Power reduction techniques for memory elements

Srikanth Katrue

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
Power Reduction Techniques for Memory Elements

by

Srikanth Katrue

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Engineering

Supervised by

Dr. Dhireesha Kudithipudi
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, NY
December 14, 2007

Approved By:

_____________________________________________        ___________      ___
Dr. Dhireesha Kudithipudi
Primary Advisor – R.I.T. Dept. of Computer Engineering

_____________________________________________
Dr. Pratapa V. Reddy
Secondary Advisor – R.I.T. Dept. of Computer Engineering

_____________________________________________
Dr. Kenneth W. Hsu
Secondary Advisor – R.I.T. Dept. of Computer Engineering
Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title: Power Reduction Techniques for Memory Elements

I, Srikanth Katrue, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

_________________________________
Srikanth Katrue

_________________________________
Date
Acknowledgements

I am very thankful to my advisor Dr. Dhireesha Kuditipudi for her guidance throughout my research. Throughout my thesis writing period, she provided me encouragement & lot of good ideas to keep my motivation focused towards completion of my thesis. I also want to thank Dr. Kenneth W. Hsu and Dr. Pratapa V. Reddy for being a part of my committee and encouraging me to complete my thesis.

I am very grateful to my parents, without whose moral support this work would have never been completed. I am also thankful to the department head, Dr. Andreas Savakis for motivating me to complete my thesis. I thank the department of Computer Engineering for providing all the computing facilities and access to needed software. I cannot forget the help and support of all my friends and colleagues who have helped in every step of the way. Finally I would like to thank everyone who has directly or indirectly helped me to complete my research.
Abstract

High performance and computational capability in the current generation processors are made possible by small feature sizes and high device density. To maintain the current drive strength and control the dynamic power in these processors, simultaneous scaling down of supply and threshold voltages is performed. High device density and low threshold voltages result in an increase in the leakage current dissipation. Large on chip caches are integrated onto the current generation processors which are becoming a major contributor to total leakage power.

In this work, a novel methodology is proposed to minimize the leakage power and dynamic power. The proposed static power reduction technique, GALEOR (GAted LEakage transistOR), introduces stacks by placing high threshold voltage transistors and consists of inherent control logic. The proposed dynamic power reduction technique, adaptive phase tag cache, achieves power savings through varying tag size for a design window. Testing and verification of the proposed techniques is performed on a two level cache system.

Power delay squared product is used as a metric to measure the effectiveness of the proposed techniques. The GALEOR technique achieves 30% reduction when implemented on CMOS benchmark circuits and an overall leakage savings of 9% when implemented on the two level cache systems. The proposed dynamic power reduction technique achieves 10% savings when implemented on individual modules of the two level cache and an overall savings of 3% when implemented on the entire two level cache system.
# Table of Contents

Acknowledgements

Abstract

Table of Contents

List of Figures

List of Tables

Glossary

1 Introduction

1.1 Impact of Technology Scaling on Power Dissipation

1.2 Power Dissipation in Microprocessor

1.3 Thesis Contributions

1.4 Thesis Organization

2 Power Dissipation Sources

2.1 Sources of Dynamic Power Dissipation

2.1.1 Switching Power

2.1.2 Short Circuit Power

2.1.3 Glitching Power

2.2 Sources of Static Power Dissipation

2.2.1 Gate Oxide Tunneling Leakage

2.2.2 Sub-Threshold Leakage

2.2.3 Reverse-Bias Junction Leakage

2.2.4 Gate Induced Drain Leakage

2.2.5 Punch Through Leakage

2.2.6 Hot Carrier Injection

2.3 Conclusion

3 Background Work

3.1 Static Power Reduction Techniques

3.1.1 Cache Decay

3.1.2 Selectively Activated Cache

3.1.3 Gated Supply Technique

3.1.4 Leakage Feedback Technique

3.1.5 Stack Technique

3.1.6 Sleepy Keeper

3.2 Dynamic Power Reduction Techniques

3.2.1 Gate Freezing

3.2.2 Transistor Sizing

3.2.3 Guarded Evaluation

3.2.4 Precomputation Logic

3.2.5 Dynamically Resizable Cache

3.3 Conclusion

4 System Level Design

v
4.1 Two Level Cache System ................................................................. 20
4.2 CPU Buffer ................................................................................. 23
4.2.1 Design Specifications ................................................................. 23
4.3 Write Buffer .................................................................................. 23
4.3.1 Design Specifications ................................................................. 24
4.4 Translation Look-Aside Buffer ....................................................... 24
4.4.1 Instruction TLB Specifications ................................................... 25
4.4.2 Data TLB Specifications .............................................................. 26
4.5 Cache ............................................................................................. 27
4.5.1 Level 1 Instruction Cache Specifications ...................................... 27
4.5.2 Level 1 Data Cache Specifications .............................................. 28
4.5.3 Level 2 Unified Cache Specifications .......................................... 29
4.6 Main Memory ................................................................................. 30
4.7 Conclusion ..................................................................................... 32

5 Proposed Power Reduction Techniques ............................................. 33
5.1 Adaptive Phase Tag Technique ...................................................... 33
5.2 GALEOR Technique ..................................................................... 35
5.3 Leakage Power Analysis in Memory Elements ............................ 39
5.3.1 ‘D’ Flip – Flop ........................................................................... 39
5.4 Conclusion ..................................................................................... 40

6 Results .............................................................................................. 42
6.1 Leakage Power Savings in Benchmark Circuits ......................... 42
6.1.1 Inverter ....................................................................................... 42
6.1.2 Buffer ......................................................................................... 43
6.1.3 Two Input NAND ...................................................................... 43
6.1.4 Two Input NOR ......................................................................... 44
6.1.5 Two Input AND ........................................................................ 45
6.1.6 Two Input OR ........................................................................... 45
6.1.7 Three Input NAND ................................................................. 46
6.1.8 Three Input NOR ................................................................. 47
6.1.9 Three Input AND ................................................................. 48
6.1.10 Three Input OR ................................................................. 49
6.1.11 Two Input XOR ...................................................................... 50
6.1.12 Two Input XNOR ................................................................. 51
6.1.13 ‘D’ Flip – Flop ........................................................................... 51
6.1.14 ‘D’ Flip – Flop with Reset Enable ........................................... 52
6.1.15 Static Leakage Power Delay Squared Product ...................... 53
6.2 Leakage Power Savings in the Two Level Cache ....................... 54
6.3 Dynamic Power Savings in the Memory System ....................... 55
6.4 Conclusion ..................................................................................... 56

7 Conclusion & Future Work ............................................................... 57
7.1 Future Work ................................................................................... 57
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Power Projections by ITRS</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Power Dissipation in Microprocessor</td>
<td>2</td>
</tr>
<tr>
<td>2.1</td>
<td>Components of load capacitance</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Short Circuit Current in Inverter</td>
<td>5</td>
</tr>
<tr>
<td>2.3</td>
<td>Leakage Current Mechanisms</td>
<td>6</td>
</tr>
<tr>
<td>2.4</td>
<td>Components of Tunneling Current</td>
<td>7</td>
</tr>
<tr>
<td>3.1</td>
<td>Dual Threshold Voltage Architecture</td>
<td>12</td>
</tr>
<tr>
<td>3.2</td>
<td>Gated Supply Technique</td>
<td>13</td>
</tr>
<tr>
<td>3.3</td>
<td>Leakage Feedback Technique</td>
<td>14</td>
</tr>
<tr>
<td>3.4</td>
<td>Stack Technique</td>
<td>14</td>
</tr>
<tr>
<td>3.5</td>
<td>Sleepy Keeper</td>
<td>15</td>
</tr>
<tr>
<td>3.6</td>
<td>Guarded Evaluation</td>
<td>17</td>
</tr>
<tr>
<td>3.7</td>
<td>Precomputation Logic</td>
<td>17</td>
</tr>
<tr>
<td>4.1</td>
<td>Two Level Cache System</td>
<td>19</td>
</tr>
<tr>
<td>4.2</td>
<td>CPU Buffer</td>
<td>23</td>
</tr>
<tr>
<td>4.3</td>
<td>Write Buffer</td>
<td>24</td>
</tr>
<tr>
<td>4.4</td>
<td>Instruction Translation Look-Aside Buffer</td>
<td>25</td>
</tr>
<tr>
<td>4.5</td>
<td>Data Translation Look-Aside Buffer</td>
<td>26</td>
</tr>
<tr>
<td>4.6</td>
<td>Level 1 Instruction Cache</td>
<td>28</td>
</tr>
<tr>
<td>4.7</td>
<td>Level 1 Data Cache</td>
<td>29</td>
</tr>
<tr>
<td>4.8</td>
<td>Level 2 Unified Cache</td>
<td>30</td>
</tr>
<tr>
<td>4.9</td>
<td>Main Memory</td>
<td>31</td>
</tr>
</tbody>
</table>
Figure 5.1 - Two Way Set Associative Adaptive Phase Tag Cache

Figure 5.2 - Two Input NAND Gate

Figure 5.3 - GALEOR Technique Implemented on a Two Input NAND Gate

Figure 5.4 - Gated Leakage ‘D’ Flip – Flop

Figure 6.1 - Leakage Power Dissipation in an Inverter

Figure 6.2 - Leakage Power Dissipation in a Buffer

Figure 6.3 - Leakage Power Dissipation in a Two Input NAND Gate

Figure 6.4 - Leakage Power Dissipation in a Two Input NOR Gate

Figure 6.5 - Leakage Power Dissipation in a Two Input AND Gate

Figure 6.6 - Leakage Power Dissipation in a Two Input OR Gate

Figure 6.7 - Leakage Power Dissipation in a Three Input NAND Gate

Figure 6.8 - Leakage Power Dissipation in a Three Input NOR Gate

Figure 6.9 - Leakage Power Dissipation in a Three Input AND Gate

Figure 6.10 - Leakage Power Dissipation in a Three Input OR Gate

Figure 6.11 - Leakage Power Dissipation in a Two Input XOR Gate

Figure 6.12 - Leakage Power Dissipation in a Two Input XNOR Gate

Figure 6.13 - Leakage Power Dissipation in a ‘D’ Flip – Flop

Figure 6.14 - Leakage Power Dissipation in a ‘D’ Flip – Flop with Reset Enable

Figure 6.15 - Static Leakage Power Savings in the Benchmark Circuits

Figure 6.16 - Leakage Power Delay Squared Product Savings in Buffer’s & TLB’s

Figure 6.17 - Leakage Power Delay Squared Product Savings in Level 1, Level 2 Caches

Figure 6.18 - % Leakage Power Delay Squared Product Savings in Memory System

Figure 6.19 - Dynamic Power Delay Squared Product Savings in Memory System
Figure 12.8 - Layout of Main Memory
List of Tables

Table 5.1 - Allocation of Tag bits based on the Hit Ratio ................................................ 34
Table 5.2 - Leakage Current in Two Input NAND gate ...................................................... 36
Table 5.3 - Leakage Current in Two Input Gated Leakage Transistor NAND gate .......... 39
Table 5.4 – Stacks added in a Gated Leakage D Flip - Flop .............................................. 40
Table 9.1 – Functional Pin Description of CPU Buffer ...................................................... 60
Table 9.2 – Functional Pin Description of Write Buffer ..................................................... 60
Table 9.3 – Functional Pin Description of Instruction TLB ................................................. 60
Table 9.4 – Functional Pin Description of Data TLB ........................................................ 61
Table 9.5 – Functional Pin Description of Level 1 Instruction Cache ............................... 62
Table 9.6 – Functional Pin Description of Level 1 Data Cache ........................................ 62
Table 9.7 – Functional Pin Description of Level 2 Unified Cache ..................................... 63
Table 9.8 – Functional Pin Description of Main Memory .................................................. 64
## Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMOS</td>
<td>N – Channel Metal Oxide Semi Conductor</td>
</tr>
<tr>
<td>PMOS</td>
<td>P – Channel Metal Oxide Semi Conductor</td>
</tr>
<tr>
<td>CMOS</td>
<td>Complementary Metal Oxide Semiconductor.</td>
</tr>
<tr>
<td>ITRS</td>
<td>International Technology Roadmap for Semiconductors</td>
</tr>
<tr>
<td>ITLB</td>
<td>Instruction Translation Look-Aside Buffer</td>
</tr>
<tr>
<td>DTLB</td>
<td>Data Translation Look-Aside Buffer</td>
</tr>
<tr>
<td>ICACHE</td>
<td>Instruction Cache</td>
</tr>
<tr>
<td>DCACHE</td>
<td>Data Cache</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Page Number</td>
</tr>
<tr>
<td>GALEOR</td>
<td>Gated Leakage Transistor</td>
</tr>
<tr>
<td>LECTOR</td>
<td>Leakage Control Transistor</td>
</tr>
</tbody>
</table>
Chapter 1 Introduction

Increase in the transistor speed and number of transistors result in high performance in the current generation processors. The performance improvements have been accompanied by an increase in the power dissipation. High power dissipation systems increase cost of cooling and reduce the system reliability.

1.1 Impact of Technology Scaling on Power Dissipation

CMOS technology has been scaled down from 1µm to 45nm over the last decade. Technology scaling reduces the gate oxide thickness and the gate length thereby increasing the transistor density and also reduces the delay. Reduced gate lengths result in an increase in the leakage power dissipation. Increased transistor densities result in an increase in the power dissipation per unit area thereby creating hotspots. In the Figure 1.1, the static power dissipation is expected to equal the dynamic power dissipation by the year 2007.

![FIGURE 1.1: 2002 Power Projections by ITRS [1]](image-url)
Scaling down the supply voltage reduces the switching power dissipation. Threshold voltage is simultaneously scaled down along with the supply voltage to improve transistor switching speed. Scaling down the threshold voltage significantly increases the leakage power dissipation.

### 1.2 Power Dissipation in Microprocessor

Caches consume almost 50% of the total leakage power dissipated. Leakage power dissipation in caches continue to increase due to the increase in the transistor density.

![POWER DISSIPATION IN MICROPROCESSOR](image)

**FIGURE 1.2: Power Dissipation in Microprocessor**

In this thesis we developed a working model of a two level cache system which is used as a test bed for running the dynamic and static power simulations. We proposed dynamic and static power reduction techniques with minimal area and performance overhead.

### 1.3 Thesis Contributions

In this thesis we developed a working model of a two level cache system which is used as a test bed for running the dynamic and static power simulations. We proposed dynamic and static power reduction techniques with minimal area and performance overhead.

### 1.4 Thesis Organization

Thesis is organized as follows. Chapter 2 deals with the different sources of power dissipation. Chapter 3 outlines the existing power reduction techniques for minimizing dynamic and leakage
power. Chapter 4 discusses the two level cache system and the functionality of each of the sub modules. In chapter 5 proposed power reduction techniques were discussed. Chapter 6 presents the results obtained by implementing the power reduction techniques on standard gates and two level cache system. Finally chapter 7 presents conclusions and suggestions for future research work.
Chapter 2 Power Dissipation Sources

Speed and area are the main objectives in designing high end systems and portable electronic devices. However design of low power systems has become a prime concern in the current generation processors. Two principal sources of power dissipation in modern processors are dynamic power and static leakage power.

2.1 Sources of Dynamic Power Dissipation

2.1.1 Switching Power

Switching power dissipation in a circuit results from charging/discharging the output load capacitance to supply/ground voltage. The power dissipation depends on supply voltage, output load capacitance and frequency of output transitions [2]. The output load capacitance is the sum of gate to drain capacitance, diffusion capacitance and wiring capacitance [2]. Switching power is expressed in equation 2.1 as follows

\[ P_{dyn} = \alpha_{0 \rightarrow 1} \cdot C_L \cdot V_{dd} \cdot V_{swing} \cdot f \]  

(2.1)

where \( C_L \) is the sum of gate, junction and interconnect capacitances as shown in Figure 2.1, \( V_{dd} \) is the supply voltage, \( V_{swing} \) is the output logic swing, \( f \) is the operating frequency and \( \alpha \) is the activity factor. We can clearly observe that the switching power has linear dependence on load capacitance, frequency of input transitions and a quadratic dependence on supply voltage.

![FIGURE 2.1: Components of load capacitance [2]](image-url)
2.1.2 Short Circuit Power

Short circuit power dissipation results due to the current flowing from supply to ground during input signal transitions. Consider an inverter circuit shown in the Figure 2.2, when the input voltage is between \( v_{tn} \) and \( v_{dd} - |v_{tp}| \) (\( v_{tn} \) is the NMOS threshold voltage and \( v_{tp} \) is the PMOS threshold voltage) and output voltage is between ground and supply voltages, a conductive path exists from supply to ground allowing the short circuit current to flow [3]. The equation for short circuit power is given by the equation 2.2

\[
P_{sc} = \frac{1}{12} k \tau_{in} (v_{dd}^3 - 2v_t v_{dd})^3 f \alpha
\]  

(2.2)

where \( \tau_{in} \) is the input transition time, \( v_t \) is the threshold voltage of the transistors, \( k \) is the effective trans-conductance parameter of the logic gate, \( v_{dd} \) is the supply voltage, \( f \) is the frequency of operation and \( \alpha \) is the switching factor. To minimize short circuit current it is desirable to have equal input and output rise/fall times since input rise/fall time of one gate is the output rise/fall time of other gate. To eliminate short circuit power completely the supply voltage is made smaller than the sum of threshold voltages of NMOS and PMOS transistors [2]. The Figure 2.2 below shows flow of short circuit current in an inverter and peak of short circuit current occurs when both NMOS and PMOS devices are turned ON.

![FIGURE 2.2: Short Circuit Current in Inverter [2]](image)
2.1.3 Glitching Power
Glitching power dissipation occurs due to mismatch in input signal path lengths in a network [5]. Glitch power is minimized when the width of the glitch is made smaller than the delay of the gate and also when all the inputs to a gate arrive simultaneously [4].

2.2 Sources of Static Power Dissipation
Static leakage power dissipation occurs due to the current flowing through the transistors in the idle state. Leakage power dissipation depends on threshold voltage, gate size and oxide thickness.

![FIGURE 2.3: Leakage Current Mechanisms [6]](image)

In the Figure 2.3 above $I_1$ is the reverse bias junction leakage, $I_2$ is subthreshold leakage, $I_3$ is gate oxide tunneling leakage, $I_4$ is current due to hot carrier injection, $I_5$ is gate induced drain leakage and $I_6$ is punch through current. Each of the currents is discussed below in detail.

2.2.1 Gate Oxide Tunneling Leakage
Scaling down the gate oxide thickness increases electric field across the gate. The high electric field causes electrons to tunnel from gate to substrate and substrate to gate resulting in gate oxide leakage current [7, 8]. Two possible mechanisms causing this tunneling phenomenon are Fowler-Nordheim tunneling and direct tunneling [6]. Fowler-Nordheim tunneling results from the electron tunneling into the conduction band of the oxide layer through a triangular potential...
barrier [6]. Thickness of the barrier, barrier height and structure of the barrier determine the tunneling probability of an electron. The current density in the FN tunneling is given by the equation 2.3

\[
J_{FN} = \frac{q^3 E_{ox}^2}{16\pi^2 h^* \varphi_{ox}} \times \exp\left(-\frac{4\sqrt{2m^* \varphi_{ox}^3}}{3h^* q E_{ox}}\right)
\]  

(2.3)

where \( E_{ox} \) is the field across the oxide \( \varphi_{ox} \) is the barrier height for electrons in the conduction band, \( m^* \) is the effective mass of electron in the conduction band of silicon, \( h \) is Planck’s constant.

In direct tunneling electrons from the silicon surface tunnel to the gate through the forbidden energy gap in silicon dioxide [6]. The current density in the direct tunneling mechanism is given by the equation

\[
J_{DT} = A \times E_{ox}^2 \times \exp\left(-\frac{B \left[1 - \left(\frac{\varphi_{ox}}{\varphi_{ox}}\right)^{3/2}\right]}{E_{ox}}\right)
\]  

(2.4)

Where

\[
A = \frac{q^3}{16\pi^2 h^* \varphi_{ox}} \quad B = \frac{4\sqrt{2m^* \varphi_{ox}^3}}{3h^* q},
\]

\( q \) is the electron charge.

Gate oxide tunneling current shown in the Figure 2.4 below comprises of leakage current through the gate to the source and drain overlap regions (\( I_{gso} \) and \( I_{gdo} \)), gate to channel leakage current (\( I_{gc} \)) and gate to substrate leakage current (\( I_{gb} \)) [9].
2.2.2 Sub-threshold Leakage

Sub-threshold leakage current flows between the source and drain of an MOS transistor when the gate voltage is less than the threshold voltage [4]. Sub-threshold conduction is given by the equation 2.5

\[ I_{ds} = \mu_o \cdot C_{ox} \cdot \frac{W}{L} \cdot (m - 1) \cdot (v_t)^2 \cdot e^{\frac{(v_{gs} - v_{th})}{m \cdot v_t}} \cdot \left(1 - e^{\frac{-v_{ds}}{v_t}}\right) \]  

(2.5)

where

\[ m = 1 + \frac{c_{dm}}{c_{ox}} \]

is the sub-threshold swing co-efficient, \( c_{dm} \) is capacitance of the depletion layer, \( c_{ox} \) is the capacitance of the oxide layer, \( \mu_o \) is the mobility, \( v_{th} \) is the threshold voltage, \( v_{gs} \) is the gate to source voltage, \( v_{ds} \) is the drain to source voltage and \( v_t = \frac{kT}{q} \) is the thermal voltage. The inverse slope of \( I_{ds} \) versus \( v_{gs} \) characteristic of a MOS device is called sub-threshold slope and is given by \( S_t = 2.3 \cdot \frac{m \cdot k \cdot T}{q} \) [4]. Sub-threshold slope indicates how effectively a transistor can be turned off when gate voltage is below the threshold voltage [6].

Reverse biasing substrate to source junction [10, 11] increases threshold voltage thereby reducing the sub-threshold current. In short channel devices, the depleted source and drain regions interact with each other reducing the potential barrier [6], thereby reducing the threshold voltage which increases the sub-threshold leakage current.

2.2.3 Reverse-Bias Junction Leakage

P-N junctions between the source/drain and substrate are reverse biased allowing a small amount of reverse bias leakage current to flow across the junctions [6]. Reverse bias leakage increases with high electric field strengths [11]. The magnitude of the leakage current depends on the area of diffusion regions and doping concentration [10]. Large doping concentrations increase leakage...
current. Reduction in the substrate doping near diffusion regions reduces the reverse bias junction leakage [6].

2.2.4 Gate Induced Drain Leakage
This leakage occurs when potential of the channel region is almost the same as potential of substrate [6]. Suppose a negative gate bias is applied to an NMOS device, the drain region underneath the gate will be depleted of any electrons [6]. Increase in the electric field results in avalanche breakdown generating minority carriers [6]. These minority carriers are swept from the channel to the substrate resulting in gate induced drain leakage current [12].

2.2.5 Punch through Leakage
In short channel devices, source and drain regions extend into channel reducing the effective length of channel [12]. When the reverse bias voltage across the drain-substrate and source-substrate junctions is increased, the diffusion regions merge with each other resulting in large leakage current flowing between drain and source junctions [6]. This leakage current is called punch through leakage current [12].

2.2.6 Hot Carrier Injection
High electric fields near the channel-oxide interface causes electrons or holes to overcome the potential barrier and enter oxide layer causing hot carrier effect [11]. These trapped electrons or holes in the oxide layer change the threshold voltage thereby affecting the sub-threshold leakage current [11]. Scaling down the supply voltage with the device dimensions controls the leakage current.

2.3 Conclusion
Scaling down the transistor channel lengths below 45 nm in deep sub-micron technology results in reducing both the supply and threshold voltages to 0.8V and 0.22V respectively. Devices with
threshold voltages as low as 0.22V result in an increase in the leakage power dissipation. However, controlling threshold voltages of the transistors is difficult in the sub-micron technologies. High computational capability is achieved by switching signals faster thereby increasing dynamic switching power. Hence the issues of static leakage power and dynamic power continue to play an important role in the processor design. Chapter 3 presents various state of art techniques for reducing both dynamic and static leakage power.
Chapter 3 Background Work

This chapter outlines the various state of art techniques for reducing both static leakage power and dynamic power. In the current literature, techniques were proposed to reduce dynamic and static leakage power at different levels of abstraction. In the following subsections architectural and circuit level techniques for power reduction will be discussed.

3.1 Static Power Reduction Techniques

3.1.1 Cache Decay

This technique uses an assumption that caches tend to store items that will not be referenced in future. Savings in the leakage power and performance overhead can be achieved by placing the unused cache items in the deep sleep mode before being evicted from cache [13]. Time based leakage control technique is used to turn OFF the unused cache items. In this technique unused lines are observed over a period of time and when it is likely that no further accesses are made, the cache lines are put to sleep reducing power dissipation [13]. Turning off the line prematurely would result in a miss and increase the access time while turning off the line late would result in an increase in the leakage power dissipation [13]. Small interval lengths increase the leakage power dissipation [13]. An adaptive variant of the time based leakage control policy initially chooses small decay time interval, length of the interval is varied based on number of unused cache lines [13]. Probable improvements can be made by dynamically selecting the time interval based on the level of parallelism between the instructions and also in the direction of using probabilistic methods to predict the time interval size.

3.1.2 Selectively activated cache

In this technique the cache is divided into blocks and threshold voltage of the each block is varied based on the utilization of the individual blocks [14]. Frequently accessed blocks are kept
awake while the rest of the blocks are put to sleep. Dual threshold voltage technique [15] as shown in the Figure 3.1 is used to vary threshold voltages of the individual blocks. During active mode the SRAM circuit is connected to supply and ground rails [15]. During the sleep mode threshold voltages of the internal transistors are increased due to substrate bias effect, thereby reducing the static leakage current [15]. A prediction table keeps track of the addresses of active blocks in the cache [14]. Special register called previous block address register stores the address of the recently accessed cache block. When the address of the currently accessed block matches the address in the previous block address register, the corresponding line is awaken and address of the block is added to the prediction table [14]. Increasing the entries in the prediction table achieves high performance thereby increasing the leakage power [14]. Block prediction [14] can be made based on the type of instruction thereby reducing the delay. Improvements can be sought in the direction of finding a better algorithm to add unused entries to prediction table.

![FIGURE 3.1: Dual Threshold Voltage Architecture [14]](image)

3.1.3 Gated-$V_{dd}$ Technique

In this technique supply voltage is turned OFF to minimize power dissipation in unused parts of the circuitry [17]. Control transistors that are added between the circuit and the supply paths are
turned ON in the active parts of the circuit and turned OFF in the unused parts of the circuit. Reduction in the leakage power is achieved due to the stacking effect introduced by the additional control transistors [17]. Implementation of this technique on the SRAM cell is shown in the Figure 3.2 where the gated control transistor is placed between ground rail and circuit reducing the standby leakage power.

![Gated Supply Technique][1]

FIGURE 3.2: Gated Supply Technique [17]

The gated-$V_{dd}$ control NMOS transistor in Figure 3.2 must be made sufficiently wide to maintain the speed. Sharing a single gated control transistor among multiple circuit blocks requires careful sizing of the gated-$V_{dd}$ control transistor [17].

3.1.4 Leakage Feedback Technique

In this technique the state of circuit is preserved by re-circulating the data through the circuit during the standby mode [18]. In the Figure 3.3 P1 and N1 transistors are sleep devices and P2 and N2 transistors are helper sleep devices having high threshold voltage [18]. During the active mode both the sleep and helper sleep devices are turned on allowing the circuit to charge/discharge to supply/ground voltages [18]. During the standby state either one of the helper sleep devices is turned on to preserve state of the circuit. Glitches in the input signal during idle state has no effect on the stored data value in the circuit.
3.1.5 Stack Technique

In this technique each transistor is split into two transistors as shown in the Figure 3.4. Consider an inverter circuit, when the input to circuit is a ground voltage the PMOS transistor is turned ON while both the NMOS transistors are turned OFF. Leakage current flowing through the NMOS transistor stack reduces due to the increase in the source to substrate voltage in the top NMOS transistor and also due to an increase in the drain to source voltage in the bottom NMOS transistor [19]. This technique introduces considerable performance overhead due to the stacking of transistors.

3.1.6 Sleepy Keeper

In this technique low threshold NMOS transistor is placed in parallel to a high threshold voltage PMOS sleep transistor in the pull-up network and low threshold voltage PMOS transistor is
placed in parallel to a high threshold voltage NMOS sleep transistor as shown in the Figure 3.5. During the active mode, both the sleep devices and additional MOS transistors are turned on to improve circuit performance [20]. During the standby mode, the sleep transistors are turned off thereby reducing the leakage current [20]. Additional MOS transistors preserve state of the circuit during idle mode. This technique reduces the output drive strength based on the fact that PMOS device passes a weak zero and NMOS device passes a weak 1, resulting in a delay increase.

![Figure 3.5: Sleepy Keeper](image)

**FIGURE 3.5: Sleepy Keeper [20]**

### 3.2 Dynamic Power Reduction Techniques

#### 3.2.1 Gate Freezing

This technique involves automatic circuit transformation for reducing the glitch power [21]. The gates with high spurious switching activity are replaced with functionally equivalent gates which are insensitive to input signal transitions by asserting a control signal [21]. Small subset of gates is used to generate the control signal. Asserting the control signal low, ensures gate is insensitive to input signal transitions [21]. Glitches are eliminated by keeping control signal low to stabilize inputs at the start of clock period and then raising clock signal high till the end of clock period.
Gates are clustered according to their arrival time to reduce the wiring overhead. This technique increases area overhead by the addition of the control logic circuitry.

### 3.2.2 Transistor Sizing

This technique achieves savings in power and delay by optimal sizing of transistors in critical and non-critical paths [22]. Delay tolerance indicates the possible worst case delay for each transistor in the circuit. The transistor sizes in non-critical paths are scaled down without affecting the tolerable delay of each transistor [22]. Reduction in the delay is achieved by searching for the most critical path instead of ‘N’ number of critical paths [22] to size devices for optimal performance. The critical path search is repeated till the user defined number of iterations and user defined improvement ratio are reached [22]. Transistor sizing in non-critical paths gives rise to new critical paths in the circuit. Most of the circuits in real time have more critical paths, where the technique achieves less savings in the dissipated power.

### 3.2.3 Guarded Evaluation

The technique is used to identify unused parts of the circuit that can be turned off without affecting the overall delay of the circuit [23]. Latches are used as transition barriers. When the latch is enabled, inputs to the latches are passed to the circuits to perform useful computation [23]. When enable goes low latches retain their previous state preventing the inputs from reaching the circuit. In the Figure 3.6, guard logic is placed before the combinational circuit to selectively turn off the inputs to the parts of the combinational circuit when not needed. Performance improvement and reduction in the area are achieved by using existing signals to generate clock disable signal [23].
3.2.4 Precomputation Logic

The technique selectively computes the output values of circuit one clock cycle before they are required [23]. The precomputed values reduce switching in the subsequent cycles. Output from the precomputation logic is used to enable/disable registers that feed inputs to combinational circuits [23]. Precomputation logic size is kept under control since increase in the size of precomputation logic could dissipate more power that would offset savings achieved by blocking the switching activity [23]. In the Figure 3.7 below output from the precomputation logic g1 and g2, turns on/off the latch R2 to allow/block the inputs reaching the combinational block A.
3.2.5 Dynamically resizable cache

Savings in the dynamic power dissipation is achieved by resizing the cache to reduce the switching activity in the circuits. Minimum size the cache can assume is called size-bound [16]. Miss-bound allows cache to adapt to working set and bound miss rate in each interval [16]. An adaptive mechanism monitors cache in fixed time intervals called sense intervals [16]. Miss counter counts number of 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 the miss-bound [16]. Inaccurate estimation of the cache size will result in redundant accesses to higher levels of memory thereby increasing delay penalty and dynamic power dissipation [16]. Mechanisms to detect optimal sense interval length and repeated sizing are required to reduce unnecessary accesses to memory. Improvements would be in finding a mechanism for calculating the time between the application phases that would be long enough to offset the resizing overhead [16].

3.3 Conclusion

Dynamic power reduction techniques explained above increase area overhead and possibly introduce new critical paths by adding additional control logic to reduce power. Static power reduction techniques introduce area and performance overhead by adding high threshold voltage transistors and increasing the number of transistors in the stack to reduce leakage power during standby state. Chapter 4 explains modules present in the two level cache model, which serves as the test environment.
Chapter 4 System Level Design

Multiple level caches tend to improve system performance by reducing the miss rate while contributing significantly towards power dissipation. A two level cache system consisting of a level 1 split cache and level 2 unified cache was designed as a test bed for a two way superscalar architecture. Overview on the two level cache system and functionality of each of the cache sub-modules are discussed below in detail.

FIGURE 4.1: Two Level Cache System
4.1 Two Level Cache System

The two level cache system shown in the Figure 4.1 consists of sub-modules like CPU buffer, instruction translation look-aside buffer, data translation look-aside buffer, level 1 instruction cache, level 1 data cache, level 2 unified cache and write buffer. Outputs of the sub-modules are valid during the positive edge of clock and when the reset is low.

Virtual addresses IREAD_VIRADDR, DREAD_VIRADDR of the instructions and data requested by the CPU are sent to instruction TLB and data TLB. Virtual address WRITE_VIRADDR of the data sent by CPU is stored in the CPU buffer. CPU buffer stores the address WRITE_VIRADDR and the data WRITE_DATA to reduce the CPU delay overhead, and the output buffered address DWRITE_VIRADDR and data WRITEDATA_OUT are sent to data TLB and level 1 data cache respectively.

When the instruction TLB is enabled IREAD_EN during a cache read, virtual address IREAD_VIRADDR is translated to physical address I_PHYADDR_READ. During a cache read, when the data TLB is enabled DREAD_EN the virtual address DREAD_VIRADDR is translated to the physical address D_PHYADDR_READ. When the data TLB is enabled WRITE_EN during a cache write, virtual address DWRITE_VIRADDR is translated to physical address PHYADDR_WRITE.

Set associative level 1 instruction cache is divided into sets. Each set is divided into lines and each line is divided into blocks. Each block consists of valid tag, dirty tag, address tag, data tag and LRU tag. Valid tag indicates if the address location in the cache contains valid data/instruction. When dirty tag is set, the data in the cache location is missing from main memory. Address tag provides the location of the stored data. LRU tag indicates the usage of the
instruction. Data tag provides the actual data stored in the cache. During a cache miss, write-back with write-allocate policy is used.

The Level 1 instruction cache is enabled I_READ to find an input address I_PHYADDR_READ match stored in the cache. When an address match occurs in Level 1 instruction cache, the instruction CPU_INST_OUT is sent to the CPU. When there is no matching address in the level 1 instruction cache, address of the missed instruction IREAD_ADDR_MISS is sent to level 2 unified cache.

Level 1 data cache is enabled D_READ to find an input address D_PHYADDR_READ match stored in the cache. When an address match occurs in Level 1 data cache, the data CPU_DATA_OUT is sent to CPU. When there is no matching address in the level 1 data cache, address of the missed data DREAD_ADDR_MISS is sent to level 2 unified cache. During a write to the level 1 data cache, the write data WRITEDATA_OUT is written to the address location PHYADDR_WRITE when the dirty flag is reset. When the dirty flag is set, the data DIRTY_DATA in the location PHYADDR_WRITE is written to the address DIRTY_ADDR location in level 2 cache and also to the write buffer location L1_BUF_ADDR.

During a level 1 instruction cache miss, level 2 unified cache is enabled I_READ_EN to find miss address IREAD_ADDR_MISS match stored in the level 2 unified cache. When the address match occurs in the level 2 unified cache, the data CPU_INST_OUT is sent to the CPU and also to the location L1_IMISS_ADDR in the level 1 instruction cache by enabling L1_IMISS_EN control signal. When there is no match, the miss address L2_IADDR_MISS is sent to the main memory.
Unified level 2 cache is enabled D_READ_EN to find miss data address DREAD_ADDR_MISS match stored in the level 2 unified cache. When the address match occurs, the data CPU_DATA_OUT is sent to CPU and also to the location L1_DMISS_ADDR in the level 1 data cache by enabling L1_DMISS_EN. During a miss in the level 2 unified cache, the address L2_DADDR_MISS of the missed data is sent to main memory. The dirty data DIRTY_DATA sent by level 1 data cache is stored in the location DIRTY_ADDR when the control signal DIRTY_EN is enabled and the dirty tag is reset. When the dirty tag is set, the data from the location DIRTY_ADDR is written to the location L2_BUF_ADDR in write buffer.

Write buffer stores the data L1_BUF_DATA sent by the level 1 data cache in the location L1_BUF_ADDR. Data L2_BUF_DATA sent by the level 2 unified cache is stored in the location L2_BUF_ADDR. When main memory is idle, the data L1_M_BUF_DATA sent by the write buffer is written to the location L1_M_BUF_ADDR. Data L2_M_BUF_DATA sent by the write buffer is stored in the location L2_M_BUF_ADDR.

Main memory looks-up the miss instruction L2_IADDR_MISS request sent by the level 2 unified cache. Miss address found in the main memory is sent to the location MEM_IL1_ADDR_MISS in the level 1 instruction cache and also sent to the location MEM_IL2_ADDR_MISS in the level 2 unified cache. Data in the miss address location in main memory is sent to MEM_IL1_INST_MISS in the level 1 instruction cache and also sent to MEM_IL2_INST_MISS in the level 2 unified cache. Miss data request L2_DADDR_MISS sent by the level 2 instruction cache is looked-up by main memory. Miss address location found in the main memory is sent to location MEM_DL1_ADDR_MISS in the level 1 data cache and also sent to the location MEM_DL2_ADDR_MISS in the level 2 unified cache. Data in the miss
address location in the main memory is sent to MEM_DL1_DATA_MISS and also sent to address MEM_DL2_DATA_MISS.

### 4.2 CPU Buffer

This module buffers the data sent by the CPU. When CPU buffer gets full, buffered data is written to the data cache. To minimize the delay in writing buffered data to data cache, size of the buffer is reduced. Data coherency between the CPU buffer and data cache need to be maintained, to prevent false data from being read by CPU. To prevent an incorrect update in the data cache contents, size of the buffer is matched with the block size of the data cache.

#### 4.2.1 Design Specifications

Size of each entry in buffer = \textit{address bits} + \textit{data bits} = 32 + 32 = 64 bits

Total number of entries in buffer = 128

Total size of buffer = \textit{size of each entry} \times \textit{total number of entries in buffer} = 64 \times 128 = 1KB

A model CPU buffer is shown in the Figure 4.2. Table I in Appendix I explains functionality of each of the input and output pins in the model CPU buffer.

#### FIGURE 4.2: CPU Buffer

### 4.3 Write Buffer

This module temporarily stores the data from the cache, allowing the cache to serve the read requests, thereby preventing stall cycles during the read operation. Write buffer searches the
main memory for a free time slot, during which data stored in the write buffer is written to main memory. Write buffer includes a logic that checks whether a read address matches any of the addresses waiting in the write buffer. When a match is detected, data is read back from the buffer instead of the main memory reducing the data access time. FIFO policy is used to send and retrieve information from the write buffer.

### 4.3.1 Design Specifications

Size of each entry in write buffer = \( \text{address bits} + \text{data bits} = 32 + 512 = 544 \text{ bits} \)

Total number of entries in buffer = 128

Total size of buffer = \( \text{size of each entry} \times \text{total number of entries in buffer} \)

\[ = 544 \times 128 \approx 8KB \]

A model write buffer is shown in the Figure 4.3. Table II in Appendix I explains functionality of each of the input and output pins in the model write buffer.

![Write Buffer Diagram](image)

**FIGURE 4.3: Write Buffer**

### 4.4 Translation Look-Aside Buffer

The TLB is used to improve speed of virtual address translation by storing the addresses of recently accessed page table entries, thereby reducing instruction/data accesses to main memory.

During a virtual memory access, CPU searches the TLB for the virtual page number of the page being accessed, operation known as TLB look-up. When the TLB entry is found with a matching VPN, a TLB hit occurred and the page table entry in the TLB is the physical address requested.
by the cache. When a miss occurs, operation of CPU is suspended until a new value is loaded from the main memory into the TLB entry where miss occurred.

### 4.4.1 ITLB Design Specifications

The instruction TLB is a 4 way set associative cache.

Total number of sets in instruction TLB = 4

Total size of instruction TLB = 16 KB

Size of each set in the instruction TLB = \( \frac{\text{total size of instruction TLB}}{\text{total number of sets in instruction TLB}} = \frac{16}{4} \text{ KB} = 4 \text{ KB} \)

Size of each block in the instruction TLB = physical address bits + (valid bits+ virtual address bits)*offset value = 22 + (1+21)16 = 374

Total number of lines in each set = \( \frac{\text{size of each set in the instruction TLB}}{\text{size of each block in the instruction TLB}} = \frac{4 \times 1024 + 8}{374} \approx 87 \)

Total offset bits = \( \ln_2(\text{offset value}) = \ln_2(16) = 4 \)

Total index bits = \( \ln_2(\text{total number of lines in each set}) = \ln_2(87) \approx 7 \)

Total number of tag bits = \( \text{total address bits} - (\text{total offset bits} + \text{total index bits}) = 32 - 4 + 7 = 21 \)

A model instruction translation look-aside buffer is shown in the Figure 4.4. Table III in Appendix I explains functionality of each of the input and output pins in the model instruction translation look-aside buffer.

---

**FIGURE 4.4: Instruction Translation Look-Aside Buffer**

4.4.2 DTLB Design Specifications

The data TLB is a 4 way set associative cache.

Total number of sets in data TLB = 4

Total size of data TLB = 64 KB

Size of each set in the data TLB = \( \frac{total\ size\ of\ data\ TLB}{total\ number\ of\ sets\ in\ data\ TLB} = \frac{64}{4} KB = 16 \) KB

Size of each block in the data TLB = physical address bits + (valid bits + virtual address bits)*offset value = 20 + (1+19)*16 = 340

Total number of lines in each set = \( \frac{size\ of\ each\ block\ in\ the\ instruction\ TLB}{size\ of\ each\ block\ in\ the\ data\ TLB} \approx 385 \)

Total offset bits = \( ln_2(\text{offset value}) = ln_2(16) = 4 \)

Total index bits = \( ln_2(\text{total number of lines in each set}) = ln_2(385) \approx 9 \)

Total number of tag bits = total address bits – (total offset bits + total index bits) = 32 – 4 + 9 = 19

A model data translation look-aside buffer is shown in the Figure 4.5. Table IV in Appendix I explains functionality of each of the input and output pins in the model data translation look-aside buffer.
4.5 Cache

Cache is a high speed buffer where frequently accessed instructions or data can be temporarily stored. Each entry in the cache consists of address tag, data and a valid bit. Address tag stores the location of the entry, data field has the information requested by the CPU and the valid bit indicates whether the entry is valid. When data requested by the CPU is found in the cache, hit occurs. During an address miss an address request is sent to a high level memory. Principle used in cache for storing data is locality of reference. Principle of locality states that the currently accessed data and its logically adjacent data are more likely to be used in the near future. Performance of the cache depends on the access time to a cache during a hit and probability of finding the data in the cache.

4.5.1 Instruction CACHE Design Specifications

The level 1 instruction cache is an 8 way set associative cache.

Total number of sets in level 1 instruction cache = 8

Total size of level 1 instruction cache = 128 KB

Set size in the level 1 instruction cache = \( \frac{\text{total size of level 1 instruction cache}}{\text{total number of sets in level 1 instruction cache}} \)

= \( \frac{128}{8} \) KB = 16 KB

Size of each block in the level 1 instruction cache = physical tag bits + (valid bits + data bits) * offset value = 24 + (1 + 512) * 16 = 8232 bits

Total number of lines in each set = \( \frac{\text{size of each set in the level 1 instruction cache}}{\text{size of each block in the level 1 instruction cache}} \)

= \( \frac{16 \times 1024 + 8}{8232} \) \approx 16

Total offset bits = \( \ln_2(\text{offset value}) = \ln_2(16) = 4 \)

Total index bits = \( \ln_2(\text{total number of lines in each set}) = \ln_2(16) \approx 4 \)

Total number of tag bits = total address bits – (total offset bits + total index bits)

= 32 – 4 + 4 = 24
A model level 1 instruction cache is shown in the Figure 4.6. Table V in Appendix I explains functionality of each of the input and output pins in the model level 1 instruction cache.

4.5.2 Data CACHE Design Specifications

The level 1 data cache is an 8 way set associative cache.

Total number of sets in level 1 data cache = 8

Total size of level 1 data cache = 256 KB

Set size in the level 1 data cache = \( \frac{\text{total size of level 1 data cache}}{\text{total number of sets in level 1 data cache}} \)

\[ = \frac{256}{8} \text{ KB} = 32 \text{ KB} \]

Size of each block in the level 1 data cache = physical tag bits + (valid bits + dirty bits + data bits) * offset value = 23 + (2 + 512) * 16 = 8247 bits

Total number of lines in each set = \( \frac{\text{size of each block in the level 1 data cache}}{\text{size of each block in the level 1 data cache}} \)

\[ = \frac{32 \times 1024 \times 8}{8247} \approx 32 \]

Total offset bits = \( \ln_2(\text{offset value}) = \ln_2(16) = 4 \)

Total index bits = \( \ln_2(\text{total number of lines in each set}) = \ln_2(32) \approx 4 \)

Total number of tag bits = total address bits – (total offset bits + total index bits)

\[ = 32 - 4 + 5 = 23 \]

A model Level 1 data cache is shown in the Figure 4.7. Table VI in Appendix I explains functionality of each of the input and output pins in the model level 1 data cache.
4.5.3 Level 2 Unified CACHE Design Specifications

Total number of sets in level 2 unified cache = 8

Total size of level 2 unified cache = 1 MB

Set size in the level 2 unified cache = \( \frac{\text{total size of level 2 unified cache}}{\text{total number of sets in level 2 unified cache}} \)

\[ = \frac{1024}{8} KB = 128 KB \]

Size of each block in the level 2 unified cache = \( 21 + (2 + 512) \times 16 = 8245 \) bits

Total number of lines in each set = \( \frac{\text{size of each set in the level 2 unified cache}}{\text{size of each block in the level 2 unified cache}} \)

\[ = \frac{128 + 1024 + 8}{8245} \approx 127 \]

Total offset bits = \( \ln_2(\text{offset value}) = \ln_2(16) = 4 \)

Total index bits = \( \ln_2(\text{total number of lines in each set}) = \ln_2(127) \approx 7 \)

Total number of tag bits = total address bits – (total offset bits + total index bits)

\[ = 32 - 4 + 7 = 21 \]

A model Level 2 unified cache is shown in the Figure 4.8. Table VII in appendix I explains functionality of each of the input and output pins in the model level 2 unified cache.
4.6 Main Memory

Miss address request sent by the cache is split into row and column addresses to access the contents of the main memory. Main memory also services the virtual address requests generated
by the TLB’s. A model of the main memory is shown in the Figure 4.9. Table VIII in Appendix I explains functionality of each of the input and output pins in the model main memory.

**FIGURE 4.9: Main Memory**
4.7 Conclusion

The modules present in the two level cache model have been fully tested on Modelsim to verify the complete functionality. Synthesizable working model of the two level cache is obtained by using Leonardo Spectrum. The two level cache model is used as a test-bed for running power analysis. The proposed power reduction techniques, adaptive phase tag cache and GALEOR will be discussed in the Chapter 5.
Chapter 5 Proposed Power Reduction Techniques

Future generation processors employ long addresses to access data in memory resulting in an increase in the tag field sizes. Increasing tag sizes result in an increase in the number of comparisons, thereby increasing the dynamic power dissipation. Static leakage power is also becoming a major concern in design of low power devices. The proposed techniques reduce both the dynamic power dissipation caused by accessing the tag fields and the leakage current flowing through the unused parts of the circuit.

5.1 Adaptive Phase Tag Technique

Phase tag cache splits the address tag and tag matching into two phases. During the phase I, first half of the cache tag is compared with the first half of incoming address tag. Phase II compares second half of the cache tag with the second half of the incoming address tag. Width of the tags remain unchanged throughout the execution of the program. Adaptive phase tag cache splits the tag arrays & tag matching into two phases by varying the width of the tags during program execution. Execution time of an application is divided into windows which samples a typical instruction mix. During the phase I, first half of the incoming tag address is matched against the first half of the cache tag array. When there is a tag match during phase I comparison, the second phase tag matching is triggered. During a tag mismatch in the phase I comparison, the second phase of tag matching is disabled. Disabling the comparisons during the phase II reduces the switching power dissipation. During the phase II, second half of the cache tag array is matched against the second half of the incoming address tag. When a tag match occurs in phase II, requested data is sent to the CPU. Hit ratio is used as a performance metric to monitor cache performance. Hit ratio is defined as

\[ HR = \frac{\text{Total number of hits in the first phase of cache}}{\text{Total number of accesses to cache}} \]
The Table 1 below shows allocation of tag bits for a model instruction cache having 23 bit wide tag address.

<table>
<thead>
<tr>
<th>Hit Ratio (HR)</th>
<th>Width of Tag1</th>
<th>Width of Tag2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>0.2</td>
<td>18</td>
<td>5</td>
</tr>
<tr>
<td>0.3</td>
<td>16</td>
<td>7</td>
</tr>
<tr>
<td>0.4</td>
<td>14</td>
<td>9</td>
</tr>
<tr>
<td>0.5</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>0.6</td>
<td>9</td>
<td>14</td>
</tr>
<tr>
<td>0.7</td>
<td>7</td>
<td>16</td>
</tr>
<tr>
<td>0.8</td>
<td>5</td>
<td>18</td>
</tr>
<tr>
<td>0.9</td>
<td>2</td>
<td>21</td>
</tr>
</tbody>
</table>

**TABLE 5.1: Allocation of Tag bits based on the Hit Ratio**

Increase in the hit ratio results in reducing width of the first part of tag thereby reducing the dynamic power dissipation. Reduced width in the first part of tag leads to large number of accesses to high level caches increasing the delay overhead.

![Diagram](Image)

**FIGURE 5.1: Two Way Set Associative Adaptive Phase Tag Cache**

Model of a 2 way set associative adaptive phase tag cache is shown in the Figure 5.1. Index field value of the physical address is used to search lines in a set associative cache. Once the line is
found the comparators A and B are used to compare the first half of the physical address with the first half of the set associative cache tag address. During the second phase of tag matching, Tri-state comparators are used to reduce switching activity. Tri-state comparators are obtained by adding tri-state buffers to the inputs of a bi-state comparator. When the enable is low, a high impedance state results at the output. When the enable is high, tri-state buffers pass the input data to output. During a TAG1 hit, the set of tri-state comparators C and D are enabled to compare the TAG2 of the physical address with the TAG2 of the set associative cache tag address. During a valid cache hit, multiplexers are used to select the instruction and data requested by CPU based on the offset value. Hit ratio calculator decides the widths of tags during phase I and phase II based on the number of hits during phase I match. Increased cache tag sizes, result in an increase in the number of tri-state buffers and also increases complexity of the hit ratio calculator logic, thereby increasing the area overhead. Resizing cache tags introduces performance overhead due to the increase in the number of phase II comparisons.

5.2 GALEOR Technique:

Sub-threshold leakage contributes maximum for static power dissipation. Consider the NAND gate shown in the Figure 5.2 with the source, gate and drain regions of each transistor representing different nodes.

![FIGURE 5.2: Two Input NAND Gate](image)
Sub-threshold leakage currents flowing through a two input NAND gate for different input combinations are explained below.

i) $A = 0, B = 0$: When both the inputs are low, the PMOS transistors MP1 and MP2 are turned ON while the NMOS transistors MN1 and MN2 are turned OFF. Reduced sub-threshold leakage currents flow through the NMOS transistors MN1 & MN2 due to the stack effect.

ii) $A = 0, B = 1$: The PMOS transistor MP1 and the NMOS transistor MN2 are turned ON while the PMOS transistor MP2 and the NMOS transistor MN1 are turned OFF. Sub-threshold leakage currents flowing through the OFF transistors MP2 & MN1 remain unaffected.

iii) $A = 1, B = 0$: PMOS transistor MP2 and NMOS transistor MN1 is turned ON while the PMOS transistor MP1 and the NMOS transistor MN2 are turned OFF. Sub-threshold leakage currents flowing through the OFF transistors MP1 & MN2 remain unaffected.

iv) $A = 1, B = 1$: When both the inputs are high, the PMOS transistors MP1 and MP2 are turned OFF while the NMOS transistors are turned ON. Sub-threshold leakage current flows through the PMOS transistors MP1 and MP2.

Leakage currents flowing through a two input NAND gate are summarized in the Table 5.2.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>MP1</th>
<th>MP2</th>
<th>MN1</th>
<th>MN2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ON</td>
<td>ON</td>
<td>$I_{\text{SUB}}: t \rightarrow w$</td>
<td>$I_{\text{SUB}}: w \rightarrow x$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>ON</td>
<td>$I_{\text{SUB}}: q \rightarrow t$</td>
<td>$I_{\text{SUB}}: t \rightarrow w$</td>
<td>ON</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$I_{\text{SUB}}: p \rightarrow t$</td>
<td>ON</td>
<td>ON</td>
<td>$I_{\text{SUB}}: w \rightarrow x$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$I_{\text{SUB}}: p \rightarrow t$</td>
<td>$I_{\text{SUB}}: q \rightarrow t$</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

**TABLE 5.2**: Leakage current in two input NAND gate
When both the inputs are low, natural stack effect is introduced thereby reducing the leakage current flowing through the circuit. Rest of the input combinations does not introduce a natural stack effect to reduce the leakage current. To reduce the leakage current flowing through the circuit, forced stack effect needs to be introduced. The proposed technique, GALEOR, introduces two additional transistors in the circuit to introduce a force stack structure in the circuit independent of the input combination. Force stack structure added reduces the leakage current by increasing the resistance of the leakage path.

GALEOR technique implemented on a two input NAND gate is shown in the Figure 5.3.
Additional gated leakage NMOS transistor GLT1 and a PMOS transistor GLT2 are introduced between the output and pull-up circuitry and the output and pull-down circuitry. Sub-threshold leakage currents flowing through a two input gated leakage transistor NAND gate for different input combinations are explained below.

i)  A = 0 , B = 0 : When both the inputs are low, the PMOS transistors MP1 and MP2 are turned ON while the NMOS transistors MN1 and MN2 are turned OFF. Intermediate node t is charged to supply voltage while the intermediate node r remains at the supply voltage. Gated leakage NMOS transistor GLT1 turns ON while the gated leakage PMOS transistor GLT2 is turned OFF since the gate voltage on the transistors is less than the threshold voltage. Gated leakage transistor GLT2 introduces a forced stack structure with the NMOS transistor stack further reducing the leakage current flow.

ii) A = 0 , B = 1 : The PMOS transistor MP1 and the NMOS transistor MN2 are turned ON while the PMOS transistor MP2 and the NMOS transistor MN1 are turned OFF. Intermediate node t is charged to supply voltage while the intermediate node r remains at the supply voltage. Gated leakage NMOS transistor turns ON while the gated leakage PMOS transistor is turned OFF. Gated leakage PMOS transistor GLT2 forces a stack effect with the OFF transistor MN1 thereby reducing the leakage current flow.

iii) A = 1 , B = 0 : The PMOS transistor MP2 and the NMOS transistor MN1 are turned ON while the PMOS transistor MP1 and the NMOS transistor MN2 are turned OFF. Intermediate node t is charged to supply voltage while the intermediate node r still remains at the supply voltage. Gated leakage NMOS transistor GLT1 is turned ON
while the gated leakage PMOS transistor GLT2 is turned OFF. Gated leakage PMOS transistor GLT2 forces a stack effect with the OFF transistor MN2 thereby reducing the leakage current flow.

iv) A = 1 , B = 1 : When both the inputs are high, the PMOS transistors MP1 and MP2 are turned OFF while the NMOS transistors are turned ON. Intermediate nodes t and r discharge to ground voltage. Gated leakage NMOS transistor GLT1 is turned OFF while the gated leakage PMOS transistor GLT2 is turned ON. Gated leakage NMOS transistor GLT1 forces a stack effect with the OFF transistors MP1 and MP2 thereby reducing the leakage current.

Sub-Threshold leakage currents flowing through a two input gated leakage transistor NAND gate are summarized in the Table 5.3.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>MP1</th>
<th>MP2</th>
<th>GLT1</th>
<th>MN1</th>
<th>MN2</th>
<th>GLT2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>( I_{\text{SUB}} : r \rightarrow v )</td>
<td>( I_{\text{SUB}} : v \rightarrow x )</td>
<td>( I_{\text{SUB}} : z \rightarrow r )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>ON</td>
<td>( I_{\text{SUB}} : q \rightarrow t )</td>
<td>ON</td>
<td>( I_{\text{SUB}} : r \rightarrow v )</td>
<td>( \text{ON} )</td>
<td>( I_{\text{SUB}} : z \rightarrow r )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>( I_{\text{SUB}} : p \rightarrow t )</td>
<td>ON</td>
<td>ON</td>
<td>( \text{ON} )</td>
<td>( I_{\text{SUB}} : v \rightarrow x )</td>
<td>( I_{\text{SUB}} : z \rightarrow r )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>( I_{\text{SUB}} : p \rightarrow t )</td>
<td>( I_{\text{SUB}} : q \rightarrow t )</td>
<td>( I_{\text{SUB}} : t \rightarrow z )</td>
<td>( \text{ON} )</td>
<td>( \text{ON} )</td>
<td>( \text{ON} )</td>
</tr>
</tbody>
</table>

**TABLE 5.3: Leakage current in two input Gated Leakage Transistor NAND gate**

### 5.3 Leakage Power Analysis in Memory Elements

#### 5.3.1 ‘D’ Flip – Flop:

The master slave ‘D’ flip-flop shown in the Figure 5.4 is built using NAND gates and INVERTERS. Leakage current flowing through the NAND gates must be reduced in order to achieve leakage savings in the D flip-flop. Reduction in the leakage must have a minimal impact on delay. Circuit delay is maintained by adding the gated leakage transistors GLT1, GLT2, GLT3 & GLT4 to the output gates G7 and G8. Addition of the gated leakage transistors reduces
the leakage current by introducing the forced stack structures as shown in the Table 5.4, thereby increasing the leakage path resistance in the pull-up and pull-down logic circuitry of the NAND gates. Gated leakage transistors reduce the output voltage swing, thereby reducing the current drive strength which increases the delay. Addition of the gated leakage transistors to the critical paths of the circuit is avoided to increase the performance overhead. Threshold voltages of the gated leakage transistors is slightly increased to further reduce leakage power dissipation.

<table>
<thead>
<tr>
<th>D</th>
<th>Clock</th>
<th>GLT1</th>
<th>GLT2</th>
<th>GLT3</th>
<th>GLT4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 → 1</td>
<td>Forced Stack</td>
<td>ON</td>
<td>ON</td>
<td>Forced Stack</td>
</tr>
<tr>
<td>1</td>
<td>0 → 1</td>
<td>ON</td>
<td>Forced Stack</td>
<td>Forced Stack</td>
<td>ON</td>
</tr>
</tbody>
</table>

**TABLE 5.4: Stacks added in a gated leakage D flip-flop**

**FIGURE 5.4: Gated Leakage D Flip-Flop**

**5.4 Conclusion:**

Adaptive phase tag cache technique reduces the switching power dissipation caused by accessing the tag fields in the long addresses. This technique introduces area overhead caused by the additional logic to reduce the dynamic switching power dissipation. Faulty calculation in the hit
rate leads to serious penalties like increasing the access time and power dissipation. The GALEOR technique presented above reduces the leakage current flowing through the transistors in the standby state. This leakage technique introduces minimal area overhead by selectively adding transistors to achieve considerable savings in the leakage power. Minimal performance overhead is achieved by adding the leakage transistors to maintain the output current drive strength of the circuit. Chapter 6 presents the results from dynamic power and static leakage power simulations for standard cells and storage elements.
Chapter 6 Results

Standard cell gates, memory storage elements and a two level cache system were chosen as benchmark circuits for running the static leakage power and dynamic switching power simulations for the 45nm technology. Leakage savings from the proposed static power reduction technique, GALEOR are compared against LECTOR and the original design without any optimization techniques implemented on. Power delay squared product was used to measure leakage power savings for all possible input combinations in each of the benchmark circuits.

6.1 Leakage Power Savings in Benchmark Circuits

6.1.1 Inverter

Leakage power savings in an inverter circuit for each of the possible input combinations is shown in Figure 6.1. Leakage current flowing through the PMOS transistor is larger than the leakage current through the NMOS transistor due to different transistor sizes. In LECTOR technique, additional leakage control transistors when turned OFF are biased near to the cut-off region creating a stack effect thereby reducing the leakage power dissipation. In GALEOR technique, additional gated leakage transistors when turned OFF are biased closer to the cut-off region, further reducing the leakage power dissipation compared to LECTOR.

![FIGURE 6.1: Leakage Power Dissipation in an Inverter](image-url)
6.1.2 Buffer

Leakage power savings in a buffer circuit for each of the possible input combinations is shown in Figure 6.2. During ‘0’ input, leakage current flows through the NMOS transistor in the first inverter & PMOS transistor in the second inverter. During ‘1’ input, leakage current flows through the PMOS transistor in the first inverter & NMOS transistor in the second inverter.

In LECTOR technique, additional leakage control transistors when turned OFF are biased near to the cut-off region creating a stack effect thereby reducing the leakage power dissipation. In GALEOR technique, additional gated leakage transistors when turned OFF are biased closer to the cut-off region, further reducing the leakage power dissipation compared to LECTOR.

![Figure 6.2: Leakage Power Dissipation in Buffer](image)

6.1.3 Two Input NAND

Leakage power savings in a two input NAND gate circuit for each of the possible input combinations is shown in Figure 6.3. During a ‘00’ input, both the NMOS transistors are turned OFF creating a stack effect to reduce the leakage current dissipation. During a ‘01’ or ‘10’ input, one of the NMOS and PMOS transistors are turned OFF which increases the leakage power. During a ‘11’ input both the PMOS transistors are turned OFF allowing large leakage current to flow through the circuit. In GALEOR, during the inputs ‘01’ & ‘10’, the gated leakage PMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation.
During the input ‘11’ the gated leakage NMOS transistor is turned OFF thereby reducing the leakage current.

6.1.4 Two Input NOR

Leakage power savings in a two input NOR gate circuit for each of the possible input combinations is shown in Figure 6.4. During a ‘00’ input, both the NMOS transistors are turned OFF allowing large leakage current to flow through the circuit. During a ‘01’ or ‘10’ input, one of the NMOS and PMOS transistors are turned OFF which increases the leakage power. During a ‘11’ input both the PMOS transistors are turned OFF creating a stack effect, thereby reducing the leakage power dissipation. In GALEOR, during the inputs ‘01’ & ‘10’, the gated leakage NMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘00’ the gated leakage PMOS transistor is turned OFF thereby reducing the leakage current.
6.1.5 Two Input AND

Leakage power savings in a two input AND gate circuit for each of the possible input combinations is shown in Figure 6.5. During a ‘00’ input, both the NMOS transistors in the NAND gate & PMOS transistor in the inverter are turned OFF reducing the leakage current flowing through the circuit. During a ‘01’ or ‘10’ input, one of the NMOS and PMOS transistors in the NAND gate and the PMOS transistor in the inverter are turned OFF thereby increasing the leakage power. During a ‘11’ input both the PMOS transistors in the NAND gate and the NMOS transistor in the inverter are turned OFF allowing high leakage current to flow through the circuit. GALEOR technique is implemented on the inverter in a two input AND gate to reduce the effect of delay. In GALEOR, during the inputs ‘01’ & ‘10’, the gated leakage NMOS transistor in the inverter is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘11’ the gated leakage PMOS transistor in the inverter is turned OFF thereby reducing the leakage current. During the input ‘00’ the gated leakage NMOS transistor in the inverter is turned OFF thereby reducing the leakage current.

![FIGURE 6.5: Leakage Power Dissipation in a Two Input AND Gate](image)

6.1.6 Two Input OR

Leakage power savings in a two input OR gate circuit for each of the possible input combinations is shown in Figure 6.6. During a ‘00’ input, both the NMOS transistors in the NOR gate & PMOS transistor in the inverter are turned OFF allowing high leakage current to flow
through the circuit. During a ‘01’ or ‘10’ input, one of the NMOS and PMOS transistors in the NOR gate and the PMOS transistor in the inverter are turned OFF thereby increasing the leakage power. During a ‘11’ input both the PMOS transistors in the NOR gate and the NMOS transistor in the inverter are turned OFF reducing the leakage current flowing through the circuit. GALEOR technique is implemented on the inverter in a two input OR gate to reduce the effect of delay. In GALEOR, during the inputs ‘01’ & ‘10’, the gated leakage PMOS transistor in the inverter is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘11’ the gated leakage PMOS transistor in the inverter is turned OFF thereby reducing the leakage current. During the input ‘00’ the gated leakage NMOS transistor in the inverter is turned OFF thereby reducing the leakage current.

![FIGURE 6.6: Leakage Power Dissipation in a Two Input OR Gate](image)

6.1.7 Three Input NAND

Leakage power savings in a three input NAND gate circuit for each of the possible input combinations is shown in Figure 6.7. During a ‘000’ input, the three NMOS transistors are turned OFF creating a stack effect to reduce the leakage current dissipation. During a ‘010’ or ‘100’ or ‘001’ input, one of the PMOS and two NMOS transistors are turned OFF reducing the leakage power. During a ‘110’ or ‘011’ or ‘101’ input, one of the NMOS and two of the PMOS devices are turned OFF increasing the leakage power dissipation. During a ‘111’ input, three of the PMOS transistors are turned OFF allowing large leakage current to flow through the circuit.
In GALEOR, during the inputs ‘010’ & ‘100’ & ‘001’, the gated leakage PMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘011’ & ‘110’ & ‘101’, the gated leakage PMOS transistor is turned OFF to create a stack effect thereby reducing the power dissipation. During the input ‘111’ the gated leakage NMOS transistor is turned OFF thereby reducing the leakage current. During the input ‘000’ the PMOS gated leakage transistor turns OFF, creating a stack effect resulting in minimum leakage power.

FIGURE 6.7: Leakage Power Dissipation in a Three Input NAND Gate

6.1.8 Three Input NOR

Leakage power savings in a three input NOR gate circuit for each of the possible input combinations is shown in Figure 6.8. During a ‘000’ input, the three NMOS transistors are turned OFF allowing maximum leakage current to flow through the circuit. During a ‘010’ or ‘100’ or ‘001’ input, one of the PMOS and two NMOS transistors are turned OFF reducing the leakage power. During a ‘110’ or ‘011’ or ‘101’ input, one of the NMOS and two of the PMOS devices are turned OFF increasing the leakage power dissipation. During a ‘111’ input, three of the PMOS transistors are turned OFF creating a stack effect which reduces the leakage power dissipation. In GALEOR, during the inputs ‘010’ & ‘100’ & ‘001’, the gated leakage NMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘011’ & ‘110’ & ‘101’, the gated leakage NMOS transistor is turned OFF to
create a stack effect thereby reducing the power dissipation. During the input ‘111’ the gated leakage NMOS transistor is turned OFF creating a stack effect resulting in minimum leakage power. During the input ‘000’ the PMOS gated leakage transistor turns OFF, thereby reducing the leakage current.

![FIGURE 6.8: Leakage Power Dissipation in a Three Input NOR Gate](image)

### 6.1.9 Three Input AND

Leakage power savings in a three input AND gate circuit for each of the possible input combinations is shown in Figure 6.9. During a ‘000’ input, the three NMOS transistors in the NAND gate & the PMOS transistor in the inverter are turned OFF creating a stack effect to reduce the leakage current dissipation. During a ‘010’ or ‘100’ or ‘001’ input, one of the PMOS and two NMOS transistors in NAND gate & PMOS transistor in the inverter are turned OFF reducing the leakage power. During a ’110’ or ‘011’ or ‘101’ input, one of the NMOS and two of the PMOS devices in the NAND gate & PMOS transistor in the inverter are turned OFF increasing the leakage power dissipation. During a ‘111’ input, three of the PMOS transistors in the NAND gate and the NMOS transistor in the inverter are turned OFF allowing large leakage current to flow through the circuit. In GALEOR, during the inputs ‘010’ & ‘100’ & ‘001’, the gated leakage NMOS transistor in the inverter is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘011’ & ‘110’ & ‘101’, the gated
leakage NMOS transistor in the inverter is turned OFF to create a stack effect thereby reducing the power dissipation. During the input ‘111’ the gated leakage PMOS transistor in the inverter is turned OFF thereby reducing the leakage current. During the input ‘000’ the gated leakage NMOS transistor in the inverter turns OFF, resulting in reduced leakage power.

![Leakage Power Dissipation in a Three Input AND Gate](image)

**FIGURE 6.9: Leakage Power Dissipation in a Three Input AND Gate**

### 6.1.10 Three Input OR

Leakage power savings in a three input OR gate circuit for each of the possible input combinations is shown in Figure 6.10. During a ‘000’ input, the three NMOS transistors in the NOR gate & the NMOS transistor in the inverter are turned OFF allowing large leakage current to flow through the circuit. During a ‘010’ or ‘100’ or ‘001’ input, one of the PMOS and two NMOS transistors in NOR gate & PMOS transistor in the inverter are turned OFF reducing the leakage power. During a ‘110’ or ‘011’ or ‘101’ input, one of the NMOS and two of the PMOS devices in the NOR gate & PMOS transistor in the inverter are turned OFF increasing the leakage power dissipation. During a ‘111’ input, three of the PMOS transistors in the NOR gate and the PMOS transistor in the inverter are turned OFF allowing large leakage current to flow through the circuit. In GALEOR, during the inputs ‘010’ & ‘100’ & ‘001’, the gated leakage PMOS transistor in the inverter is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘011’ & ‘110’ & ‘101’, the gated leakage PMOS transistor in the inverter is turned OFF to create a stack effect thereby reducing the power...
dissipation. During the input ‘111’ the gated leakage PMOS transistor in the inverter is turned OFF thereby reducing the leakage current. During the input ‘000’ the gated leakage NMOS transistor in the inverter turns OFF resulting in reduced leakage power.

6.1.11 Two Input XOR

Leakage power savings in a two input XOR gate circuit for each of the possible input combinations is shown in Figure 6.11. During ‘00’, ‘01’, ‘10’ & ‘11’ inputs, two NMOS transistors & two PMOS transistors are turned OFF creating a stack effect to reduce the leakage power dissipation. In GALEOR, during the inputs ‘00’ & ‘11’, the gated leakage PMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘10’ & ‘01’ the gated leakage NMOS transistor is turned OFF thereby reducing the leakage current.
6.1.12 Two Input XNOR

Leakage power savings in a two input XNOR gate circuit for each of the possible input combinations is shown in Figure 6.12. During ‘00’, ‘01’, ‘10’ & ‘11’ inputs, two NMOS transistors & two PMOS transistors are turned OFF creating a stack effect to reduce the leakage power dissipation. In GALEOR, during the inputs ‘00’ & ‘11’, the gated leakage NMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘10’ & ‘01’ the gated leakage PMOS transistor is turned OFF thereby reducing the leakage current.

![FIGURE 6.12: Leakage Power Dissipation in a Two Input XNOR Gate](image)

6.1.13 ‘D’ Flip-Flop

Leakage power savings in a D flip-flop circuit for each of the possible input combinations is shown in Figure 6.13. During ‘d=0 clk=0’, ‘d=0 clk=1’, ‘d=1 clk=0’ & ‘d=1 clk=1’ inputs, two NMOS transistors stack, PMOS & an NMOS transistors at the output NAND gates are turned OFF reducing the leakage power dissipation. In GALEOR, during the inputs ‘00’ & ‘01’, the gated leakage PMOS transistor is turned OFF to create a stack effect thereby reducing the leakage power dissipation. During the input ‘10’ & ‘11’ the gated leakage NMOS transistor is turned OFF thereby reducing the leakage current.
6.1.14 ‘D’ Flip-Flop with Reset Enable

Leakage power savings in a D flip-flop circuit with reset enable for each of the possible input combinations is shown in Figure 6.14. During ‘d=0 clk=0 rst=1’ & ‘d=0 clk=1 rst=1’ inputs, the three NMOS in the three input NAND gate create stack effect, thereby reducing the leakage power. During ‘d=1 clk=0 rst=1’, ‘d=1 clk=1 rst=1’, ‘d=0 clk=0 rst=0’ & ‘d=0 clk=1 rst=0’ inputs, two NMOS transistors in the three input NAND gate create stack effect to reduce the leakage power. During ‘d=1 clk=0 rst=0’ & ‘d=1 clk=1 rst=0’ inputs, two NMOS transistors in the two input NAND gate are turned OFF creating stack effect to reduce the leakage power. In GALEOR, during inputs ‘d=0 clk=0 rst=1’ & ‘d=0 clk=1 rst=1’, gated leakage PMOS transistor is turned OFF in the three input NAND gate & gated leakage NMOS transistor in the two input NAND gate are turned OFF creating a stack effect to maximize reduction in the leakage power dissipation. During inputs ‘d=1 clk=0 rst=1’, ‘d=1 clk=1 rst=1’, ‘d=0 clk=0 rst=0’ & ‘d=0 clk=1 rst=0’, gated leakage NMOS transistor in the two input NAND gate & gated leakage PMOS transistor in the three input NAND gate create stack effect reducing the leakage power dissipation. During inputs ‘d=1 clk=0 rst=0’ & ‘d=1 clk=1 rst=0’, gated leakage PMOS transistor in the two input NAND gate & gated leakage NMOS transistor in the three input NAND gate is turned OFF creating a stack effect to reduce the leakage power.
FIGURE 6.14: Leakage Power Dissipation in a D Flip-Flop with Reset Enable

6.1.15 Static Leakage Power Delay Squared Product

Product of leakage power & delay square is used as a metric to measure leakage power savings in the combinational and sequential circuits. % improvement in the leakage power for the benchmark circuits is shown in the Figure 6.15. The two input NAND gate achieves better savings compared to inverter or buffer due to the fact that stack effect of 3 transistors which includes gated leakage transistor further reduces the leakage current. XOR and the XNOR gates achieve better leakage reduction compared to two input NAND/NOR or the three input NAND/NOR gates since stack effect is forced for all the input combinations thereby reducing the leakage current. Three input NAND/NOR achieve better savings in leakage reduction compared to a two input NAND/NOR gate due to the increase in the number of stack transistors in the cut-off.

FIGURE 6.15: Static Leakage Power savings in the benchmark Circuits
6.2 Leakage Power Savings in the Two Level Cache

The two level cache system was used as a test bed for running the leakage power simulations. GALEOR technique was implemented on the memory elements inside the cache to reduce the leakage power dissipation. Further reduction in the leakage power is achieved by implementing the leakage power technique at the unused sections of the circuit. Static leakage power delay squared product is used to measure the leakage power savings in the two level cache system. Savings in leakage power delay squared product for the two level cache system are shown in the Figure 6.16 & Figure 6.17. % Savings in the leakage power delay squared product for the cache system is shown in the Figure 6.18. The threshold voltage of the gated leakage transistors is increased. Leakage power savings in the Level 2 cache is maximum due to the presence of large number of unused pins in the memory elements & also increase in the number of memory elements. Implementing GALEOR technique on such pins would result in small performance overhead.

![Figure 6.16: Leakage Power Delay Squared Product Savings in Buffer’s & TLB’s](image1)

![Figure 6.17: Leakage Power Delay Squared Product Savings in Level 1, Level 2 Caches](image2)
6.3 Dynamic Power Savings in the Memory System

Two level cache system was also used as a test bed for running dynamic power simulations. Input instruction mix was divided into 5 windows, each of size 10. Large hit ratio’s reduce the number of tag comparisons thereby reducing the switching power dissipation. Dynamic power delay squared product is used to measure the dynamic power savings in the memory system. Dynamic power delay squared product savings & % savings for the memory system are shown in the Figure 6.19 & Figure 6.20.
6.4 Conclusion

Adaptive phase tag cache, achieves 14% savings in the dynamic power in the modules of the memory system while introducing minimal area overhead. GALEOR achieves 30% savings in the leakage power in the memory system. The leakage power technique also achieves 35% reduction in the leakage power in the standard cell gates. Minimal area overhead is introduced by the proposed leakage technique.
Chapter 7 Conclusion & Future Work

Scaling down the device sizes and reduced threshold voltages has improved the system performance while increasing the leakage power dissipation. Leakage power is becoming a critical design constraint in low power portable devices. State of art techniques for reducing both dynamic & static power were discussed. A two level cache system is used as a test bed for running dynamic & static power simulations. Power reduction techniques for reducing the dynamic switching power and static leakage power were presented. Proposed dynamic power reduction technique controls the tag width based on the hit ratio in a window of instructions. Proposed leakage power reduction technique introduces high threshold voltage transistors to create stack effect thereby reducing the leakage current. Adaptive phase tag cache technique achieved 12% reduction in the switching power & performance overhead on memory system with 5% area overhead. GALEOR technique achieved 25% savings in the leakage power & performance overhead while introducing 20% area overhead in the standard cell gates & memory elements. GALEOR technique also achieved 9% reduction in the leakage power & performance overhead while introducing minimal area overhead on the memory system.

7.1 Future Work

Dynamic power technique can be modified to achieve better savings in the switching power dissipation by calculating number of hits for a particular instruction type in a window. GALEOR technique could achieve higher savings in the leakage power by maintaining the current drive strength in the circuits. Control mechanism can be designed to place gated leakage transistors in the circuit based on the intermediate circuit values.
Chapter 8 Bibliography


Chapter 9 APPENDIX A

This section contains the functional pin description of all the signals in the two level cache
system.

9.1 CPU Buffer

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>addr_in1[31:0], addr_in2[31:0]</td>
<td>Virtual address generated by CPU</td>
</tr>
<tr>
<td>data_in1[31:0], data_in2[31:0]</td>
<td>Input data to be written to the buffer</td>
</tr>
<tr>
<td>write1, write2</td>
<td>Enable data writes to buffer</td>
</tr>
<tr>
<td>Clock</td>
<td>enables buffer to store data when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>clears the contents of the buffer when asserted high</td>
</tr>
<tr>
<td>addr_out1[31:0], addr_out2[31:0]</td>
<td>Virtual address output to the data TLB</td>
</tr>
<tr>
<td>data_out1[31:0], data_out2[31:0]</td>
<td>Output data written to the data cache</td>
</tr>
<tr>
<td>accept1, accept2</td>
<td>Enables CPU to stop sending data when asserted high</td>
</tr>
<tr>
<td>en1, en2</td>
<td>Enables data cache to receive data from CPU buffer</td>
</tr>
</tbody>
</table>

TABLE 9.1: Functional Pin Description of CPU Buffer

9.2 Write Buffer

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>buf_addr1_in[31:0], buf_addr2_in[31:0]</td>
<td>Physical address sent by cache</td>
</tr>
<tr>
<td>buf_data1_in[31:0], buf_data2_in[31:0]</td>
<td>Input data to be written to the buffer</td>
</tr>
<tr>
<td>buf_en1_in, buf_en2_in</td>
<td>Enables writing data to the buffer</td>
</tr>
<tr>
<td>clock1</td>
<td>enables buffer to store data when asserted high</td>
</tr>
<tr>
<td>clock2</td>
<td>enables buffer to output data when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>clears the contents of the buffer when asserted high</td>
</tr>
<tr>
<td>buf_addr1_out[31:0], buf_addr2_out[31:0]</td>
<td>Output address to the memory</td>
</tr>
<tr>
<td>buf_data1_out[31:0], buf_data2_out[31:0]</td>
<td>Output data written to the memory</td>
</tr>
<tr>
<td>buf_en1_out, buf_en2_out</td>
<td>Enables memory to receive data from buffer</td>
</tr>
<tr>
<td>full1, full2</td>
<td>Enables cache to stop sending data to buffer</td>
</tr>
</tbody>
</table>

TABLE 9.2: Functional Pin Description of Write Buffer

9.3 Instruction Translation Look-Aside Buffer

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
</table>
### TABLE 9.3: Functional Pin Description of Instruction Translation Look-Aside Buffer

#### 9.4 Data Translation Look-Aside Buffer

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>mem_tlb_en1, mem_tlb_en2</td>
<td>Enables writing of miss address to the TLB</td>
</tr>
<tr>
<td>mem_tlb_en1_write1, mem_tlb_en2_write2</td>
<td></td>
</tr>
<tr>
<td>mem_tlb_viraddr1[31:0], mem_tlb_viraddr2[31:0]</td>
<td>Miss address written to TLB by main memory</td>
</tr>
<tr>
<td>mem_tlb_viraddr1_write1[31:0], mem_tlb_viraddr2_write2[31:0]</td>
<td></td>
</tr>
<tr>
<td>vir_addr1[31:0], vir_addr2[31:0]</td>
<td>Input virtual address from the CPU</td>
</tr>
<tr>
<td>vir_en1, vir_en2</td>
<td>Enabled during virtual address input from CPU</td>
</tr>
<tr>
<td>Clock</td>
<td>Enables the TLB to search for the physical address when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>Clear the contents of TLB when asserted high</td>
</tr>
<tr>
<td>phy_addr1[31:0], phy_addr2[31:0]</td>
<td>Generates the physical address to the instruction cache</td>
</tr>
<tr>
<td>tlb_mem_en1, tlb_mem_en2</td>
<td>Control signal sent to memory during a TLB miss</td>
</tr>
<tr>
<td>tlb_mem_viraddr1[31:0], tlb_mem_viraddr2[31:0]</td>
<td>Virtual address request sent to the main memory during a TLB miss</td>
</tr>
<tr>
<td>physaddr1_en1, physaddr2_en2</td>
<td>Control signal sent to notify instruction cache about the physical address input</td>
</tr>
<tr>
<td>hit_1, hit_2</td>
<td>Enabled during a TLB hit</td>
</tr>
<tr>
<td>phy_addr1[31;0], phy_addr2[31;0]</td>
<td>Generates the physical addresses to the data cache</td>
</tr>
<tr>
<td>phy_addr1_write1, phy_addr2_write2</td>
<td></td>
</tr>
<tr>
<td>clock</td>
<td>Enables the TLB to search for the physical address when asserted high</td>
</tr>
<tr>
<td>phy_add1r1[31:0], phy_addr2[31:0]</td>
<td>Generates the physical addresses to the data cache</td>
</tr>
<tr>
<td>phy_addr1_write1, phy_addr2_write2</td>
<td></td>
</tr>
<tr>
<td>hit_1, hit_2, hit_3, hit_4</td>
<td>Enabled during a TLB hit</td>
</tr>
<tr>
<td>tlb_mem_en1, tlb_mem_en2</td>
<td>Control signal sent to the memory during a TLB miss</td>
</tr>
<tr>
<td>tlb_mem_en1_write1, tlb_mem_en2_write2</td>
<td></td>
</tr>
</tbody>
</table>
Virtual address request sent to the memory during a TLB miss

Control signal sent to notify instruction cache about the physical address input

### TABLE 9.4: Functional Pin Description of Data Translation Look-Aside Buffer

#### 9.5 Level 1 Instruction Cache

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2_addr_miss1[31:0], L2_addr_miss2[31:0]</td>
<td>Address of the missed data written to the Level 1 instruction cache</td>
</tr>
<tr>
<td>L2_miss_en1, L2_miss_en2</td>
<td>Enable Level 1 instruction cache to load missed data</td>
</tr>
<tr>
<td>L2_data_miss1[511:0], L2_data_miss2[511:0]</td>
<td>Missed data written to the Level 1 instruction cache</td>
</tr>
<tr>
<td>addr_in1[31:0], addr_in2[31:0]</td>
<td>Physical address input from instruction TLB</td>
</tr>
<tr>
<td>clock</td>
<td>Enables the cache to find the requested data by CPU when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears contents of the cache when asserted high</td>
</tr>
<tr>
<td>read1, read2</td>
<td>Enables cache to read address from instruction TLB</td>
</tr>
<tr>
<td>inst_out1[31:0], inst_out2[31:0]</td>
<td>Generates the requested output data to CPU</td>
</tr>
<tr>
<td>addr_out1[31:0], addr_out2[31:0]</td>
<td>Miss address request sent to the Level 2 unified cache</td>
</tr>
<tr>
<td>hit_1, hit_2</td>
<td>Enabled during a cache hit</td>
</tr>
<tr>
<td>miss_1, miss_2</td>
<td>Enabled during a cache miss</td>
</tr>
<tr>
<td>en_out1, en_out2</td>
<td>Control signal sent to the Level 2 unified cache during a Level 1 instruction cache miss</td>
</tr>
</tbody>
</table>

### TABLE 9.5: Functional Pin Description of Level 1 Instruction Cache

#### 9.6 Level 1 Data Cache

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2_addr_miss1[31:0], L2_addr_miss2[31:0]</td>
<td>Address of the missed data written to the Level 1 data cache</td>
</tr>
<tr>
<td>L2_miss_en1, L2_miss_en2</td>
<td>Enable Level 1 data cache to load missed data</td>
</tr>
<tr>
<td>L2_data_miss1[511:0], L2_data_miss2[511:0]</td>
<td>Missed data sent from Level 2 unified cache to the Level 1 data cache</td>
</tr>
<tr>
<td>readaddr_in1[31:0], readaddr_in2[31:0]</td>
<td>Physical address input from data TLB</td>
</tr>
<tr>
<td>I/O Ports</td>
<td>Function</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>writeaddr_in1[31:0], writeaddr_in2[31:0]</td>
<td>Physical address where data needs to be written by the CPU buffer</td>
</tr>
<tr>
<td>writedata_in1[511:0], writedata_in2[511:0]</td>
<td>Data to be written to the Data cache by the CPU buffer</td>
</tr>
<tr>
<td>clock</td>
<td>Enables the cache to find the data requested by CPU when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears contents of the cache when asserted high</td>
</tr>
<tr>
<td>read1, read2</td>
<td>Enables cache to read address from data TLB</td>
</tr>
<tr>
<td>write1, write2</td>
<td>Enables cache to write data from CPU to the data TLB locations</td>
</tr>
<tr>
<td>cpu_data_out1[31:0], cpu_data_out2[31:0]</td>
<td>Generates the requested output data to CPU</td>
</tr>
<tr>
<td>missaddr_out1[31:0], missaddr_out2[31:0]</td>
<td>Miss address request sent to the Level 2 unified cache</td>
</tr>
<tr>
<td>hit_1, hit_2</td>
<td>Enabled during a cache hit</td>
</tr>
<tr>
<td>miss_1, miss_2</td>
<td>Enabled during a cache miss</td>
</tr>
<tr>
<td>buf_en1_out, buf_en2_out</td>
<td>Enables write buffer to receive dirty blocks from the data cache</td>
</tr>
<tr>
<td>buf_addr1_out[31:0], buf_addr2_out[31:0]</td>
<td>Addresses of the dirty blocks from the cache are written to the write buffer</td>
</tr>
<tr>
<td>buf_data1_out[511:0], buf_data2_out[511:0]</td>
<td>Dirty blocks of data are sent to write buffer from the data cache</td>
</tr>
<tr>
<td>en_read1_L2, en_read2_L2</td>
<td>Control signal sent to the Level 2 unified cache during a Level 1 data cache read miss</td>
</tr>
<tr>
<td>en_write1_L2, en_write2_L2</td>
<td>Control signal sent to the Level 2 unified cache when writing data to dirty blocks in data cache</td>
</tr>
<tr>
<td>l2_data1[511:0], l2_data2[511:0]</td>
<td>Dirty data sent to the Level 2 unified cache to update its contents</td>
</tr>
<tr>
<td>l2_addr1[31:0], l2_addr2[31:0]</td>
<td>Addresses of the dirty blocks sent from the data cache to the Level 2 unified cache</td>
</tr>
</tbody>
</table>

**TABLE 9.6: Functional Pin Description of Level 1 Data Cache**

### 9.7 Level 2 Unified Cache

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>inst_addr1[31:0], inst_addr2[31:0]</td>
<td>Miss instruction request sent by the Level 1 instruction cache to Level 2 unified cache</td>
</tr>
<tr>
<td>en_instaddr1, en_instaddr2</td>
<td>Enabled when a miss instruction request is sent by the Level 1 instruction cache to Level 2 unified cache</td>
</tr>
<tr>
<td>data_addr1[31:0], data_addr2[31:0]</td>
<td>Miss data request sent by the Level 1 data cache to Level 2 unified cache</td>
</tr>
<tr>
<td>en_dataaddr1, en_dataaddr2</td>
<td>Enabled when a miss data request is sent by the Level 1 data cache to Level 2 unified cache</td>
</tr>
<tr>
<td>mem_instaddr_miss1[31:0], mem_instaddr_miss2[31:0]</td>
<td>Missed data address and instruction address sent by the main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>Signal</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>mem_dataaddr_miss1[31:0]</td>
<td>Missed data and instructions sent by the main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>mem_dataaddr_miss2[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_inst1[511:0], mem_inst2[511:0]</td>
<td>Missed data and instructions sent by the main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>mem_data1[511:0], mem_data2[511:0]</td>
<td></td>
</tr>
<tr>
<td>mem_inst_en1, mem_inst_en2, mem_data_en1, mem_data_en2</td>
<td>Enabled when the requested missed data or instructions are sent by main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>l1_addr1[31:0], l1_addr2[31:0]</td>
<td>Dirty block address sent from data cache to Level 2 unified cache</td>
</tr>
<tr>
<td>l1_data1[511:0], l1_data2[511:0]</td>
<td>Dirty block data sent from data cache to Level 2 unified cache</td>
</tr>
<tr>
<td>en_L1_data1, en_L1_data2</td>
<td>Enabled when dirty block data is sent from Level 1 data cache to Level 2 unified cache</td>
</tr>
<tr>
<td>Clock</td>
<td>Enables the unified cache to find the requested information by Level 1 data cache or Level 1 instruction cache when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears contents of the Level 2 unified cache when asserted high</td>
</tr>
<tr>
<td>hit_1, hit_2, hit_3, hit_4</td>
<td>Enabled when a hit occurs in the Level 2 unified cache</td>
</tr>
<tr>
<td>miss_1, miss_2, miss_3, miss_4</td>
<td>Enabled when a miss occurs in the Level 2 unified cache</td>
</tr>
<tr>
<td>inst_addr1_out[31:0], inst_addr2_out[31:0], data_addr1_out[31:0], data_addr2_out[31:0]</td>
<td>Miss data and instruction requests sent by the Level 2 unified cache to main memory</td>
</tr>
<tr>
<td>en1, en2, en3, en4</td>
<td>Enabled when miss instruction and data requests sent by the Level 2 unified cache to main memory</td>
</tr>
<tr>
<td>inst_addr1_hit1[31:0], inst_addr2_hit2[31:0]</td>
<td>Address of the hit instruction sent by Level 2 unified cache to Level 1 instruction cache</td>
</tr>
<tr>
<td>data_addr1_hit1[31:0], data_addr2_hit2[31:0]</td>
<td>Address of the hit data sent by Level 2 unified cache to Level 1 data cache</td>
</tr>
<tr>
<td>inst_out1[511:0], inst_out2[511:0]</td>
<td>Hit instruction sent by Level 2 unified cache to Level 1 instruction cache</td>
</tr>
<tr>
<td>data_out1[511:0], data_out2[511:0]</td>
<td>Requested hit data sent by Level 2 unified cache to Level 1 data cache</td>
</tr>
<tr>
<td>en_inst_out1, en_inst_out2</td>
<td>Enabled when hit instruction is sent by the Level 2 unified cache to Level 1 instruction cache</td>
</tr>
<tr>
<td>en_data_out1, en_data_out2</td>
<td>Enabled when hit data is sent by the Level 2 unified cache to Level 1 data cache</td>
</tr>
<tr>
<td>cpu_inst_out1[31:0], cpu_inst_out2[31:0], cpu_data_out1[31:0], cpu_data_out2[31:0]</td>
<td>Generates the requested instruction and data outputs to the CPU</td>
</tr>
<tr>
<td>cpu_inst_en1, cpu_inst_en2, cpu_data_en1, cpu_data_en2</td>
<td>Enabled when generating the CPU requested instruction and data information</td>
</tr>
<tr>
<td>buf_en1_out, buf_en2_out</td>
<td>Enables write buffer to receive dirty blocks from the Level 2 unified cache</td>
</tr>
</tbody>
</table>
### TABLE 9.7: Functional Pin Description of a Level 2 Unified Cache

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>buf_addr1_out[31:0] , buf_addr2_out[31:0]</td>
<td>Addresses of the dirty blocks from the Level 2 unified cache are written to the write buffer</td>
</tr>
<tr>
<td>buf_data1_out[511:0] , buf_data2_out[511:0]</td>
<td>Dirty blocks of data from Level 2 unified cache are sent to the write buffer</td>
</tr>
</tbody>
</table>

#### 9.8 Main Memory

<table>
<thead>
<tr>
<th>I/O Ports</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>tlb_mem_inst1_en1 , tlb_mem_inst2_en2</td>
<td>Enables virtual address requests sent by the instruction TLB to the main memory</td>
</tr>
<tr>
<td>tlb_mem_data1_en1 , tlb_mem_data2_en2</td>
<td>Enables virtual address requests sent by the data TLB to the main memory</td>
</tr>
<tr>
<td>tlb_mem_write1_en1 , tlb_mem_write2_en2</td>
<td>Enables virtual address requests sent by the data TLB to the main memory</td>
</tr>
<tr>
<td>tlb_mem_inst1_viraddr1[31:0] , tlb_mem_inst2_viraddr2[31:0]</td>
<td>Virtual address requests sent by the instruction TLB to the main memory</td>
</tr>
<tr>
<td>tlb_mem_data1_viraddr1[31:0] , tlb_mem_data2_viraddr2[31:0]</td>
<td>Virtual address requests sent by the data TLB to the main memory</td>
</tr>
<tr>
<td>tlb_mem_write1_viraddr1[31:0] , tlb_mem_write2_viraddr2[31:0]</td>
<td>Virtual address requests sent by the data TLB to the main memory</td>
</tr>
<tr>
<td>inst_addr1_in[31:0] , inst_addr2_in[31:0]</td>
<td>Miss instruction address request sent by the Level 2 unified cache to main memory</td>
</tr>
<tr>
<td>en_instaddr1 , en_instaddr2</td>
<td>Enabled when miss instruction address request sent by the Level 2 cache to main memory</td>
</tr>
<tr>
<td>data_addr1_in[31:0] , data_addr2_in[31:0]</td>
<td>Miss data address request sent by the Level 2 unified cache to main memory</td>
</tr>
<tr>
<td>en_dataaddr1 , en_dataaddr2</td>
<td>Enabled when miss data address request sent by the Level 2 cache to main memory</td>
</tr>
<tr>
<td>L1_buf_en1_in , L1_buf_en2_in</td>
<td>Enabled when dirty block data is sent from write buffer to main memory</td>
</tr>
<tr>
<td>L1_buf_data1_in[511:0] , L1_buf_data2_in[511:0]</td>
<td>Dirty block data sent from write buffer to main memory</td>
</tr>
<tr>
<td>L1_buf_addr1_in[31:0] , L1_buf_addr2_in[31:0]</td>
<td>Dirty block address sent from write buffer to main memory</td>
</tr>
<tr>
<td>L2_buf_en1_in , L2_buf_en2_in</td>
<td>Enabled when dirty block data is sent from write buffer to main memory</td>
</tr>
<tr>
<td>L2_buf_data1_in[511:0] , L2_buf_data2_in[511:0]</td>
<td>Dirty block data sent from write buffer to main memory</td>
</tr>
<tr>
<td>L2_buf_addr1_in[31:0] , L2_buf_addr2_in[31:0]</td>
<td>Dirty block address sent from write buffer to main memory</td>
</tr>
<tr>
<td>Clock</td>
<td>Enables the main memory to find the information requested by Level 2 unified cache when asserted high</td>
</tr>
<tr>
<td>Reset</td>
<td>Clears contents of the main memory when asserted high</td>
</tr>
<tr>
<td>Pin Description</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>mem_L1_addr1_inst[31:0]</td>
<td>Address of the hit instruction is sent by the main memory to Level 1 instruction and data caches</td>
</tr>
<tr>
<td>mem_L1_addr2_inst[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L1_addr1_data[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L1_addr2_data[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L1_inst1_out[511:0]</td>
<td>Hit instruction data is sent by the main memory to Level 1 instruction and data caches</td>
</tr>
<tr>
<td>mem_L1_inst2_out[511:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L1_data1_out[511:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L1_data2_out[511:0]</td>
<td></td>
</tr>
<tr>
<td>en_L1_inst1 , en_L1_inst2</td>
<td>Enabled when the requested data is sent by the main memory to instruction and data caches</td>
</tr>
<tr>
<td>en_L1_data1 , en_L1_data2</td>
<td></td>
</tr>
<tr>
<td>mem_L2_addr1_inst[31:0]</td>
<td>Address of the hit instruction is sent by the main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>mem_L2_addr2_inst[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L2_addr1_data[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L2_addr2_data[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L2_inst1_out[511:0]</td>
<td>Hit instruction data is sent by the main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>mem_L2_inst2_out[511:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L2_data1_out[511:0]</td>
<td></td>
</tr>
<tr>
<td>mem_L2_data2_out[511:0]</td>
<td></td>
</tr>
<tr>
<td>en_L2_inst1 , en_L2_inst2</td>
<td>Enabled when the requested data is sent by the main memory to Level 2 unified cache</td>
</tr>
<tr>
<td>en_L2_data1 , en_L2_data2</td>
<td></td>
</tr>
<tr>
<td>mem_tlb_inst1_en1 , mem_tlb_inst2_en2</td>
<td>Enabled when the virtual address requested by the instruction TLB is sent by the main memory</td>
</tr>
<tr>
<td>mem_tlb_data1_en1 , mem_tlb_data2_en2</td>
<td>Enabled when the virtual address requested by the data TLB is sent by the main memory</td>
</tr>
<tr>
<td>mem_tlb_write1_en1 , mem_tlb_write2_en2</td>
<td></td>
</tr>
<tr>
<td>mem_tlb_inst1_viraddr[31:0]</td>
<td>Virtual address requested by the instruction TLB is sent by the main memory</td>
</tr>
<tr>
<td>mem_tlb_inst2_viraddr[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_tlb_data1_viraddr[31:0]</td>
<td>Virtual address requested by the data TLB is sent by the main memory</td>
</tr>
<tr>
<td>mem_tlb_data2_viraddr[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_tlb_write1_viraddr[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_tlb_write2_viraddr[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_cpu_inst_out[31:0]</td>
<td>Instructions requested by the CPU is generated by the main memory</td>
</tr>
<tr>
<td>mem_cpu_data_out[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_cpu_inst_out[31:0]</td>
<td></td>
</tr>
<tr>
<td>mem_cpu_data_out[31:0]</td>
<td></td>
</tr>
<tr>
<td>inst_addr1_out[31:0]</td>
<td>Miss instruction address request is sent to the slower memory</td>
</tr>
<tr>
<td>inst_addr2_out[31:0]</td>
<td></td>
</tr>
<tr>
<td>data_addr1_out[31:0]</td>
<td>Miss data address request is sent to the slower memory</td>
</tr>
<tr>
<td>data_addr2_out[31:0]</td>
<td></td>
</tr>
<tr>
<td>inst_addr1_en1 , inst_addr2_en2</td>
<td>Enabled when there is instruction miss or data miss in main memory</td>
</tr>
<tr>
<td>data_addr1_en1 , data_addr2_en2</td>
<td></td>
</tr>
<tr>
<td>found1 , found2 , found3 , found4</td>
<td>Enabled when the data requested by the CPU is found in the main memory</td>
</tr>
</tbody>
</table>

**TABLE 9.8: Functional Pin Description of Main Memory**
Chapter 10 APPENDIX B

This section contains the functional simulation results of all the modules in the two level cache system.

10.1 CPU Buffer

FIGURE 10.1: Functional Simulation of CPU Buffer

10.2 Write Buffer

FIGURE 10.2: Functional Simulation of WRITE Buffer
10.3 Instruction Translation Look-Aside Buffer

FIGURE 10.3: Functional Simulation of Instruction Translation Look-Aside Buffer

10.4 Data Translation Look-Aside Buffer

FIGURE 10.4: Functional Simulation of Data Translation Look-Aside Buffer
10.5  Level 1 Instruction Cache

FIGURE 10.5: Functional Simulation of Level 1 Instruction Cache

10.6  Level 1 Data Cache

FIGURE 10.6: Functional Simulation of Level 1 Data Cache
10.7 Level 2 Unified Cache

FIGURE 10.7: Functional Simulation of Level 2 Unified Cache
10.8 Main Memory

FIGURE 10.8: Functional Simulation of Main Memory
Chapter 11 APPENDIX C

This section contains gate level netlists for all the modules of the two level cache system.

11.1 Schematic of CPU Buffer

FIGURE 11.1: Schematic of CPU Buffer

11.2 Schematic of Write Buffer

FIGURE 11.2: Schematic of Write Buffer
11.3 Schematic of Instruction Translation Look-Aside Buffer

FIGURE 11.3: Schematic of Instruction Translation Look-Aside Buffer

11.4 Schematic of Data Translation Look-Aside Buffer

FIGURE 11.4: Schematic of Data Translation Look-Aside Buffer

11.5 Schematic of Level 1 Instruction Cache

FIGURE 11.5: Schematic of Level 1 Instruction Cache

11.6 Schematic of Level 1 Data Cache

FIGURE 11.6: Schematic of Level 1 Data Cache
11.7  Schematic of Level 2 Unified Cache

FIGURE 11.7: Schematic of Level 2 Unified Cache

11.8  Schematic of Main Memory

FIGURE 11.8: Schematic of Main Memory
Chapter 12 APPENDIX D

This section contains layouts of all the modules of the two level cache system.

12.1 Layout of CPU Buffer

FIGURE 12.1: Layout of CPU Buffer
12.2 Layout of Write Buffer

FIGURE 12.2: Layout of Write Buffer
12.3 Layout of Instruction Translation Look-Aside Buffer

FIGURE 12.3: Layout of Instruction Translation Look-Aside Buffer
12.4 Layout of Data Translation Look-Aside Buffer

FIGURE 12.4: Layout of Data Translation Look-Aside Buffer
12.5 Layout of Level 1 Instruction Cache

FIGURE 12.5: Layout of Level 1 Instruction Cache
12.6 Layout of Level 1 Data Cache

FIGURE 12.6: Layout of Level 1 Data Cache
12.7 Layout of Level 2 Unified Cache

FIGURE 12.7: Layout of Level 2 Unified Cache
12.8 Layout of Main Memory

FIGURE 12.8: Layout of Main Memory