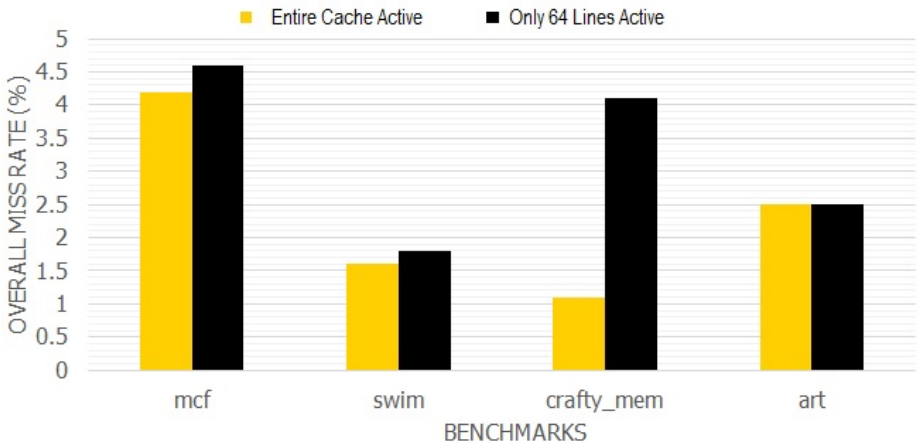

Figure 3.2 Over all miss rate.

performance which indicates that a lot of leakage power reduction can be achieved by turning off cache lines for such applications.

Please note that our results do not suggest that smaller sized L1 caches can be deployed. In case of smaller caches, there will be contention conflict and capacity misses, which will have significant performance penalty. Also note that k can be chosen based on the application being executed.

Section 4.1 analyzes the dynamic and leakage power reduction achieved by different sizes of the active cache lines list.

3.3 LowLEAC vs smaller caches

LowLEAC essentially keeps only a certain portion of the cache active at any given time. Thus it gives a preliminary idea similar to that of using a smaller sized cache to save leakage power dissipated. But there is a major advantage of turning off lines from a bigger cache than using a smaller cache. Using a smaller cache definitely reduces the leakage power but it also increases the amount of conflict misses. For example, if we consider a 4-way set associative cache of 512 lines and keep only 128 lines turned on (as per LowLEAC design) we can still have line numbers 0, 127, 255, 383 turned on at the same time. On the other hand if we use a cache of only 128 lines all the above line numbers will be mapped to the same physical line in cache

www.manaraa.com

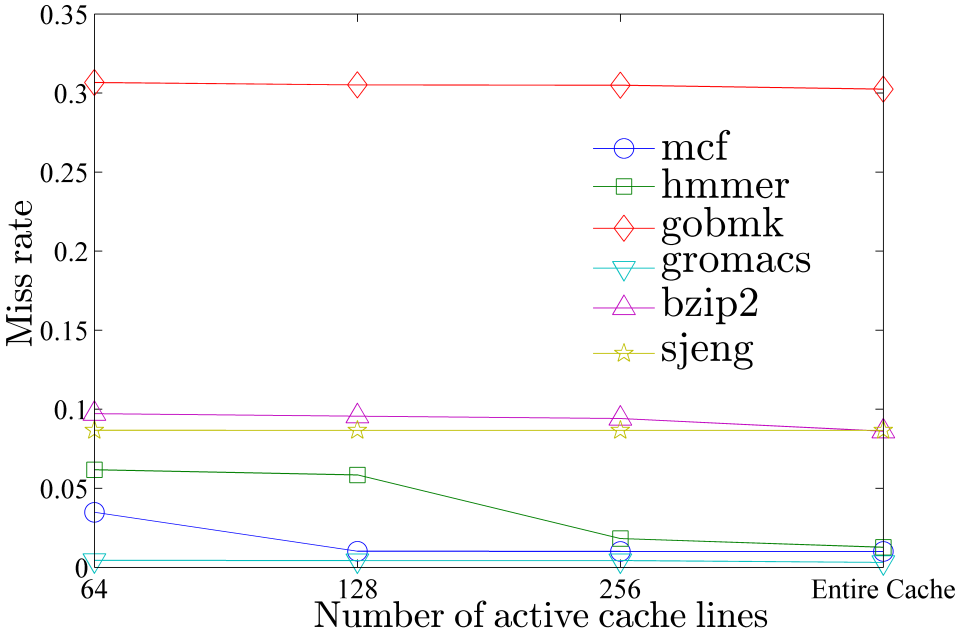

Figure 3.3 L1 data cache miss rate for various sizes of active cache lines list.

and will result in conflict misses if accessed in near succession.

Figure 3.4 shows the miss rates in a smaller cache made of 64 lines versus the miss rate in a larger cache of 2048 lines where only 64 lines are maintained active using LowLEAC technique. In case of all the benchmarks, the bigger cache with fewer cache lines turned on has a far smaller miss rate than a smaller cache. Hence, we recommend to have a larger cache and keep limited cache lines turned on instead of using a smaller cache. In this way we achieve the leakage reduction advantages of a smaller cache without compromising performance.

Our simulation results also suggest that, the size of the active cache lines can be adapted to achieve the best trade off between power savings and performance loss for a given application. For example, a system may include k=256 implemented in four concatenable queues of size 64 each.

3.4 Back up data in cNVSRAM cells

The extra dynamic power dissipation to fetch data from the next level of cache is a major overhead of turning off cache lines. Figure 3.5 shows the total power saving for various

www.manaraa.com

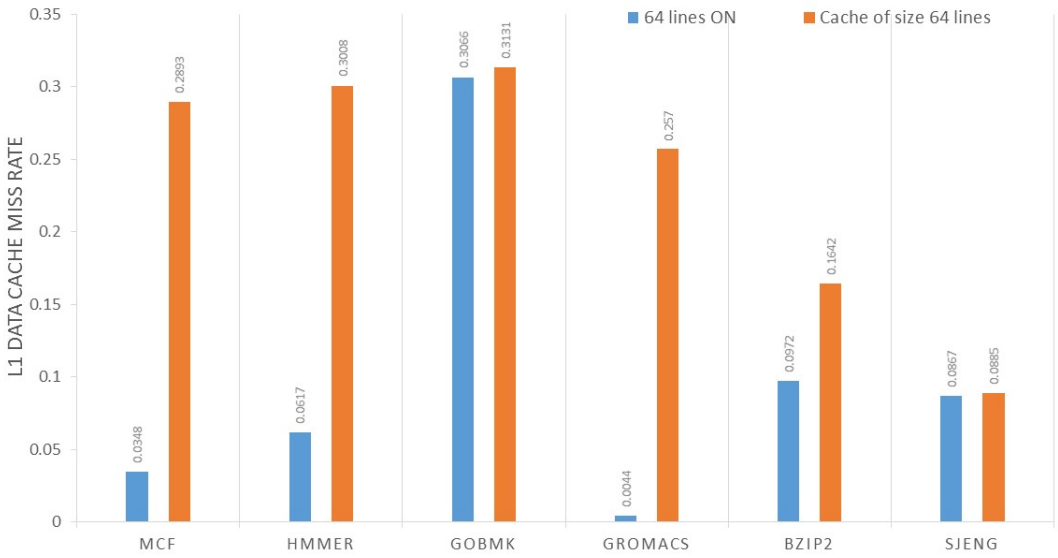

Figure 3.4 L1 data cache miss rate for turning off lines from a bigger cache vs a smaller cache.

spec2006 benchmarks implementing the LowLEAC technique. The blue portion depicts the actual leakage savings achieved while the orange portion depicts the extra dynamic power that was consumed to fetch data from the next cache level as well as the power used in switching of the cells to and from sleep mode. It shows that the potential energy savings that can be achieved are reduced because of this extra dynamic power.

3.4.1 cNVSRAM memory cells

Better power saving could be achieved by avoiding this access of the next cache level by backing up data in a non-volatile storage when a line is turned off. To achieve this, we adopt a cache made with CMOS technology compatible ultra low leakage energy non-volatile SRAM cells proposed by Wang et. al.[2]. The cNVSRAM cache helps in reducing the performance penalty due to capacity misses and consumes less power as compared to conventional cache structures. In the normal operation mode, cNVSRAM uses a conventional 8T SRAM cell for its operation giving the same performance and dissipating leakage energy the same as an 8T

www.manaraa.com

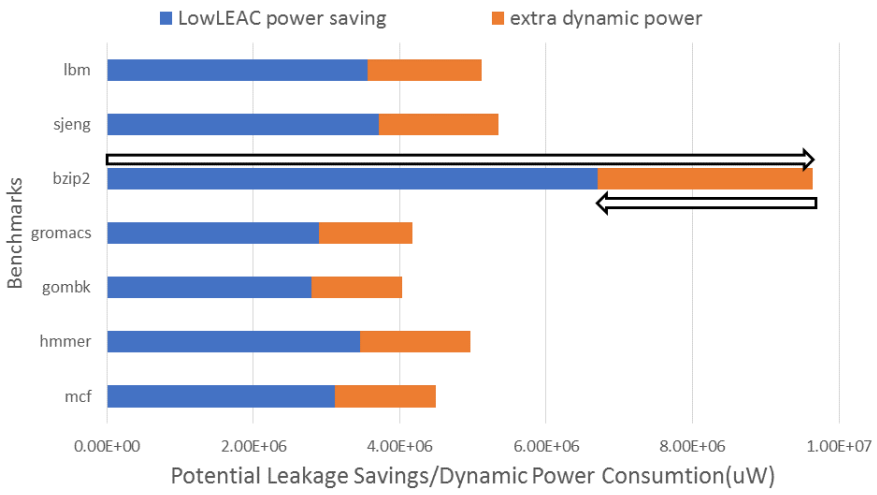

Figure 3.5 Reduction in the power saving caused due to extra dynamic power consumption.

SRAM cell. When a line is turned off, its cells are put into sleep mode and the data is backed up in the non-volatile portion of the cNVSRAM cell. Thus the leakage power dissipation in unused lines is reduced by turning off the power supply to them. At the same time, the performance penalty and extra dynamic power overhead due to data loss, is completely avoided with the back up in the non-volatile part of the cell.

Figure 3.6 shows the structure of a cNVSRAM memory cell with the non-volatile storage highlighted [2]. The control signal generated by the buffer control logic (see Figure 3.7), drives the cells into wake (WAK) and sleep (SLP) modes. When a cache line is turned on, a pulse is sent to wake up the cells in that line. In the working mode, the cNVSRAM cells work similar to conventional 8T SRAM cells with similar performance and power dissipation. In sleep mode, the data is backed up in the non-volatile part and the power supply to the transistor is completely shut off avoiding any leakage power dissipation.

www.manaraa.com

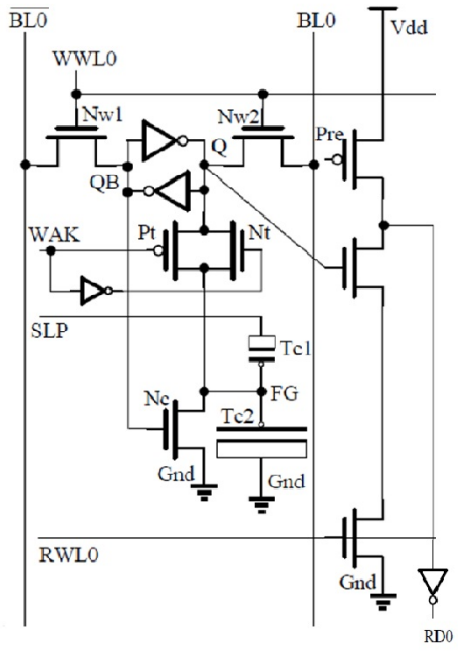

Figure 3.6 cNVSRAM cell structure [2].

3.4.2 cNVSRAM timing and area overheads

As stated by Wang et. al. [2], the only performance overhead incurred is the extra write time due to the delay caused by the extra capacitance that makes up the non-volatile part. Read operations are not influenced by this capacitance and therefore, does not incur any penalty in the access time of the cell. The access time of typical memory SRAM cells is much larger than the write time, thus making the write time a less dominant factor in the timing overhead.

A charge pump is used to maintain the voltage levels of the non-volatile data back up. The charge pumps and the capacitors in cNVSRAM cells cause an area overhead as compared to conventional 8T SRAM cells. As mentioned in [2], this overhead is reduced to a minimum with compact routing and reduces even further for multi-ported memory structures. As compared to conventional cache, the cells and charge pumps cause about 25% and 3.3% area overhead, respectively. For the achieved energy savings, this overhead is tolerable. Details about the access times and area of cNVSRAM are given in Chapter 4.

www.manaraa.com

3.4.3 Why cNVSRAM?

cNVSRAM cells are more effective for larger memories with more sleep time. The power supply to the transistor is completely shut off during the sleep mode. Energy savings will be achieved if the sleep time of the cell is larger than the time taken to back up the data. The larger the sleep time, the more energy savings achieved. In LowLEAC architecture, a significant portion of the cache remains turned off at all times and will be turned on only when data is accessed again. This elevated sleep time of LowLEAC cache lines extracts the maximum energy savings from cNVSRAM cells. The non-volatile back up also avoids the overhead of dynamic power to fetch lost data from next cache level. Thus cNVSRAM kind of cell is most appropriate to enhance the leakage energy savings achieved by LowLEAC technique.

3.5 LowLEAC architecture

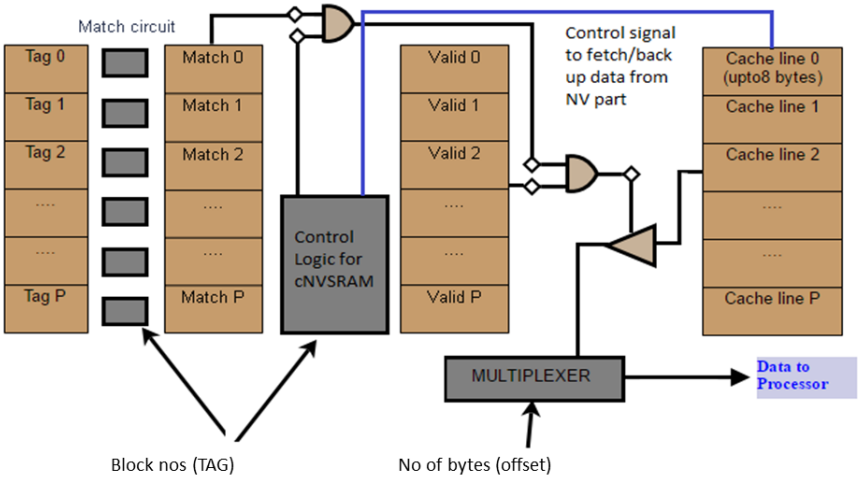

Figure 3.7 Basic block diagram of LowLEAC architecture.

Figure 3.7 shows the architecture in detail. The blocks in thick gray boxes show any logic that was added/modified from the basic cache architecture in order to implement LowLEAC.

www.manaraa.com

The cache shown is an L1 data cache implemented using the cNVSRAM cells [2]. The control logic for cNVSRAM consists of the buffer with active cache line numbers. This block is implemented using shift registers and a look up table to find the required line number from the buffer.

When the tag from the address matches a tag in the cache, the match flag is set to high. It should be noted that the tags remain powered on at all times. The tag is also used by the control logic block to derive the line number and check if it is present in the active lines buffer. The control logic generates a signal to indicate if the desired line is active or not. It also sends signals to wake up inactive lines and back up data from lines being turned off. This control signal also activates the charge pump that keeps the data backed up in the non-volatile portion of the cell. Thus the control logic block is responsible for sending wake up and sleep pulses to the cNVSRAM cells. Using the match flag, control logic output and valid bit, the data from the desired cache line can be read and transferred to the processor.

www.manaraa.com

CHAPTER 4. SIMULATION RESULTS AND ANALYSIS

This chapter describes the simulation methodology used to evaluate the design of LowLEAC caches. It also analyzes the area and power costs involved with the LowLEAC implementation and compares them to the achieved power savings.

4.1 Simulation Methodology

A modified version of the HotLeakage simulator[21] was used to simulate the LowLEAC cache architecture. HotLeakage is an architectural simulator based off SimpleScalar. It uses the micro-architectural power simulator Wattch[22] and cache simulator CACTI-P[17] for timing and power analysis.

HotLeakage simulates an architectural model for subthreshold and gate leakage considering parameters like temperature and operating voltage which directly affect the leakage current in transistors. Along with the leakage power dissipated by cache structures, HotLeakage models the extra dynamic and leakage power dissipated by the leakage reduction technique viz. switching modes of SRAM cells etc. HotLeakage calculates the subthreshold and gate leakage currents for each transistor and uses it to model the effective leakage power consumption in caches and other architectural components. It also models the following costs incurred by the leakage reduction techniques.

- Dynamic power due to the extra hardware.
- Leakage power due to the extra hardware.
- Dynamic power due to the mode transitions (active to passive and vice-versa).
- Dynamic power due to extra latency (or state loss) in accessing the structure.

www.manaraa.com

In modeling the effectiveness of the control techniques, the HotLeakage simulator measures the costs against the benefits. This helps in justifying the penalties incurred in LowLEAC implementation with respect to its benefits.

The HotLeakage code is modified to support the implementation of queue to hold the active cache line numbers. L1 data cache is modified to support the active lines buffer of size 256. Since our focus is on L1 data cache, all the other data and instruction caches are kept with traditional cell implementation. It should also be noted that the L1 data cache tags remain powered on at all times. HotLeakage calculates static power dissipation based on the ratio of inactive to active lines. The dynamic power consumption is calculated based on the number of accesses to data in all levels of cache. HotLeakage is further modified to take into account the extra static and dynamic power consumption by the buffer and charge pumps to maintain cNVSRAM cell data.

4.1.1 Simulation parameters

We simulate the Alpha out-of-order superscalar architecture, whose cache configurations are similar to an Intel Core 2 processor. Our simulations include running 100 million instructions of various benchmarks from the spec 2006 benchmark suite. We let 20 million instructions to fast forward at initialization to avoid the compulsory misses.

Table 4.1 shows the input parameters used in the HotLeakage cache modeling simulation.

Table 4.1 Parameters of HotLeakage simulation.

Parameter	**Value**
Operating Temperature	353 K
CMOS Technology	70 nm
L1 Data Cache Size	64 kB
L1 Data Cache Associativity	varying
L1 Data Cache Line Size	32 bits
L1 Data Cache Supply Voltage	0.9V
L1 Data Cache Latency	2 Cycles
L2 Data Cache Latency	11 Cycles
Main-Memory Latency	97 Cycles

www.manaraa.com

4.1.2 Impact of queue sizes on power consumption

Figure 4.1 shows the net power consumption of the L1 data cache. For each benchmark, the size of the active cache lines queue was varied between 64 lines, 128 lines, 256 lines, and the entire cache (2048 lines) on. Although more savings are expected with a smaller number of the active cache lines, the miss penalty incurred is also higher. As a result, the extra dynamic power consumption for fetching data from the next level of cache is also more. Most benchmarks show that queue sizes as small as 64 or 128 actually consume more power than the original cache due to the extra dynamic power consumption.

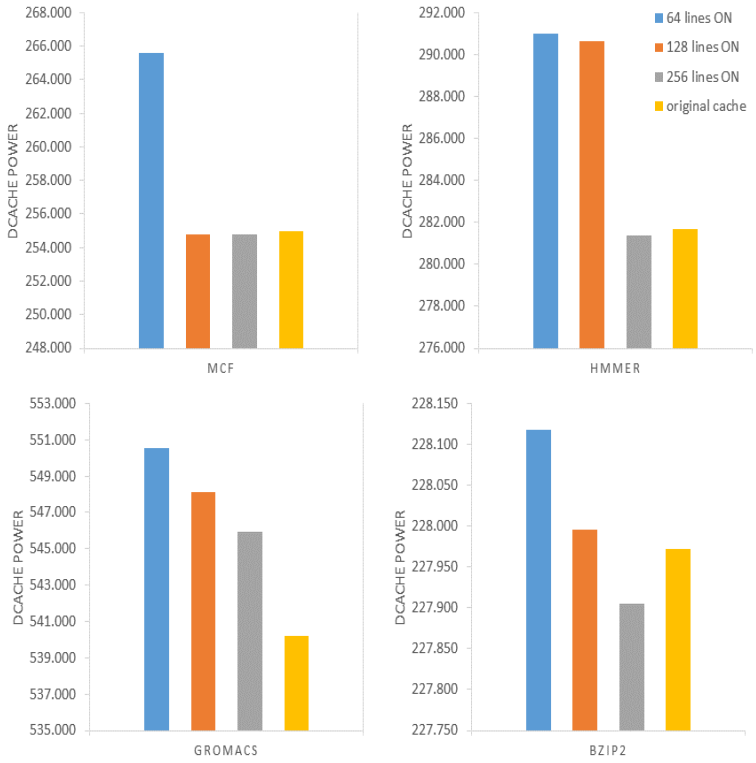

Figure 4.1 L1 data cache power consumption for various number of cache lines active.

Striking a balance between the number of active lines and a reasonable or permissible miss rate is the key to achieve maximum savings since extra dynamic power consumption highly depends on the miss rate increase. Figure 3.3 in Chapter 3 shows the miss rates for the above active lines queue sizes. Though a smaller queue means more leakage energy savings, it also results in a higher miss rate and more extra dynamic energy to fetch data from next cache level.

www.manaraa.com

Thus it is important to select an optimal number of active cache lines that yield minimum miss rate overhead in order to achieve maximum leakage energy savings.

4.1.3 Power savings without data back-up

Figure 4.1 shows that a list size like 256 lines is ideal for most applications and it shows marginal power savings over the original cache without any extra leakage reduction technique implemented. Figure 3.5 in Chapter 3, shows that savings can be highly affected due to this excess dynamic power dissipation, even with an optimal buffer size. Hence to further improve these savings, the best option is to avoid the extra dynamic power dissipation of fetching data from the next cache level. Thus, non-volatile back up plays a crucial role in improving overall power savings after choosing the optimal buffer size.

4.1.4 Modeling the cNVSRAM cell in HotLeakage simulator

For the cNVSRAM cells, we consider cell parameters based on 65 nm SMIC technology as described in [2]. Table 4.2 shows the cell parameter values. Each cNVSRAM cell has an area of $4.3\mu m \times 2.7\mu m$. A total of 64 charge pumps are needed to maintain data back up in 64kB L1 cache. Each charge pump requires $32\mu m \times 35\mu m$ area and it can keep 1kB of data backed up in low power mode. Table 4.2 shows a summary of the area, power consumption and timing penalty of a cNVSRAM cell. The HotLeakage power and timing simulator is modified to incorporate these parameters of the cNVSRAM cell.

Table 4.2 Parameters of cNVSRAM cell.

Element	**Area**	**Power**	**Timing**
cNVSRAM cell	$4.3\mu m \times 2.7\mu m$	1.57μW	$1.4E^{-11}$s(write time)
Charge pump	$32\mu m \times 35\mu m$	56μW	353 ns

The leakage saving of the cNVSRAM cell highly depends on the time required to recover data from the non-volatile back up. Thus it is important to keep this time minimum. In order to do so the transmission gate transistor Nt, which controls the data restoration, is given the minimum size of 120 nm/60 nm supported by 65 nm SMIC technology. The gate oxide thickness is 2.35 nm. The sizes of the tunneling and coupling capacitors is chosen such that it

www.manaraa.com

has a minimum impact on the restoration time which is limited to 11 ps. The ratio of C2/C1 is 14 and their sizes are W_{c1} = 120 nm, L_{c1} = 60 nm, W_{c2} = 420 nm, and L_{c2} = 240 nm [2]. These sizes are incorporated in the modified HotLeakage simulator.

It should be noted that since the basic HotLeakage simulator is based on 6 transistor SRAM cells, the simulation equations are scaled to achieve the equivalent of an 8 transistor SRAM cell and the cNVSRAM cell. We compare the performance of LowLEAC made of cNVSRAM cells with that of a regular cache made of 6T SRAM cells. This comparison helps us to prove the effectiveness of the technique in itself and gives an idea of the possible areas of improvement. We primarily look at the energy efficiency of LowLEAC as a whole compared to a cache which does not implement any leakage reduction technique. A detailed analysis of other physical factors of comparison between 6T and cNVSRAM cells shall not fall under the scope of this work.

4.1.5 Area overhead of cNVSRAM cell

The cNVSRAM cell implements a complex cell structure with more transistors than a conventional 6T SRAM cell in order to back up data. It also implements charge pumps to maintain this data. All this ends up in adding to the chip area resulting in an area cost. Using the values from table 4.2 in McPAT and CACTI-P simulators, the area and timing costs involved in the implementation of LowLEAC are calculated. For a data cache of 64kB, the area overhead in the cache size is approximately 25% for 65nm technology. This is a reasonably small value when considering the leakage savings obtained.

4.2 LowLEAC's Dcache power consumption

The effectiveness of LowLEAC with data back up depends highly on the size of the active lines' queue. The cNVSRAM cells have a transitional energy overhead which consists of energy to drive charge pumps that back up data. It may seem that keeping the entire cache turned off and backed up will save maximum power, but in reality the transitional energy overhead involved in it is quite high. This results in bringing down the effective savings achieved. To

www.manaraa.com

avoid this, we choose only a part of the cache to be turned off, preferably, the one not used recently.

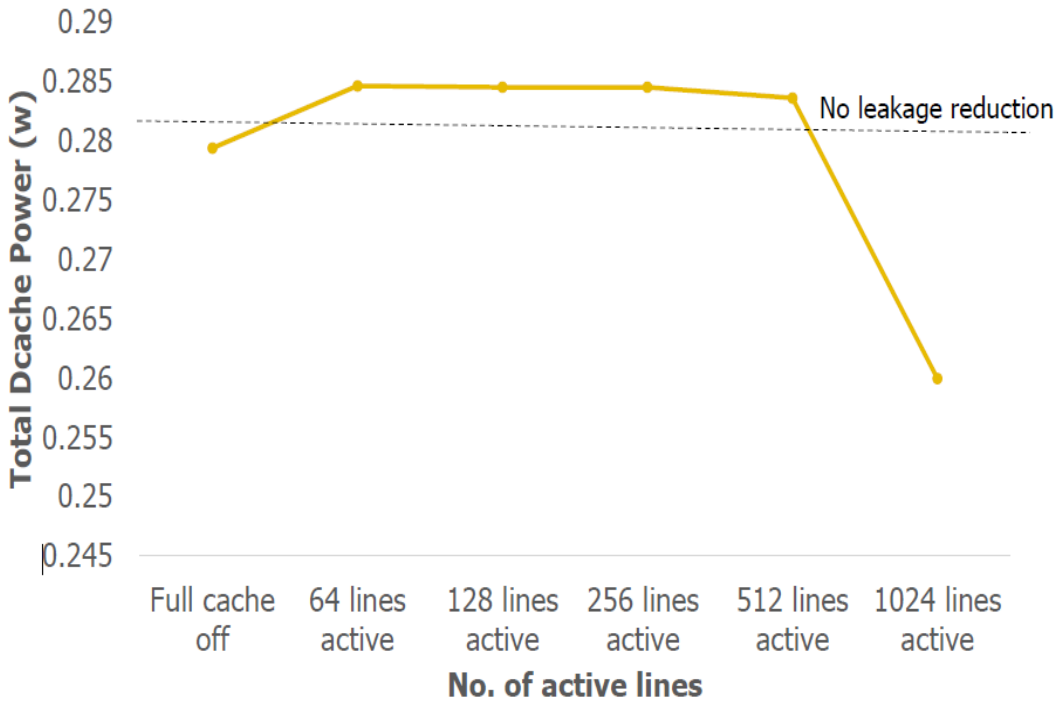

Figure 4.2 LowLEAC's L1 data cache power consumption for various queue sizes.

Figure 4.2 shows the total L1 data cache power consumption of LowLEAC for various queue sizes ranging from zero (entire cache off) to 1024 lines (half cache off). It can be seen that the net power consumption is much lower for a queue size of 1024 lines than keeping the entire cache off. For intermediate queue sizes, the total power consumption is actually more than the base power consumption due to transitional energy overhead exceeding the savings. Thus it is very important that for the maximum effectiveness of LowLEAC we choose an optimal queue size that achieves a balance between the leakage energy savings and the transitional energy overheads. This optimal queue size varies depending on the application.

www.manaraa.com

4.3 Energy savings of LowLEAC data cache

Based on Figure 4.2, an optimal queue size of 1024 lines is selected for analysis since it gives a reasonable trade-off between the power saved and overhead costs for most benchmarks. To further enhance the energy savings, we introduce data back up in cNVSRAM cells. The non volatile back up helps save the extra dynamic power used in fetching the lost data from the next level of cache. In an operational state, cNVSRAM cells work as conventional SRAM and they consume similar dynamic energy.

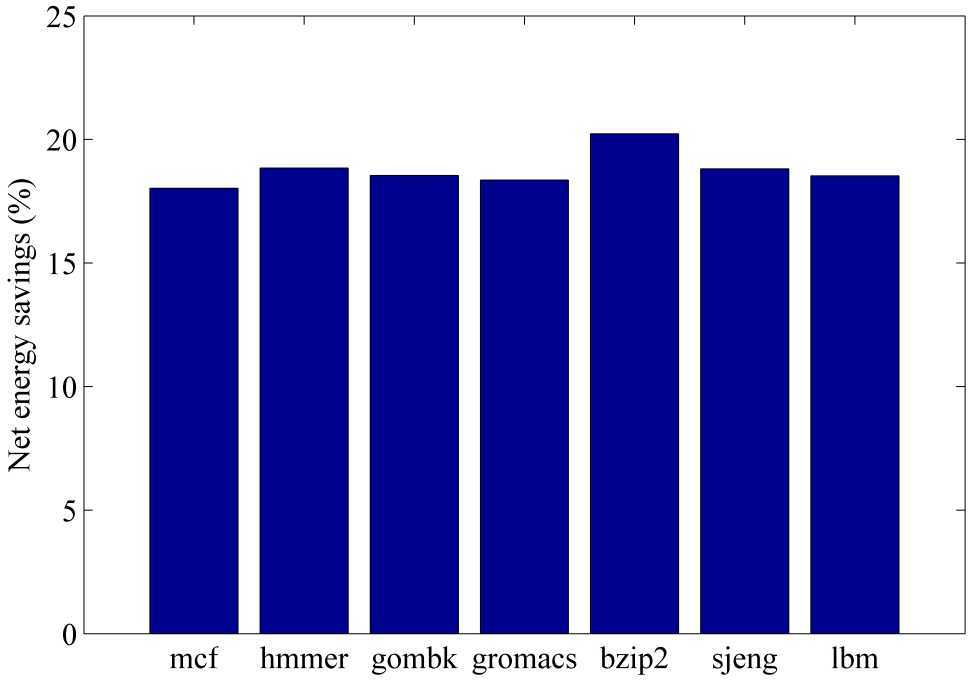

Figure 4.3 LowLEAC's net L1 data cache energy savings.

Further modifying HotLeakage, the 6T SRAM cell parameters are scaled to match 8T SRAM cell as the performance of cNVSRAM cells in active mode is similar to that of 8T SRAM [2]. Thus, the dynamic energy of the cache in active mode is similar to that of an 8T SRAM cell.

Figure 4.3 shows the net energy savings achieved by LowLEAC technique implemented using cNVSRAM cells. LowLEAC achieves around 18% leakage energy savings as compared

www.manaraa.com

to a conventional L1 data cache made of 6T SRAM cell without any leakage energy reduction technique. These savings are calculated after considering the power consumption by the cNVSRAM cells, the charge pump and the extra logic needed to maintain the list of active cache line numbers. LowLEAC with the cNVSRAM cell gives better leakage energy savings despite having more number of transistors than a conventional 6T SRAM cell. For a minor area overhead, LowLEAC achieves significant leakage energy reduction with almost no performance penalty.

www.manaraa.com

CHAPTER 5. SUMMARY

This chapter summarizes the work described in this thesis and mentions the future research scope and possible enhancements to this topic.

5.1 Design summary

Low Leakage Energy Architecture for Caches (LowLEAC) is an architecture-level technique proposed to minimize the leakage power dissipation in CMOS SRAM cache. It is based on keeping inactive lines of the L1 data cache that are not used recently and do not contain data that might be needed in future, powered off. The cache memory is implemented using CMOS compatible non-volatile SRAM cell called cNVSRAM. This cell works in two modes, active mode facilitates full use of the cell while in sleep mode, it is turned off to avoid leakage power dissipation. The data in an inactive line is stored in a non-volatile back-up before turning it off, to avoid the need to fetch it from next level of cache. Thus it helps in reducing the power and performance penalty occurred due to loss of data from the turned off cache lines.

5.2 Results summary

With a small area penalty of 25%, LowLEAC achieves significant leakage power reduction without compromising the performance. It shows around 18% energy savings in L1 data cache as compared to conventional 6T SRAM cache. Thus, LowLEAC proves to be an ideal architecture for mobile computing devices with limited battery capacity, as well as, helps large server stations to limit their cooling costs incurred. LowLEAC is beneficial for higher level caches too, leveraging more energy savings from their longer sleep times.

www.manaraa.com

5.3 Future directions

As a part of future study, we plan to randomly switch off lines instead of maintaining the active lines queue. When a line is turned on, another randomly selected line will be switched off. This will help save the energy expenditure involved in maintaining the queue, further enhancing the net energy savings. The aim is to study its impact on the penalties as compared to the additional energy savings achieved by avoiding the queue.

LowLEAC can also prove to be a good fit for higher level caches since the sleep time is more and hence savings are more. Although the area costs increase since higher level cache sizes are bigger than the L1 cache thus affecting LowLEAC's overall efficiency. Future study and modifications to the design like having a runtime adaptable queue size can help minimize costs and maximize savings. Such a scalable active lines queue sound particularly suitable for larger memories where the size of queue can be a considerable factor in deciding the power savings. Furthermore, scalable queues could be implemented using FPGAs or reconfigurable hardware to achieve speed and area savings. LowLEAC design can also be enhanced/updated for instruction caches or more recently proposed off-die DRAM caches. Design enhancements would also be required to use LowLEAC for higher level partitioned caches in multi-core systems.

www.manaraa.com

Bibliography

[1] John L. Hennessy and David A. Patterson. *Computer Architecture- A Quantitative Approach*. MK publications, 2012.

[2] J. Wang, L. Wang, H. Yin, Z. Wei, Z. Yang, and N. Gong. cnv sram: Cmos technology compatible non-volatile sram based ultra-low leakage energy hybrid memory system. *IEEE Transactions on Computers*, PP(99):1–1, 2014. ISSN 0018-9340. doi: 10.1109/TC.2014.2375187.

[3] S. Kaxiras, Zhigang Hu, and M. Martonosi. Cache decay: exploiting generational behavior to reduce cache leakage power. In *Computer Architecture, 2001. Proceedings. 28th Annual International Symposium on*, pages 240–251, 2001. doi: 10.1109/ISCA.2001.937453.

[4] Semiconductor Industry Association. International technology roadmap for semiconductors, 2013. URL http://public.itrs.net.

[5] K. Flautner, Nam Sung Kim, S. Martin, D. Blaauw, and T. Mudge. Drowsy caches: simple techniques for reducing leakage power. In *Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on*, pages 148–157, 2002. doi: 10.1109/ISCA.2002.1003572.

[6] Douglas C Burger, James R Goodman, and Alain Kagi. *The declining effectiveness of dynamic caching for general-purpose microprocessors*. University of Wisconsin-Madison, Computer Sciences Department, 1995.

[7] David A Wood, Mark D Hill, and Richard E Kessler. *A model for estimating trace-sample miss ratios*, volume 19. ACM, 1991.

www.manaraa.com

[8] Wm A Wulf and Sally A McKee. Hitting the memory wall: implications of the obvious. *ACM SIGARCH computer architecture news*, 23(1):20–24, 1995.

[9] Gordon E Moore et al. Cramming more components onto integrated circuits. *Proceedings of the IEEE*, 86(1):82–85, 1998.

[10] Robert Chau, Justin Brask, Suman Datta, Gilbert Dewey, Mark Doczy, Brian Doyle, Jack Kavalieros, Ben Jin, Matthew Metz, Amlan Majumdar, et al. Application of high-κ gate dielectrics and metal gate electrodes to enable silicon and non-silicon logic nanotechnology. *Microelectronic Engineering*, 80:1–6, 2005.

[11] Huiyang Zhou, Mark C. Toburen, Eric Rotenberg, and Thomas M. Conte. Adaptive mode control: A static-power-efficient cache design. *ACM Trans. Embed. Comput. Syst.*, 2(3):347–372, August 2003. ISSN 1539-9087. doi: 10.1145/860176.860181. URL http://doi.acm.org/10.1145/860176.860181.

[12] S.-H. Yang, M.D. Powell, B. Falsafi, K. Roy, and T.N. Vijaykumar. An integrated circuit/architecture approach to reducing leakage in deep-submicron high-performance i-caches. In *High-Performance Computer Architecture, 2001. HPCA. The Seventh International Symposium on*, pages 147–157, 2001. doi: 10.1109/HPCA.2001.903259.

[13] Michael Powell, Se-Hyun Yang, Babak Falsafi, Kaushik Roy, and T. N. Vijaykumar. Gated-vdd: A circuit technique to reduce leakage in deep-submicron cache memories. In *Proceedings of the 2000 International Symposium on Low Power Electronics and Design*, ISLPED '00, pages 90–95, New York, NY, USA, 2000. ACM. ISBN 1-58113-190-9. doi: 10.1145/344166.344526. URL http://doi.acm.org/10.1145/344166.344526.

[14] H. Hanson, M.S. Hrishikesh, V. Agarwal, S.W. Keckler, and D. Burger. Static energy reduction techniques for microprocessor caches. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 11(3):303–313, June 2003. ISSN 1063-8210. doi: 10.1109/TVLSI.2003.812370.

www.manaraa.com

[15] Shin'ichiro Mutoh, Takakuni Douseki, Yasuyuki Matsuya, Takahiro Aoki, Satoshi Shigematsu, and Junzo Yamada. 1-v power supply high-speed digital circuit technology with multithreshold-voltage cmos. *Solid-State Circuits, IEEE Journal of*, 30(8):847–854, 1995.

[16] Koji Nii, Hiroshi Makino, Yoshiki Tujihashi, Chikayoshi Morishima, Yasushi Hayakawa, Hiroyuki Nunogami, Takahiko Arakawa, and Hisanori Hamano. A low power sram using auto-backgate-controlled mt-cmos. In *Low Power Electronics and Design, 1998. Proceedings. 1998 International Symposium on*, pages 293–298. IEEE, 1998.

[17] Sheng Li, Ke Chen, Jung Ho Ahn, Jay B. Brockman, and Norman P. Jouppi. Cacti-p: Architecture-level modeling for sram-based structures with advanced leakage reduction techniques. In *ICCAD: International Conference on Computer-Aided Design*, pages 694–701, 2011.

[18] V. George, S. Jahagirdar, C. Tong, K. Smits, S. Damaraju, S. Siers, V. Naydenov, T. Khondker, S. Sarkar, and P. Singh. Penryn: 45-nm next generation intel x00ae; core x2122; 2 processor. In *Solid-State Circuits Conference, 2007. ASSCC '07. IEEE Asian*, pages 14–17, Nov 2007. doi: 10.1109/ASSCC.2007.4425784.

[19] Y. Ye, S. Borkar, and V. De. A new technique for standby leakage reduction in high-performance circuits. In *VLSI Circuits, 1998. Digest of Technical Papers. 1998 Symposium on*, pages 40–41, June 1998. doi: 10.1109/VLSIC.1998.687996.

[20] Yingmin Li, D. Parikh, Yan Zhang, K. Sankaranarayanan, M. Stan, and K. Skadron. State-preserving vs. non-state-preserving leakage control in caches. In *Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings*, volume 1, pages 22–27 Vol.1, Feb 2004. doi: 10.1109/DATE.2004.1268822.

[21] Yan Zhang, Dharmesh Parikh, Karthik Sankaranarayanan, Kevin Skadron, and Mircea Stan. Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects. *University of Virginia Dept of Computer Science Tech Report CS-2003*, 5, 2003.

[22] David Brooks, Vivek Tiwari, and Margaret Martonosi. *Wattch: a framework for architectural-level power analysis and optimizations*, volume 28. ACM, 2000.

www.manaraa.com