Copyright
by
Stephen Bijansky, Jr.
2008

The Dissertation Committee for Stephen Bijansky, Jr.
certifies that this is the approved version of the following dissertation:

TuneChip: Post-Silicon Tuning of Dual-Vdd Designs

Committee:

Adnan Aziz, Supervisor

Abhijit Choudhury

Anna Gál

Michael Orshansky

Nur Touba

TuneChip: Post-Silicon Tuning of Dual-Vdd Designs

by

Stephen Bijansky, Jr., B.S.; M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2008

Dedicated to my wife Devon,

my companions Gatsby and Sport,

and my advisors Margarida and Adnan.

Acknowledgments

I would first and foremost like to thank Margarida Jacome. I am grateful for all of the guidance, support, and encouragement that you have shown throughout the years. You were a constant source of ideas across a broad range of areas. Because of you, I have been able to expand my horizons and investigate new and exciting technologies. I hope that one day I am able to instill the same passion for learning and scholarship in future students as you have in me.

I would also like to thank Adnan Aziz for providing all of your assistance and feedback. Your help in focusing both my dissertation and my course of action has been invaluable. I hope that we are able to work together again in the future.

I would also like to thank David Nagel who first got me started in graduate research. I learned so much from my time with you, and I truly appreciate all of the individual attention that you provided. And to everyone else involved with the Parallel Data Lab, especially Garth Gibson and Greg Ganger, I thank you for providing such a tremendous environment and opportunites.

I also wish to thank David Mills for helping me with my first thesis. It was the beginning of a long journey, and I am grateful that it got off to a good start. And to all of my professors throughout the years, I have always appreciated being able to get a small classroom experience no matter the size of the university.

I am especially appreciative of all my horn professors throughout the years.

David Reif for teaching me the foundations of music, preparing me for the transition to college, and many exciting tennis matches. Cynthia Carr for expanding my musical abilities, our many duets, providing accompaniment, teaching us the delicate art of constructive feedback during horn ensemble, and introducing me to the joys of playing in an orchestra. Dennis Abelson for providing a welcome diversion to graduate school life and arranging for more opportunities to play with an ensemble. All of your lessons were highlights of my week. Individual instruction is marvelous, and I am grateful of all the time that I spent with each of you.

I would also like to acknowledge and thank the Semiconductor Research Corporation, the University of Texas at Austin College of Engineering, the anonymous donor who sponsored my Engineering Doctoral Fellowship, and the donor who sponsored my Carnegie Mellon Fellowship for their support of my graduate studies. Your support has allowed me to concentrate more on my studies than would have otherwise been possible.

I would like to thank my parents who have provided me with so many wonderful opportunties and support throughout my life.

And most of all, I would like to thank my wife, Devon, who has always supported me though our many years together. Returning to school has taken longer than I originally envisioned, and there have been many difficulties along the way. I could not have completed this without your support, encouragement, love, and especially editing.

God bless and thank you all.

TuneChip: Post-Silicon Tuning of Dual-Vdd Designs

Publication No. ____________________

Stephen Bijansky, Jr., Ph.D.
The University of Texas at Austin, 2008

Supervisor: Adnan Aziz

As process technologies continue their rapid advancement, transistor features are shrinking to almost unimaginable sizes. Some dimensions can be measured at the atomic level. One consequence of these smaller devices is that they have become more susceptible to deviations from nominal than previous process nodes. To illustrate, as few as one hundred atoms determine how much voltage is needed to turn a transistor on and off. With over two billion transistors on a single chip, it is easy to imagine how even the tiniest of variations can affect many transistors throughout the entire chip. To compensate for these deviations, chip designers add margin to their designs. Even more margin is then added for increased safety. All of this margin leads to chips that are slower than a nominal design would be. At the other end of the spectrum, these same deviations might result in chips that are faster than needed. However, faster is not always better, as these faster chips usually require more power. Even worse, these deviations sometimes produce chips that are both slower and use more power than a nominal design. TuneChip is designed to mitigate the effects of these process variations by speeding up areas of a chip that

need to run faster while at the same time reducing power in parts of a chip that are operating faster than needed.

TuneChip attacks the variation problem by changing the voltage on small areas of the chip in response to the type of variation for that particular area. Since voltage has a strong relationship to the speed of a chip, TuneChip can increase the speed of areas that need to go faster. At the same time, TuneChip can decrease the speed of other areas on the chip that are too fast. Even more important than speed for current designs, though, is power. Changing the voltage has a quadratic relationship with the amount of power consumed by that device. Specifically, a 10% reduction in supply voltage yields a 20% reduction in energy. Moreover, it is not only battery powered devices that benefit from reduced energy consumption; some high performance designs are limited by how much they can cool the chip. Cost-effective cooling technology is not scaling at anywhere near the same rate as transistor geometries. Reducing a chip's power consumption also reduces excess heat.

In order to selectively change the voltage of specific areas of the design, TuneChip starts by partitioning the chip into smaller blocks. A dual voltage design style with two voltage grids spans the entire chip. In order to best react to variations particular to an individual chip, each block is assigned a supply voltage only after manufacturing. First, the chip is tested at high voltage and high power in order to verify the correct functionality of that chip. If the chip passes its functionality testing, each individual block is tested to determine how fast it is operating. Blocks that need to run faster are configured to connect to the high supply voltage grid, and

blocks that are able to run slower are configured to connect to the low supply voltage grid. The configurable block supply voltage connection is accomplished with pmos pass transistors that act like switches. By having only one pmos pass transistor switch turned on at a time, each block has a choice of two supply voltages.

Table of Contents

Abstract **vii**

List of Tables **xiii**

List of Figures **xiv**

Glossary **xv**

Chapter 1. Introduction **1**

1.1 VLSI Design Cycle . . . 1

1.2 Process Variation . . . 3

1.3 Contributions . . . 5

Chapter 2. TuneLogic **8**

2.1 Dual Voltage Design . . . 9

2.1.1 Tunable Block . . . 10

2.1.2 32×32-bit multiplier . . . 11

2.1.3 Overhead of Power Transistor and Level Converter . . . 12

2.2 Block Characterization . . . 13

2.2.1 Methodology . . . 13

2.2.2 Results . . . 14

2.3 Voltage Assignment Algorithm . . . 16

2.4 Experiments . . . 18

2.4.1 Setup for Experiments . . . 18

2.4.2 Key Takeaways and Design Considerations . . . 19

2.5 Related Work . . . 23

2.6 Conclusion . . . 24

Chapter 3. TuneFPGA **26**
3.1 Dual Voltage CLB Design 27
3.1.1 Tunable CLB 27
3.1.2 Area Overhead 28
3.2 CLB Delay Statistics 29
3.3 Post-Manufacturing Delay Measurement 32
3.4 CAD Flow 33
3.5 Experiments 34
3.5.1 Setup for Experiments 35
3.5.2 Experimental Results 35
3.5.3 Guard Banding Voltage Selection 39
3.6 Related Work 40
3.7 Conclusion and Future Work 42

Chapter 4. TuneRAM **43**
4.1 Background: SRAM Cell Design 44
4.2 Adaptive SRAM 47
4.2.1 Memory Operation 48
4.2.2 Adaptive Voltage Supply 51
4.2.3 Voltage Assignment Algorithm 53
4.3 Experiments 56
4.3.1 Results 57
4.3.2 Key Takeaways 58
4.4 Related Work 59
4.5 Conclusion 61

Chapter 5. TuneInterconnect **62**
5.1 Introduction 62
5.2 Dual Voltage Repeater 62
5.3 Interconnect Characterization 63
5.3.1 Methodology 65
5.3.2 Results 66
5.4 Conclusion 70

Chapter 6. Conclusion **72**
6.1 Conclusion 72
6.2 Future Work 73

Appendices **74**

Appendix A. Voltage Assignment NP-Complete Proof **75**

Appendix B. Bitline Energy Derivation **78**

Bibliography **80**

Vita **90**

List of Tables

2.1 32-bit multiplier benchmark 20
2.2 10 instances of a 32-bit multiplier benchmark 21

3.1 TuneFPGA benchmarks 36
3.2 TuneFPGA benchmarks (cont.) 37

4.1 SRAM array energy 57

5.1 Interconnect delays 70

List of Figures

1.1 TuneChip design flow 6

2.1 TuneLogic dual-Vdd block 10
2.2 TuneLogic block with 16 full adder cells 11
2.3 Dual-Vdd delay distribution for a single block 14
2.4 Dual-Vdd energy distribution for a single block 15

3.1 TuneFPGA dual-Vdd CLB 28
3.2 Dual-Vdd delay distribution for a single CLB 30
3.3 Dual-Vdd power distribution for a single CLB 31

4.1 6T SRAM schematic 45
4.2 SRAM column schematic 47
4.3 Bitline timing diagram 48
4.4 RC circuit schematic 49
4.5 SRAM bitline voltage development 50
4.6 SRAM array 52
4.7 BIST block diagram 54

5.1 TuneInterconnect dual-Vdd block 63
5.2 3 segment PI model used for wire simulations 64
5.3 k segment wire with k repeaters 64
5.4 Delay for a 10 mm interconnect using various number of repeaters . 65
5.5 Delay distribution for a 10 mm interconnect 67
5.6 Energy distribution for a 10 mm interconnect 67
5.7 Delay distribution of a fixed voltage design versus TuneInterconnect 68

Glossary

6T SRAM	6-transistor static random access memory.
Addr	Address.
ALM	Adaptive logic module.
AVS	Adaptive voltage scaling.
BIST	Built-in self test.
BL	Bit line.
BLB	Bit line bar.
CAD	Computer aided design.
CLB	Configurable logic block.
CLK	Clock.
CMP	Chemical mechanical polishing.
Cox	Capacitance of the oxide layer.
CR	Cell ratio.
Demux	Demultiplexer.
FOM	Figure of merit.
FPGA	Field-programmable gate array.
IR	Current-resistive.
LER	Line edge roughness.
Lpd	Length of pull-down transistor.
LUT	Lookup table.
MEMS	Microelectromechanical systems.
Mux	Multiplexer.
NAND3X4	3-input NAND gate with strength 4$\times$ nominal.
NBTI	Negative bias temperature instability.
PD	Pull-down transistor.

PG	Pass gate.
PMIC	Power management IC.
PRECH	Pre-charge.
PU	Pull-up transistor.
PVT	Process, voltage, and temperature.
RC	Resistive-capacitive.
RD_EN	Read enable.
RDF	Random dopant fluctuation.
RTL	Register transfer level.
SA_EN	Sense amplifier enable.
SNM	Static noise margin.
SPICE	Simulation Program with Integrated Circuit Emphasis.
SRAM	Static random access memory.
Td	Time d.
TLB	Translation lookaside buffer.
μ_n	Electron mobility.
VAA	Voltage assignment algorithm.
VAP	Voltage assignment problem.
Vbl	Bitline development voltage.
Vdd	Voltage supply.
Vddmem	Voltage supply dedicated to the memory array.
Vddmin	Minimum allowed supply voltage.
Vddwl	Word line voltage supply.
VH	High value voltage supply (i.e., larger than VL).
VL	Low value voltage supply (i.e., less than VH).
VLSI	Very large scale integration.
Vth	Threshold voltage of a transistor.
WKP	Weighted-knapsack problem.
WL	Word line.
WNS	Worst negative slack.
Wpd	Width of pull-down transistor.
WR_EN	Write enable.

Chapter 1

Introduction

As process technologies advance, feature sizes continue to shrink. Because the devices are smaller, they have become more susceptible to deviations from nominal than previous process nodes. To compensate for these deviations, chip designers add margin to their designs. This margin leads to chips that are slower than a nominal design would be. At the other extreme, these same deviations might result in chips that are faster than needed, but these faster chips usually require more power. Even worse, these deviations sometimes produce chips that are both slower and use more power than a nominal design. TuneChip is designed to mitigate the effects of process variation by speeding up areas of a chip that need to run faster while also reducing power in parts of a chip that that are operating faster than needed.

1.1 VLSI Design Cycle

Before we discuss process variations in more depth, it is helpful to describe the VLSI design cycle [52] so that process variations can be put into proper context. Chips go through a number stages as a part of the VLSI design cycle. The high level phase starts with a *specification* that contains the high level requirements, such as frequency and power, that are necessary for a successful product. Next,

the *architecture* and *microarchitecture* are developed, giving a broad overview of the overall structure of the design. Examples of architectural choices include how many and what type of caches to use, how many functional units to include, and how to pipeline the design. This architectural design is used to develop the first estimates of the performance and power of the chip.

The implementation phase begins with *logic design*, which uses the architecture as a guide to create a register transfer level (RTL) description of the chip. The RTL is used to simulate the design in order to verify that the chip will function as intended. *Circuit design* implements the logic design using transistors and interconnects. Next, *physical design* converts the circuit design to a physical layout using shapes and layers. All throughout the logic design, circuit design, and physical design stages, the design is simulated with increasing levels of accuracy. Adjustments are made, and the design goes through many iterations to improve the performance, power, and yield of the chip.

After the implementation phase finalizes the physical design, the manufacturing phase begins by sending the chip to a foundry for *fabrication*. The foundry takes the physical design and creates photo-lithographic masks that are used in the various manufacturing steps. Once the fabrication process completes, the finished die is *packaged*. Finally, the packaged chip undergoes *testing* to determine if it meets the specification from the beginning of the design process.

1.2 Process Variation

Foundries provide device models that can be used to simulate the behavior of the chip. These simulations are used to predict how a design will operate after manufacturing. At the lowest level, designers can use the computer program SPICE for the most accurate level of detail. However, SPICE requires a large amount of computing resources, which makes it practical only for small blocks. The small block simulations can then be used to create higher order models that capture the functionality of the blocks, but these higher order models lack some of the details associated with SPICE simulation.

In order to guarantee that a design can be manufactured, the foundries also provide a set of layout rules to which the designs must adhere. For instance, a layout rule might dictate the distance between two adjacent transistors. In more complex technologies, another layout rule could specify the orientation of the transistors. Further, the layout rules could give the requirements for the amount of interconnect metal that is needed for each area of the chip.

Until recently, the device models and layout rules provided by the foundries allowed chip designers to accurately predict and simulate their designs before fabrication. Scaling process technologies, though, have contributed to an increase in complexity in the fabrication process. This increased complexity leads to fabricated devices with greater deviations from their nominal design values. Furthermore, the decrease in absolute device size leads to additional problems of their own. For example, it has been observed that the variation of threshold voltage is inversely proportional to the square root of the transistor area [55].

Borkar *et al.* [11] show in their work the impact of process, voltage, and temperature variations. For a batch of microprocessors all on the same wafer, variations in transistor channel length and threshold voltage contributed to a 30% difference in chip frequencies, while the standby leakage current varied by as much as $20\times$. In a recent dual core 65nm Xeon processor, leakage accounted for 30% of the total power [49]. The processing cores were responsible for 67% of that leakage power while 22% was attributed to the L3 cache.

Process variations can be broadly characterized as either die-to-die (inter-die) or within-die (intra-die) [58]. The within-die variations are then broken down into systematic or random variations. One random variation example is random dopant fluctuation (RDF) in the channel of a transistor [35]. A 70 nm transistor with a width of 140 nm has approximately 100 dopant atoms in the channel. Small changes in the number of dopant atoms can result in large threshold voltage changes. Other examples of random variations are gate line edge roughness (LER) [23] and local oxide thickness variations [7].

Visweswariah [60] points out that the interconnect is becoming an additional source of large process variation. For instance, chemical mechanical polishing (CMP) creates dishing and erosion that change the resistance of the interconnect by up to 20% [44]. Unlike transistor variability, which has some correlation of parameters between many of the transistors on a single die, the metal layer variations have little correlation between metal layers.

Some of the techniques to help alleviate the problems with process variations are multiple Vt transistors, power gating, state assignment, transistor body

bias, changing frequency, and changing supply voltage [42]. Even with these techniques, increasing the number of critical paths in a chip will increase the likelihood that one of the critical paths will be made slower because of process variations.

1.3 Contributions

As described in the previous section, designers have control over many aspects of the chip at various levels: architecture, microarchitecture, logic, and circuits. This dissertation is focused at the circuit design level, where the designer can control details such as transistor width, transistor length, voltage threshold, voltage supply, and well biasing. For instance, the designer can use transistors of different strengths to help meet delay constraints for critical path elements. For devices that are not on the critical path, the designer might choose to use high threshold voltage devices in order to reduce leakage power.

At a higher level, the chip architect might choose to have a range of supply voltages at which the chip can operate. By selecting a higher supply voltage, the chip will operate faster, but the chip will also use more power. Conversely, a lower supply voltage leads to a slower chip that uses less power. An aggressive technique to control power is to include dual supply voltages for either the entire chip or parts of the chip. Again, the circuit designer could choose to use the lower voltage supply for circuit paths that easily meet their delay constraints.

The TuneChip design flow is shown in Figure 1.1. The new steps specific to TuneChip appear below the dashed line. Since these new steps take place after manufacturing, designers will still be able to use all of the current tools and techniques

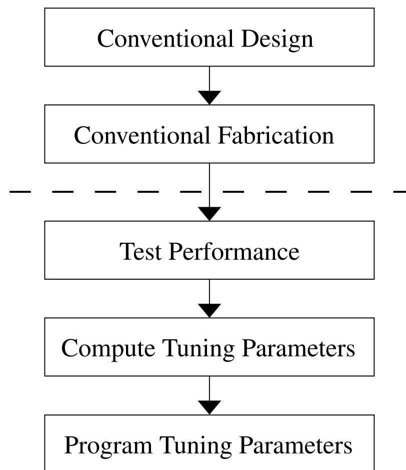

Figure 1.1: TuneChip design flow

to which they are accustomed.

TuneChip proposes a dual voltage design style with two voltage grids spanning across the entire chip. Each chip is partitioned into blocks, each of which is connected to the appropriate supply grid through a pmos pass transistor. By having only one pmos pass transistor turned on at a time, each block can have one of two supply voltage choices. In order to best react to individual variations particular to an individual chip, the voltage supply selection is performed after the chip has been manufactured.

This dissertation focuses on tuning using a dual voltage supply because changes in the supply voltage have such a large impact on both the speed and power consumption of the chip. Recall that by selecting a higher supply voltage, the chip will operate faster but will use more power; a lower supply voltage leads to a slower chip that uses less power. The designer can use the lower voltage supply for logic on

paths that easily meet their delay constraints. These non-critical paths can therefore run slower and use less power.

The contributions of this dissertation are organized as follows:

- Chapter 2 describes the power saving and yield improvements for post-silicon tuning in general logic gates.
- Chapter 3 shows that post-silicon tuning of FPGAs is beneficial in terms of power and yield.
- Chapter 4 applies tuning to improve power consumption in dense on-chip memories.
- Chapter 5 discusses interconnect variability and tuning interconnect repeaters.
- Chapter 6 gives key take-aways from this work and discusses areas for future research.

Prior work is presented in each individual chapter.

Chapter 2

TuneLogic

This chapter presents a CAD methodology and a quantitative study, using realistic data on modern process technologies, of the benefits of post-manufacturing tuning in terms of yield and power for dual-Vdd logic and its cost in terms of area, delay, and power. We have found that for a custom 32×32-bit pipelined multiplier in a representative modern process, post-manufacturing tuning increases yield and decreases power compared with a conventional fixed-Vdd design that selects the voltage supply based on nominal (i.e., pre-manufacturing) device parameters. This multiplier achieves a 100% yield with TuneLogic using 23.6 pJ of energy, while a fixed 1.0 V design uses 34.6 pJ of energy. A preliminary version of this work was presented at ACISC [9], and the complete version of this work will be presented at ISQED [10].

A detailed comparison with previous work is given in Section 2.5. Broadly speaking, our approach differs from previous work in that we tune individual designs after fabrication. As a result, this tunable design is able to respond to actual chip variations instead of estimated values. After the chip is fabricated, at-speed testing (e.g., via BIST [56]) determines which tunable blocks will use a high voltage supply and which tunable blocks will use a low voltage supply. Since this tuning

and configuration takes place after manufacturing, designers will still be able to use all of the current tools and techniques to which they are accustomed.

2.1 Dual Voltage Design

We focus on tuning using a dual voltage supply because changes in the supply voltage have such a large impact on both the speed and power consumption of the chip. Recall that, to a first approximation, gate delay is inversely proportional to supply voltage, whereas switching energy increases quadratically with supply voltage. In a traditional dual-Vdd design, the designer selects the lower voltage supply for logic on paths that easily meet their delay constraints. These non-critical paths can therefore run slower and use less power.

The dual voltage design style uses two supply voltage grids. Logic is partitioned into tunable blocks of gates in close physical proximity. Each tunable block of gates is connected to the appropriate supply grid through a pmos pass transistor. By having only one pmos pass transistor turned on at a time, each tunable block can have one of two supply voltage choices.

When multiple voltage supplies exist on the chip, level converters are required at all junctions between low voltage outputs and high voltage inputs. In its simplest form, a level converter is a buffer with additional transistors to prevent short circuit current. In our designs, level converters will be incorporated into the flip-flops; we design in a way to ensure that a low-Vdd gate never drives a high-Vdd gate.

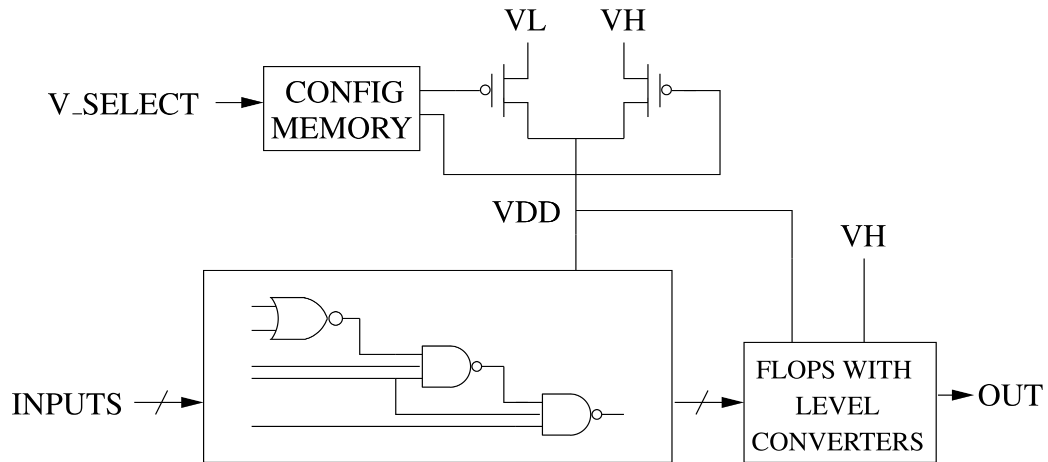

Figure 2.1: Tunable dual-Vdd block with configurable voltage supply PMOS transistors. The logic gates are connected to the local Vdd. VH is connected to the high supply voltage grid, and VL is connect to the low supply voltage grid. The flops include level converters. Therefore, the block output is always high voltage.

2.1.1 Tunable Block

It is clearly impossible to set the supply voltage on a per-gate basis—the overhead of power transistors and their associated configuration logic would be enormous. In our design style, logic gates are partitioned into *tunable blocks*. The generic tunable block is shown in Figure 2.1. This tunable block uses normal logic gates and flops with level converters. All of the gates are connected to a common Vdd that can be switched between high supply voltage and low supply voltage depending on the voltage select configuration memory. The flops with level converters always have a high voltage output, and they accept either low voltage or high voltage inputs. The common Vdd supply voltage configuration is stored in memory with complementary outputs so that only one pmos pass transistor can be turned on

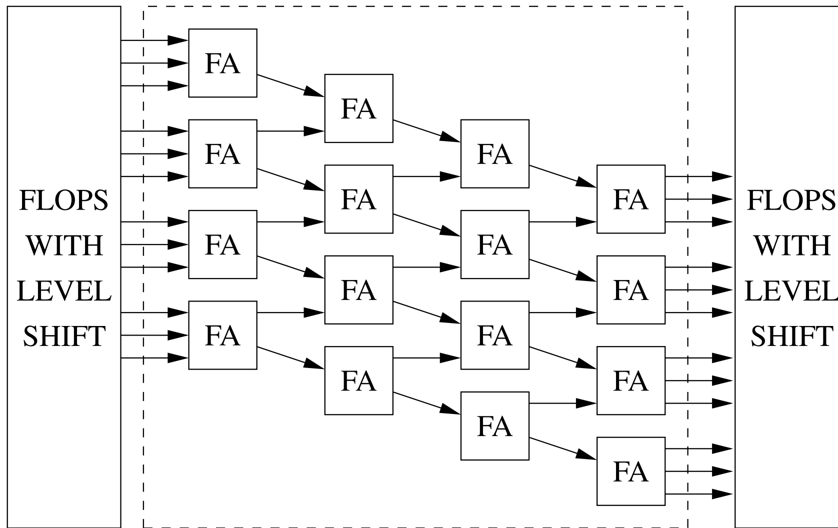

Figure 2.2: Tunable block logic gates using 16 full adder cells. All full adder cells within the dashed box are connected to a local Vdd supply. Only edges on the critical path are shown.

at a time. The supply voltage configuration is programmed before normal operation of the tunable block begins.

2.1.2 32×32-bit multiplier

We designed a 32×32 bit pipelined array multiplier to study the advantages of tunable blocks. Modern graphics processors can have up to 320 multipliers on a single chip [6], making the pipelined multiplier a suitable case study for tunable design. The multiplier was constructed using carry-save full adders (FA) followed by a ripple carry vector merging step, similar to that described in [45]. The multiplier was designed at the schematic level using Cadence and converted to Verilog for functionality testing.

The tunable block for our multiplier is shown in Figure 2.2. The design was partitioned into tunable blocks that contained 16 FA cells—4 deep and 4 wide. The partition for the tunable block was chosen in order to provide reasonable block size granularity for tuning while minimizing the area overhead of the pass transistor. The pass transistor area overhead of a vertical partition of 16 FA cells was prohibitively large due to the fact that there could be 16 FA cells switching simultaneously. In our partition, there are a maximum of 4 FA cells switching simultaneously during the clock cycle within the tunable block. This partition method allows pass transistor area overhead to be kept small while maintaining reasonable tunable block size, as the pass transistor only has to support the current for 4 FA cells simultaneously switching. The logic depth of one pipeline stage was one tunable block.

2.1.3 Overhead of Power Transistor and Level Converter

In order to size the pmos pass transistor for our tunable block, a SPICE simulation sweep was performed. The width was varied from $10\times$ minimum size to $300\times$ minimum size. The chosen pass transistor size of $110\times$ minimum size resulted in a 3.4% increase in tunable block delay compared to having no pass transistor. After a suitably fingered layout, the area of one pass transistor was $1.75\times$ the size of a NAND3X4 gate, which is a relatively small standard cell gate. In situations in which total area and power are more important than delay, a smaller pass transistor can be used. Future technologies, such as MEMS [17], could allow an even smaller pass transistor. MEMS switches are slow but high-quality; we do not need to switch the power transistor after the voltage assignment has been made,

making MEMS ideal for our application.

The STR6 level converter [29] is composed of a buffer with 6 additional minimum-sized devices. Since flops already have an internal buffer structure, the STR6 level converter can be incorporated with minimal area overhead. Furthermore, a well-designed level converter flop would only have a level converter in the low-Vdd supply path. If a block is assigned to use the high-Vdd supply, there would be no level converter delay overhead. For the low-Vdd supply path, the preliminary estimate is that there will be a 30 ps setup delay associated with the level converter.

2.2 Block Characterization

2.2.1 Methodology

We performed Monte Carlo simulations on the multiplier tunable block. The 65 nm Berkeley PTM model [12] was used for the SPICE transistor models. For each transistor in the tunable block other than the large power transistor, both the transistor length and the threshold voltage were modeled as uncorrelated Gaussian distributions with the $3\sigma/\mu$ variation set at 20% [13, 53]. The high voltage supply was set to 1.2 V and the low voltage supply was set to 0.8 V. The simulation temperature was 85 °C.

We measured the performance of our tunable block in 10,000 Monte Carlo simulations using HSPICE. The tunable block was simulated for the worse-case delay, which was measured from the 50% crossing of the primary input signal to the 50% crossing of the input to the flop.

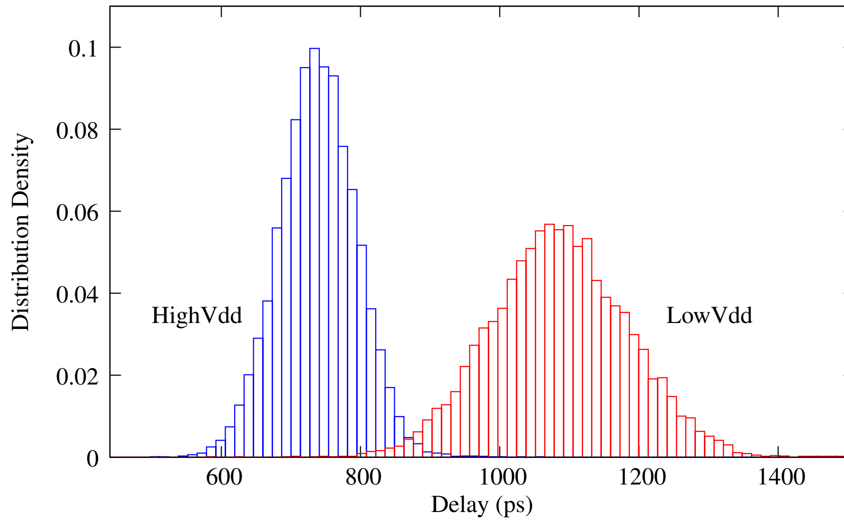

Figure 2.3: Dual-Vdd delay distribution for a single block

2.2.2 Results

The worst-case output transition delay distribution is shown for all 10,000 trials in Figure 2.3. In this figure, the bars on the left represent the delay when the tunable block of gates is connected to the high voltage supply through the pmos pass transistor. Then the tunable block of gates is disconnected from the high voltage supply and connected to the low voltage supply through the other pmos pass transistor. The bars on the right represent the delay when the tunable block of gates are connected to the low voltage supply. In order to present an equal comparison, the number of bars on the left is equal to the number of bars on the right. This graph shows that there is a large spread of delay values for this tunable block when intra-chip variation is taken into account. Moreover, the low-Vdd distribution has a larger amount of variation—caused by the fact that V_{th} is a larger fraction of the supply voltage, and hence V_{th} variations have a larger impact.

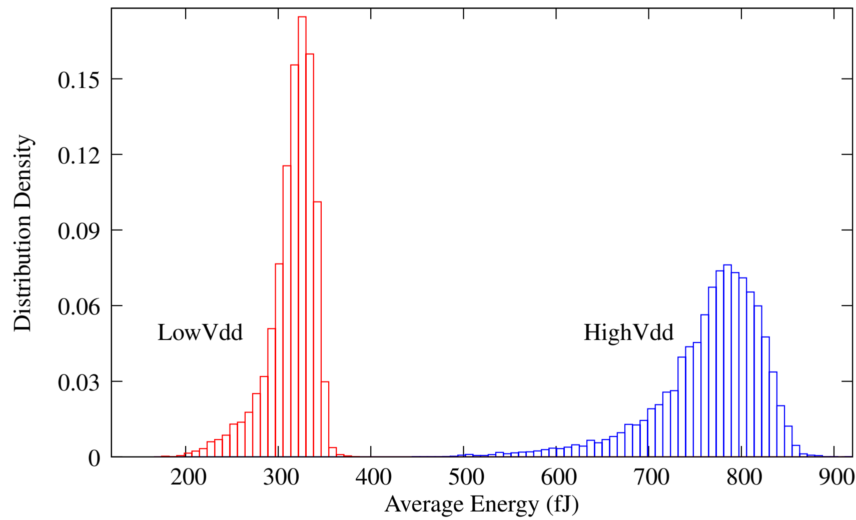

Figure 2.4: Dual-Vdd energy distribution for a single block

The switching energy was measured for 10 ns starting when the primary input signals change. All primary outputs and internal nodes reach their final values long before the end of the 10 ns energy measurement period. During the following 10 ns, leakage energy is measured for an additional 10 ns period. Dynamic energy is calculated by subtracting the leakage energy from switching energy. The energy values are measured for both the tunable block gates and the power transistor connection to the voltage grid. Figure 2.4 show the dynamic energy values; the energy spreads are similar to the delay spreads.

Taken together, these graphs show that there is potential to fine tune delay while simultaneously reducing dynamic power. The following summarizes the potential energy savings for a single tunable block of 16 full adder cells:

1. The delay increases from a mean of 738 ps using high-Vdd to 1036 ps

using low-Vdd.

2. The energy decreases from a mean of 765 fJ using high-Vdd to 314 fJ using low-Vdd.

The ratio of the energy for high-Vdd to low-Vdd is 2.4, which is close to the 2.25 predicted from the voltage scaling rules, confirming that the energy is scaling proportionally to Vdd^2. If the designer is able to decrease the speed on non-critical path tunable blocks by switching those tunable blocks to a low voltage supply, energy can be reduced. Even with tunable blocks on the critical path, if there is enough slack, some of those tunable blocks could use a low supply voltage while the other tunable blocks use a high supply voltage. Depending on the number of tunable blocks that can be assigned to use a low supply voltage, there is potential to reduce total energy by 50%.

2.3 Voltage Assignment Algorithm

After the die is manufactured, voltage assignment is performed. First, all of the tunable blocks are assigned to use a high-Vdd supply. Then, an at-speed test is performed to determine if the chip meets the delay target. If the chip fails this high-Vdd at-speed test, the chip is marked as failing. Otherwise, an iterative assignment algorithm selects tunable blocks to power down while checking that the delay target is still being met.

Selecting the smallest set of tunable blocks to assign to the high-Vdd supply is NP-complete (see Appendix A). Therefore, we devised a simple greedy algorithm

Algorithm 2.1 Voltage Assignment Algorithm

```
 1: Assign all blocks to use the high supply voltage
 2: if chip fails at-speed testing then
 3:    chip is marked as failing and is discarded
 4: end if
 5: Assign criticality rating to each block using nominal transistor parameters
 6: BlockQueue ⇐ blocks sorted by criticality rating
 7: while !BlockQueue.empty() do
 8:    TestBlock ⇐ BlockQueue.pop()
 9:    Configure TestBlock to use the low supply voltage
10:    if chip passes at-speed testing then
11:       Permanently assign TestBlock to use the low supply voltage
12:    else
13:       Permanently assign TestBlock to use the high supply voltage
14:    end if
15: end while
```

for voltage assignment, shown in Algorithm 2.1, that uses at-speed testing. Using nominal pre-manufacturing transistor parameters, each tunable block is given a criticality rating based on the amount of timing slack for that tunable block. A tunable block with a large amount of slack is a good candidate for being powered down. Consequently, the voltage assignment algorithm begins by selecting the tunable block with the most timing slack. The selected tunable block is assigned to use a low-Vdd supply. Next, the at-speed test is performed to determine if the design still meets the delay target. If the at-speed test passes, the selected tunable block is permanently assigned to use the low-Vdd supply. Otherwise, if the at-speed test fails, the selected tunable block is permanently assigned to use the high-Vdd supply. This

process is repeated until all of the tunable blocks have been assigned to use either a high-Vdd or a low-Vdd voltage supply.

2.4 Experiments

The multiplier was instantiated on 10,000 chips using the tunable block data from Section 2.2. Each tunable block has independent variations. The testing algorithm of Section 2.3 was simulated for each chip for a range of delay targets. The delay, energy, and percentage of tunable blocks connected to the low-Vdd supply was recorded for each chip.

2.4.1 Setup for Experiments

Results are summarized in Tables 2.1 and 2.2. Details about the experimental setup are as follows:

- The TuneLogic approach was compared to a design that uses a single fixed voltage supply, without any power transistors.
- Yield, energy, and the percentage of tunable blocks connected to the low-Vdd supply are measured for each target delay. The energy is reported only for designs that meet the target delay.
- The range of target delays was chosen such that, for Table 2.1, TuneLogic has a yield of 90% at the fastest target delay and the fixed 0.9 V has a yield of 95% at the slowest target delay. A target delay based on a fixed 0.8 V with a yield of 95% would have resulted in a target delay that was 1489 ps, which was too slow for this range of comparisons.

- Table 2.1 is a single 32-bit multiplier consisting of 64 tunable blocks, each of which is 16 FAs. Table 2.2 is a design consisting of ten 32-bit multipliers, i.e., a total of 640 tunable blocks.

2.4.2 Key Takeaways and Design Considerations

TuneLogic's flexibility allows significant yield and power advantages compared to fixed voltage over a large number of delay targets, as shown in Table 2.1.

- For a 1200 ps target delay, TuneLogic used 15% less energy and had 5% higher yield than the low power fixed 0.9 V design.
- Even at the fastest target delay of 925 ps, TuneLogic had 4% higher yield than the fixed 1.1 V. For this delay, TuneLogic also used 10% less energy than a fixed voltage of 1.2 V even though only 5% of the TuneLogic blocks were low-Vdd. In general, the TuneLogic block at 1.2 V uses 7% less energy than fixed at 1.2 V because the TuneLogic power transistor reduces the amount of short circuit current compared to the same block without any power transistor.
- When more tunable blocks are considered, Table 2.2 shows that variations have an even larger impact on yield. For a 1200 ps target delay in Table 2.1, fixed 0.9 V has yield of 95%, but in Table 2.2, the yield is 60%.
- It is important to note that we chose relatively conservative 20% $3\sigma/\mu$ variation parameters. Other work has predicted variation parameters of 30% [43] or even higher [32].
- In a traditional 65 nm design, the FO4 inverter delay going from the TT

Table 2.1: 32-bit multiplier using 2 different voltage assignments. The multiplier was instantiated on 10,000 chips, each of which have independent variations. TuneLogic: Post-manufacturing voltage assignment with High-Vdd 1.2 V and Low-Vdd 0.8 V. Fixed: Each chip uses a fixed voltage supply.

Target Delay (ps)	Yield (%)						Average Low-Vdd Blocks (%)
		Fixed Voltage (No Power Transistor)					
	TuneLogic	1.2 V	1.1 V	1.0 V	0.9 V	0.8 V	TuneLogic
925	90	96	86	10	0	0	5
970	96	99	95	65	0	0	12
1016	99	100	99	89	3	0	23
1062	100	100	100	95	45	0	40
1108	100	100	100	99	80	0	59
1154	100	100	100	100	89	0.6	75
1200	100	100	100	100	95	16	87

(a) Yield

Target Delay (ps)	Average Energy (pJ)						Average Low-Vdd Blocks (%)
		Fixed Voltage (No Power Transistor)					
	TuneLogic	1.2 V	1.1 V	1.0 V	0.9 V	0.8 V	TuneLogic
925	47.6	52.6	43.0	34.6	n/a	n/a	5
970	45.7	52.6	43.0	34.6	n/a	n/a	12
1016	42.4	52.6	43.0	34.6	27.3	n/a	23
1062	37.4	52.6	43.0	34.6	27.3	n/a	40
1108	31.9	52.6	43.0	34.6	27.3	n/a	59
1154	27.2	52.6	43.0	34.6	27.3	20.9	75
1200	23.8	52.6	43.0	34.6	27.3	21.0	87

(b) Energy

Table 2.2: 10 instances of a 32-bit multiplier using 2 different voltage assignments

Target Delay (ps)	Yield (%)						Average Low-Vdd Blocks (%)
	TuneLogic	Fixed Voltage (No Power Transistor)					TuneLogic
		1.2 V	1.1 V	1.0 V	0.9 V	0.8 V	
925	36	68	22	0	0	0	4
970	69	88	60	2	0	0	11
1016	88	100	88	34	0	0	23
1062	100	100	100	60	0.1	0	40
1108	100	100	100	88	11	0	59
1154	100	100	100	100	31	0	75
1200	100	100	100	100	60	0	87

(a) Yield

Target Delay (ps)	Average Energy (pJ)						Average Low-Vdd Blocks (%)
	TuneLogic	Fixed Voltage (No Power Transistor)					TuneLogic
		1.2 V	1.1 V	1.0 V	0.9 V	0.8 V	
925	476.4	526.0	430.4	n/a	n/a	n/a	4
970	457.0	526.0	430.4	346.1	n/a	n/a	11
1016	423.6	526.0	430.4	346.2	n/a	n/a	23
1062	374.4	526.0	430.4	346.2	272.7	n/a	40
1108	319.4	526.0	430.4	346.2	272.9	n/a	59
1154	272.3	526.0	430.4	346.2	272.9	n/a	75
1200	237.8	526.0	430.4	346.2	272.9	n/a	87

(b) Energy

corner to the SS corner can vary by up to 40% [61]. Since TuneLogic uses a post-manufacturing model, it would be able to provide the same yield and power advantages using corner case variation as our simulated parametric variations.

These experiments also allow some insights into the following important design considerations:

- For designs with fast target delays, the power transistor should be sized larger. In Table 2.1, TuneLogic's lower yield than fixed 1.2 V is a result of the 3.5% delay overhead associated with the $110\times$ power transistor. If there was a $590\times$ power transistor, TuneLogic would have less than a 1% delay overhead, which would result in comparable yields with fixed 1.2 V. At the same time, TuneLogic would still be able to offer lower power by assigning some of those blocks to use the low-Vdd power supply.
- Across all target delays, there could be additional energy savings by using a voltage supply that is more tailored to each target. For the fast target delay, using a low-Vdd supply of 1.0 V would allow TuneLogic to assign more tunable blocks to use the low-Vdd supply. For the slower target delays, using a high-Vdd supply of 1.0 V would save power for those tunable blocks assigned to use the high-Vdd supply.
- For slower target delays, a large percentage of tunable blocks were able to use the low voltage supply. This means that a tunable block size of 16 full adder cells was a reasonable choice. Furthermore, a finer-grained tunable block size would probably only result in small incremental improvements.

Overall, TuneLogic is effective in both increasing yield and reducing energy.

2.5 Related Work

Liang *et al.* [32] present an assignable dual-Vdd for a multiplier split into 6 stages. Unlike TuneLogic, Liang uses a brute force approach to iteratively test the power consumed by all 2^6 possible voltage assignments to find the best configuration. If a design has a large number of tunable blocks, this brute force approach would require an exponential amount of time to measure changes in power for every permutation. Another limitation of Liang's work is that it does not discuss the size of power transistor, the approach taken to size the power transistor, or delay and area overhead of the power transistor. Also, there is no discussion as to the best tunable block size. For instance, is there a valid design constraint for using only 6 programmable tunable blocks, or was 6 chosen to reduce the amount of time spent during power testing?

Agarwal *et al.* [2] use a programmable dual-Vdd supply to provide multiple performance modes. The number of tunable blocks using a high-Vdd supply varies depending on whether the design is in a fast mode that uses more energy or in a slow mode that uses less energy. There are also intermediate modes that statically assign only a subset of tunable blocks to use the high-Vdd supply. Agarwal's intermediate voltage assignment is determined during the design phase, while TuneLogic uses a post-manufacturing assignment that can compensate for actual process variations instead of estimated process variations.

Adaptive voltage scaling (AVS) [15, 20] uses a single supply voltage that

can be varied based on the performance of a particular chip. TuneLogic is orthogonal to AVS. TuneLogic could use the same AVS voltage setting mechanism to set its high-Vdd supply voltage. Then, TuneLogic would use a low-Vdd supply voltage that is 200 mV less that the high-Vdd supply. TuneLogic's dual-Vdd designs would use even less power than AVS. Using AVS to set the high-Vdd supply would allow TuneLogic to achieve even larger power savings than those presented in Section 2.4.

Li *et al.* [31] present additional work on pre-manufacturing dual-Vdd voltage assignment. In Li's work, the voltage assignment is based on power sensitivity using nominal transistor parameter values. One limitation of this approach is that when dealing with random process variations, it is unknown at design time which tunable blocks will have the largest power sensitivity. Additionally, this pre-manufacturing assignment tries to assign short paths to use the low-Vdd supply, but in an actual design with pipeline stages, most paths have been optimized by resizing and retiming to have the same delay. For the TuneLogic pipelined multiplier in the nominal case, no full adder cell can be slowed without slowing the entire design. Therefore, there are very few paths that can be assigned to a low-Vdd supply voltage based on pre-manufacturing information.

2.6 Conclusion

We have presented a design style, associated CAD algorithms, tools, and a methodology to tune logic post-manufacturing. TuneLogic makes it possible to more tightly meet the design goals while achieving *additional* yield increases and energy decreases over a range of design goals. TuneLogic is dependent only on

post-silicon delay testing; in particular, it is independent of the delay distribution and correlations between tunable blocks.

Future work on TuneLogic could include developing AVS-based grid voltage selection, automating the partitioning of logic into blocks, and quantifying the cost of at-speed testing.

Chapter 3

TuneFPGA

In addition to mitigating the effects of process variation, another motivation for this work is the increasing cost of ASIC design. The mask cost alone for recent technology processes has risen into the millions of dollars [46]. Once the cost of chip architects and circuit designers is factored in, only a select few applications will be able to justify the cost to produce an ASIC. As the speed of FPGAs increases, signal processing applications will be able to migrate away from high cost ASICs to more cost-effective FPGAs [45, page 32]. Current commercial FPGA chips in 65nm process technology have cores that run at 600MHz with 338,000 logic elements and 578 multipliers [4, 5]. While these state-of-the-art FPGA chips with the largest number of gates might cost thousands of dollars, the price should drop by a factor of ten within one or two process generations. Kuon and Rose [30] present experimental measurements that show FPGA performance to be about 4 times slower than a standard-cell ASIC. By working to further improve FGPA performance for future process technologies, FPGAs will become an even more popular alternative to ASICs. This work was presented at DAC [8].

As in the previous chapter, TuneFPGA also tunes the individual chips after manufacturing. A distinguishing feature of TuneFPGA, though, is that it individu-

ally tests each configurable logic block's (CLB) performance, as opposed to testing only an entire path. Then, TuneFPGA uses the measured CLB performance to assignment supply voltages. Last, the chip is programmed with the CLB supply voltage assignments using the infrastructure already present on FGPA chips. It is important to emphasize that, as in TuneLogic, designers will still be able to use all of their current tools since TuneFPGA takes place after manufacturing.

3.1 Dual Voltage CLB Design

As in the other chapters, TuneFPGA continues to concentrate on a dual voltage supply. Since there will be multiple voltage supplies on the chip, level converters are required at all junctions between low voltage outputs and high voltage inputs. A level converter is a buffer with additional transistors to help prevent short circuit current. TuneFPGA incorporates the Kulkarni *et al.* [29] STR6 level converter into the output multiplexer. By placing a level converter in the output multiplexer, TuneFPGA ensures that all CLB outputs will be high voltage. In order to preserve speed, the high-Vdd path through the multiplexer bypasses this level converter.

3.1.1 Tunable CLB

Figure 3.1 shows the CLB we designed. This CLB uses a lookup table (LUT), a register, and a multiplexer (MUX) to bypass the register. Compared with the Altera ALM and Xilinx CLB, the major addition to our CLB is that there are pass transistor connections to both a high-supply voltage grid and a low-supply volt-

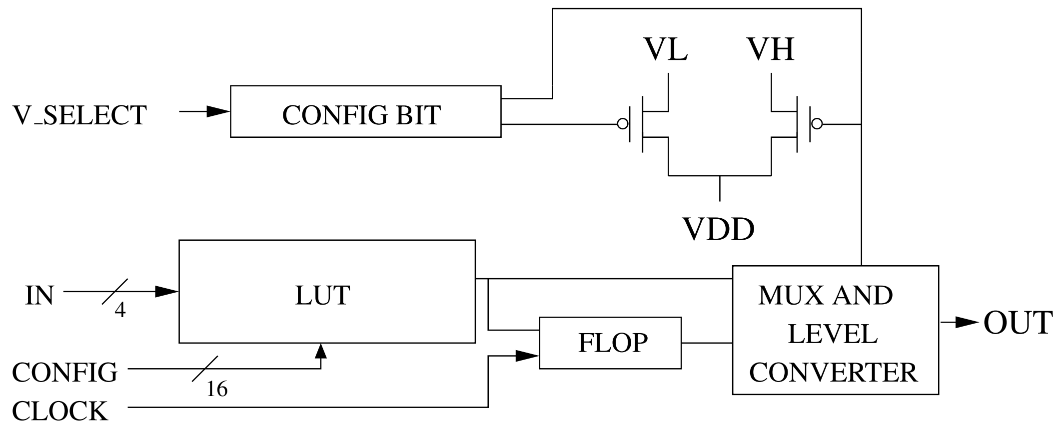

Figure 3.1: Tunable dual-Vdd CLB with configurable voltage supply PMOS transistors. The lookup table, flop, and multiplexer are connected to the local Vdd. VH is connected to the high supply voltage grid, and VL is connect to the low supply voltage grid. The multiplexer includes a level converter in the VL path only. Therefore, the CLB output is always high voltage.

age grid [31]. The LUT, flop, and multiplexer are all connected to a local Vdd for each CLB. The CLB can switch from one supply voltage grid to the other by turning on one of the large pmos pass transistors connected to the CLB's Vdd interconnect. Then, the local Vdd is used to connect to the CLB elements. The supply voltage configuration is stored in a register with complementary outputs so that only one pmos pass transistor can be turned on at a time. In this design, the supply voltage configuration is programmed before normal operation of the CLB begins.

3.1.2 Area Overhead

In order to size the pmos pass transistor, a SPICE simulation sweep was performed. The width was varied from $50\times$ minimum size to $500\times$ minimum size.

The chosen pass transistor size of $133\times$ the minimum size resulted in less than a 1% increase in LUT delay compared to having no pass transistor. After layout, the area of one pass transistor was comparable to a standard cell gate. In situations in which total area is more important than delay, a smaller pass transistor can be used.

The STR6 level converter [29] is composed of a buffer with 6 additional small transistors. These 6 additional transistors have about the same transistor width as a buffer. The overall area cost of the level converter is about the same as 2 buffers, which is small compared with the total area of a CLB cell.

3.2 CLB Delay Statistics

We performed Monte Carlo simulations on the dual-Vdd CLB shown in Figure 3.1. The 65 nm Berkeley PTM model [12] was used for the SPICE transistor models. For each transistor in the CLB, both the transistor length and threshold voltage were modeled as uncorrelated Gaussian distributions with the $3\,\sigma/\mu$ variation chosen to be 20% [53]. The high voltage supply was set to 1.2 V, and the low voltage supply was set to 0.8 V [19]. The simulation temperature was 85 °C.

We measured the performance of our CLB in 100,000 Monte Carlo simulations using HSPICE. The LUT was programmed for the worst-case delay configuration, which was fifteen logic-zeros and one logic-one. This leads to a slow rise time for LUT output, which in turns leads to a slow fall time for the CLB output because of the inverting multiplexer. Delay was measured from the 50% crossing of the input signal to the 50% crossing of the output signal.

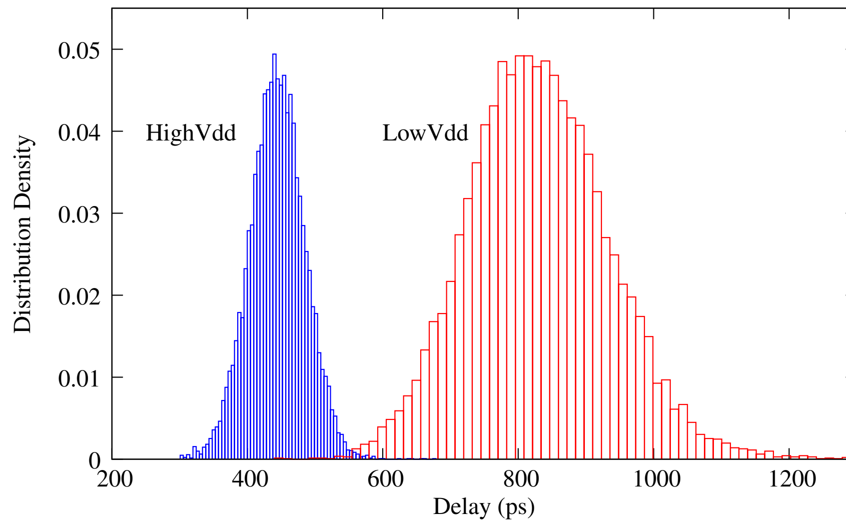

Figure 3.2: Dual-Vdd delay distribution for a single CLB

In Figure 3.2, the worst-case falling output transition delay distribution is shown for all 100,000 trials. In this figure, the bars on the left represent the delay when all of the CLB elements are connected to the high voltage supply through the pmos pass transistor. Then, the CLB is disconnected from the high voltage supply and connected to the low voltage supply through the other pmos pass transistor. The bars on the right represent the delay when all of the CLB elements are connected to the low voltage supply. In order to present an equal comparison, the number of bars on the left is equal to the number of bars on the right. This graph shows that there is a large spread of delay values for this CLB when intra-chip variation is taken into account. Moreover, the low-Vdd distribution has a larger amount of variation.

The average power was measured from when the input changes by 1% until the output reaches 99% of its final value. The power includes both switching and leakage. Figure 3.3 show the total power values; the power spreads are similar to

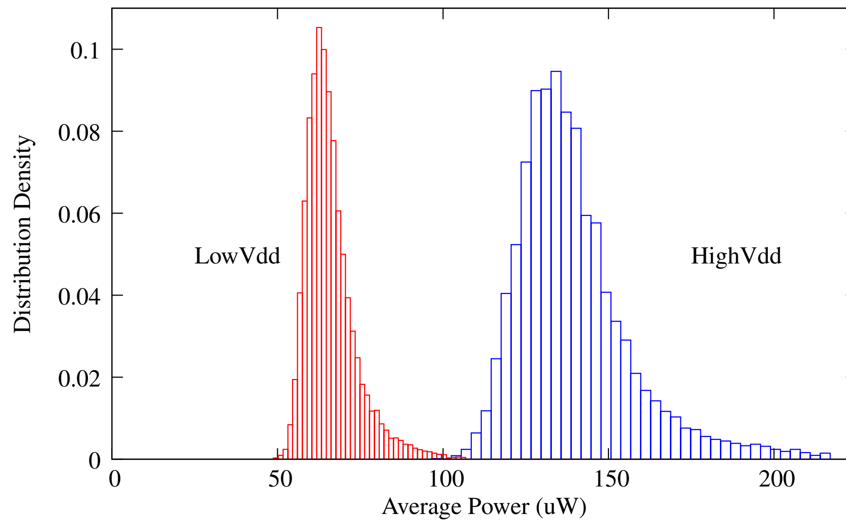

Figure 3.3: Dual-Vdd power distribution for a single CLB

the delay spreads. There are also similar graphs for leakage power alone.

Taken together, these graphs show that there is potential to reduce both dynamic power and leakage power. The following summarizes the potential power savings for a single CLB when using a dual-Vdd approach:

1. The delay increases from a mean of 444 ps using high-Vdd to 831 ps using low-Vdd.
2. The power decreases from a mean of 140 μw using high-Vdd to 70 μw using low-Vdd.
3. The leakage power decreases from a mean of 25.7 μw using high-Vdd to 15.7 μw using low-Vdd.

If the designer is able to decrease the speed on non-critical path CLBs by switching those CLBs to a low voltage supply, power can be reduced. Even with CLBs on the

critical path, if there is enough slack, some of those CLBs could use a low supply voltage while the other CLBs use a high supply voltage. Overall, there is potential to reduce total power by almost 50%.

3.3 Post-Manufacturing Delay Measurement

TuneFPGA uses post-manufacturing CLB delay measurement, which is the subject of ongoing research. Nabaa *et al.* [41] present a tunable body bias FPGA that includes a CLB characterizer. Nabaa's characterizer uses a phase detector to measure delay. The characterizer sends a clock signal to the CLB under test. The output of the CLB under test is sent back to the characterizer. The phase detector compares the initial clock signal to the CLB output. Using this method, the characterizer measures the delay of both the CLB and any intermediate routing resources. The characterizer starts with CLBs that are adjacent to the characterizer, proceeding to sequentially test CLBs further away while compensating for the already measured delay through intervening CLBs and interconnects. Nabaa has estimated that the area of the characterizer is about nine FPGA tiles.

Another promising approach would be to implement delay characterization based on Razor by Ernst *et al.* [21]. By using a shadow latch and comparator logic, Razor has mechanisms to monitor when a delay error has taken place. For TuneFPGA, a test input could run through the CLB in successively faster clock cycles until there is a delay error. Additionally, neighboring CLBs could perform the shadow latching and comparator logic needed for Razor testing using existing CLB resources.

Other delay characterization methods include the work by Dhar *et al.* [18], which introduces an adaptive voltage scaling controller using an inexpensive ring oscillator to measure speed. By placing multiple ring oscillators throughout the design, CLB delay can be approximated based on the delay of the nearest ring oscillator.

3.4 CAD Flow

TuneFPGA integrates existing design tools with a custom C++ program we wrote to implement the voltage assignment algorithm described below. The inputs to TuneFPGA are a manufactured design with CLB delay measurements, a logic netlist to use for static timing analysis, and a target delay. The output is a voltage assignment for each CLB. Since TuneFPGA is based on static timing analysis, it is very fast.

TuneFPGA is now ready for the voltage assignment algorithm (VAA), shown in Algorithm 3.1. TuneFPGA VAA starts by assigning all CLBs to use a low-Vdd supply. Static timing analysis is done for the entire design using the measured CLB delay values. CLBs on a path that does not meet the design delay are marked as failing CLBs. A greedy selection algorithm then assigns a high-Vdd supply to the failing CLB that most improves the worst negative slack (WNS). Static timing analysis is repeated for the entire design, and the greedy selection algorithm is run again. This iterative process continues until either all of the paths meet the target delay or all of the failing CLBs have been switched to a high-Vdd voltage supply. The average runtime of TuneFPGA-VAA on the benchmark with the largest number

Algorithm 3.1 Voltage Assignment Algorithm

1: Assign all CLBs to use the low supply voltage
2: Perform static timing analysis using the measured CLB delay values
3: ClbFailingQueue $\Leftarrow$ CLBs on paths that do not meet the design delay
4: **while** !ClbFailingQueue.empty() && Low supply voltage CLBs > 0 **do**
5: UpdateCLB $\Leftarrow$ CLB that most improves the worst negative slack when assigned to the high supply voltage
6: Permanently assign UpdateCLB to use the high supply voltage
7: Repeat static timing analysis
8: ClbFailingQueue $\Leftarrow$ CLBs on paths that do not meet the design delay
9: **end while**

of CLBs (C5315) was 0.4 seconds on a 3.2 GHz Intel Xeon.

3.5 Experiments

We use the ISCAS-85 combinational logic benchmarks to study the advantages of post-silicon tunable logic. The benchmarks include both control flow and datapath logic. The benchmarks were analyzed using the TuneFGPA methodology described in Section 3.4. All of the benchmarks were synthesized, optimized, and then mapped to 4-input LUTs using the RASP version of SIS and FlowMap [16]. Individual CLB delay and power values were randomly assigned using the CLBs simulated in Section 3.2. These benchmarks have a logic depth from 4 CLBs (C499) to 13 CLBs (C3540).

3.5.1 Setup for Experiments

Results are summarized in Tables 3.1 and 3.2. Key details about the experimental setup are as follows:

- Our implementation of Li's [31] pre-manufacturing voltage assignment has a 10% guard band. As discussed in more detail in Section 3.5.3, this was the best guard band that we found.
- Yield, power, and the percentage of high-Vdd supply CLBs are calculated for each target delay. The range of target delays was chosen such that the all high-Vdd has a yield of 50% at the fastest target delay and the all low-Vdd has a yield of 50% at the slowest target delay.
- The figure of merit (FOM) for each voltage assignment is the yield/power ratio, which is then normalized to the TuneFPGA FOM value for each target delay.

3.5.2 Experimental Results

The key take-aways from the experimental results in Tables 3.1 and 3.2 are as follows:

- TuneFPGA had exactly the same yield as High; this is to be expected because TuneFPGA can set all CLBs to high-voltage if needed. TuneFPGA improved the geometric mean of the power across all benchmarks by 40%.
- TuneFPGA had better yield and used less power than Li [31]. TuneFPGA

Table 3.1: Benchmark results using 4 different voltage assignments. Each benchmark was instantiated on 1000 chips, each of which has independent variations. TuneFPGA: post-manufacturing voltage assignment using TuneFPGA. Li [31]: our implementation of Li's pre-manufacturing dual-Vdd voltage assignment. All High: every CLB is connected to the high-Vdd supply. All Low: every CLB is connected to the low-Vdd supply.

Target Delay (ns)	Yield (%)				Average Power (mW)				High-Vdd CLBs (%)	
	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]
2.00	50.6	50.6	50.6	0	10.4	10.4	10.4	n/a	99.9	100
2.32	100	100	100	0	9.8	10.4	10.4	n/a	89.1	100
2.64	100	95.1	100	0	9.0	9.7	10.4	n/a	72.9	86.5
2.97	100	60.5	100	0	8.0	9.1	10.4	n/a	54.6	75.7
3.29	100	9.9	100	0	6.9	7.4	10.4	n/a	33.5	43.2
3.61	100	89.8	100	0.1	5.7	7.4	10.4	5.0	9.7	43.2
3.93	100	50.1	100	50.1	5.2	5.2	10.4	5.2	0.9	0

(a) C499: Implemented with 74 CLBs

Target Delay (ns)	Yield (%)				Average Power (mW)				High-Vdd CLBs (%)	
	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]
6.12	50.0	50.0	50.0	0	45.3	45.8	71.3	n/a	27.3	28.7
7.05	100	99.8	100	0	41.6	45.0	71.3	n/a	17.0	26.5
7.99	100	99.7	100	0	39.1	42.2	71.3	n/a	10.0	18.7
8.92	100	99.6	100	0	37.5	40.7	71.3	n/a	5.6	14.5
9.85	100	99.6	100	0	36.5	39.9	71.3	n/a	2.7	12.2
10.78	100	91.5	100	0	35.7	37.8	71.3	n/a	0.7	6.3
11.71	100	90.5	100	50.0	35.5	35.6	71.3	35.5	0.1	0.2

(b) C3540: Implemented with 509 CLBs

Table 3.2: Additional benchmarks

Target Delay (ns)	Yield (%)				Average Power (mW)				High-Vdd CLBs (%)	
	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]
4.25	50.4	50.1	50.4	0	64.7	68.6	102	n/a	27.6	35.3
4.90	100	86.5	100	0	59.6	65.4	102	n/a	17.7	29.1
5.55	100	93.8	100	0	55.7	63.1	102	n/a	10.1	24.6
6.20	100	67.1	100	0	52.5	56.6	102	n/a	3.9	11.9
6.86	100	92.5	100	0	51.3	53.5	102	n/a	1.5	5.7
7.51	100	87.4	100	0	50.9	52.0	102	n/a	0.7	2.8
8.16	100	95.6	100	50.2	50.6	50.9	102	50.6	0.1	0.6

(a) C5315: Implemented with 725 CLBs

Target Delay (ns)	Yield (%)				Average Power (mW)				High-Vdd CLBs (%)	
	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]	All High	All Low	Tune FPGA	Li [31]
11.55	50.3	50.3	50.3	0	56.4	57.2	74.0	n/a	52.6	54.9
13.28	100	100	100	0	50.1	56.8	74.0	n/a	35.8	53.8
15.01	100	100	100	0	43.9	53.6	74.0	n/a	19.1	45.3
16.74	100	100	100	0	39.1	48.4	74.0	n/a	6.2	31.2
18.48	100	99.6	100	0	37.5	41.3	74.0	n/a	1.8	12.1
20.20	100	99.4	100	0	37.1	37.3	74.0	n/a	0.8	1.3
21.94	100	99.6	100	50.0	36.9	37.0	74.0	36.8	0.1	0.6

(b) C6288: Implemented with 528 CLBs

improved the geometric mean of the yield/power ratio across all benchmarks by 27%.

Further observations from analyzing the results are as follows:

- Designs with mostly short paths show a greater impact from variations. For the C499 benchmark, which has a logic depth of 4 CLBs, selecting voltages post-manufacturing results in yield increases of up to $10\times$ compared with selecting voltages pre-manufacturing with the same delay target.
- Li [31] pre-manufacturing assignment yields are not always monotonically increasing because each target delay has a custom high-Vdd map. Therefore, some target delays will have a better yield for a given assignment map than other target delays.
- Since the Li [31] pre-manufacturing assignment has a 10% guard band, power values for a given target delay are usually more closely matched to a TuneFPGA assignment with a 10% guard band as well. C3540 has a pre-manufacturing power value of 42.2 mW at 7.99 ns, while the TuneFPGA power is 41.6 mW at 7.05 ns.
- High has a large jump in yield between the fastest target and the second fastest target. For C499 with High, a 2.00 ns target has a yield of 50% and a 2.32 ns target has a yield of 100%. Figure 3.2 shows that high-Vdd CLBs have delay variations of about 0.20 ns. Therefore, small changes in target delay effectively provide margin for high-Vdd CLB delay variations.

3.5.3 Guard Banding Voltage Selection

As a comparison to TuneFPGA, we also experimented with a pre-manufacturing dual-Vdd voltage assignment based on the work of Li *et al.* [31]. Li performed voltage assignment based on power sensitivity. Our implementation of Li's voltage assignment uses the same methodology as TuneFPGA except that the CLB delays use nominal transistor parameter values. The list of CLBs that use a high-Vdd supply is then saved for individual chip configuration.

In order to improve the yield of the pre-manufacturing voltage assignment, the target delay was given a 10% guard band. For example, if the target delay after manufacturing was 5 ns, then the pre-manufacturing voltage assignment would switch enough cells so that the design would have at most 4.5 ns of delay when using nominal transistor values. Since the cell delays are discrete values, the actual target delay would most likely be less than 4.5 ns. One example is that C499 has the same voltage assignment for both 3.29 ns and 3.61 ns target delays, yet a target delay of 3.29 ns has a yield of 10%, while target delay 3.61 ns has a yield of 90%. Guard band values from 0% to 20% were used, with 10% resulting in the best yield/power ratio.

It is noteworthy that designs with no guard band led to abysmal yield. For C5315, using *nominal* transistor values, a target delay of 4.90 ns is always achievable using pre-manufacturing assignment. However, with process variation, the yield drops. A guard band of 10% results in a pre-manufacturing yield of 87%. A 5% guard band has a yield of 64%, while no guard band has just a 3% yield.

3.6 Related Work

As in TuneLogic from the previous chapter, TuneFPGA primarily differs from previous work in that the tunable CLB is configured after fabrication is completed. Therefore, this tunable CLB responds to measured variability instead of predicted values.

Li *et al.* [31, 34] propose adding both supply voltage programmability and interconnect voltage programmability to FPGAs. They include designs for integrating multiple supply voltages into the FPGA framework. Voltage supply assignment is based on available timing slack for each path during the pre-manufacturing design phase. Compared to single voltage supply FPGAs, Li *et al.* report power savings of up to 47% and energy-delay product savings of up to 27%. While Li's work on using dual voltage supplies is similar to TuneFPGA, there is an advantage to performing voltage supply assignment only after each chip is fabricated. Because our method tunes individual chips, it is possible to more tightly meet the design goals.

Katsuki *et al.* [27] characterize the performance of each LUT for each fabricated chip. Katsuki *et al.* use a course-grained measurement technique that counts the number of LUTs that a one signal passes through in a single clock cycle. Using this performance information, the LUTs are then placed based on critical path delay. Each chip goes through this placement optimization stage, which has the potential to be very time-consuming. TuneFPGA also proposes to characterize each LUT, but it will use the same placement and routing for each chip. What will change is that each CLB will have an additional configuration parameter that will select one of two supply voltages in order to meet design objectives. Assigning this addi-

tional configuration parameter should take significantly less time than performing an individual place and route for each chip.

Matsumoto *et al.* [37] propose having multiple possible placement and route configurations for each design as another way to change the critical path for a particular fabricated chip. After each chip is fabricated, the multiple configurations are tested to determine the configuration that has the best delay for that particular chip. With this method, the goal is for the critical path of each configuration to use different resources while at the same time striving to have the same delay. Depending on the size of the design, this process could take a significant amount of design time in order to develop multiple placement and route configurations. In TuneFPGA, there is only one placement and route for all of the fabricated chips. Each CLB will have design features that can be configured to satisfy the design requirements of the chip. Consequently, our work will not require multiple placement and routing configurations.

While we have opted for the greedy approach of starting with low supply voltages for all CLBs and iteratively selecting a CLB that most improves the worst negative slack, there are a number of approaches to parameter selection that take a more global view of the design. Such approaches are exemplified by the work of Chen *et al.* [14], who formulate delay-constrained gate area minimization as a constrained optimization problem. The objective is the sum of gate sizes, and the constraint is that each output settles by the required arrival time. Chen *et al.* solve the constrained optimization problem using Lagrangian relaxation. They point out a special structure in the constraints that guides the update mechanism for the La-

grange multipliers, leading to fewer iterations. Similar approaches have been taken to other design parameter selection problems. It would be straightforward to adapt TuneFPGA's voltage assignment algorithm to be more global. However, this would considerably increase the runtime, which is undesirable since this assignment is computed on a per-chip rather than a per-design basis.

3.7 Conclusion and Future Work

TuneFPGA increases the yield and decreases the power of designs that have process variations, even when compared to guard banding. TuneFPGA is dependent only on a post-silicon delay map. While FPGA interconnect is not addressed in the current work, we realize that interconnect contributes a large portion of the total FPGA delay and power [57]. We anticipate that these TuneFPGA techniques will transfer to interconnect in a similar manner as CLBs. Future work will also investigate whether there are any additional gains from modeling correlated inter-chip variation in addition to the presented uncorrelated intra-chip variation.

Chapter 4

TuneRAM

Modern computer architecture makes extensive use of SRAMs. For instance, SRAM cells are prominent in multi-level caches, register files, and TLBs. In many current designs, SRAMs constitute more than half of chip area and more than half of the number of devices [49]. Consequently, SRAMs impact area, power, timing, yield, and schedule. This work was presented at ICCD [39].

Additionally, SRAMs often use transistors that are smaller than those found in logic gates, and these smaller transistors are more sensitive to process variations. For example, the threshold voltage variation for minimally sized devices has been estimated to be more than twice the variation for nominally sized devices [28]. Balancing the tradeoffs between small area, low power, fast reads and writes, and sensitivity to process variations are an essential part of any SRAM design optimization.

Another concern for SRAM design is that, for sub 90nm process technology, pmos *negative bias temperature instability* (NBTI) [36] has led to situations in which the pmos threshold voltage can change over time. These NBTI threshold voltage changes reduce the *static noise margin* (SNM) of the SRAM. In order to compensate for a reduced SNM, the minimum allowed supply voltage V_{ddmin} is

increased. In fact, SRAM stability and yield are often the limiting factors in determining the minimum supply voltage for a design. Therefore, increasing SRAM stability and yield would directly lead to higher chip yield and lower supply voltage.

In this chapter, we assign supply voltages post-manufacturing to columns of SRAMs cells. This way, the SRAM supply voltage is based on actual silicon instead of relying on 5σ estimated values. Furthermore, we are able to customize supply voltage to a small group of SRAM cells instead of requiring a fixed supply voltage for the entire SRAM array. We use realistic process data to study the effects of our post-manufacturing voltage assignment. Our results show that post-manufacturing voltage assignment results in a 22% reduction in bitline energy compared with a fixed voltage design.

The remaining sections of this chapter are organized as follows: Section 4.1 gives background on SRAM cell design. In Section 4.2, we describe our approach to reducing SRAM power. Section 4.3 contains experimental results. Section 4.4 presents related work in SRAM design. Section 4.5 summarizes this work.

4.1 Background: SRAM Cell Design

On-chip memories are constructed using large arrays of 6T SRAM cells combined with supporting logic for control and address decoding. Typically, the 6T SRAM cells constitute 80–90% of the total on-chip memory area.

Figure 4.1 is a schematic of a normal 6T SRAM cell. The main function of this cell is to store data. SRAM cell design involves balancing a number of

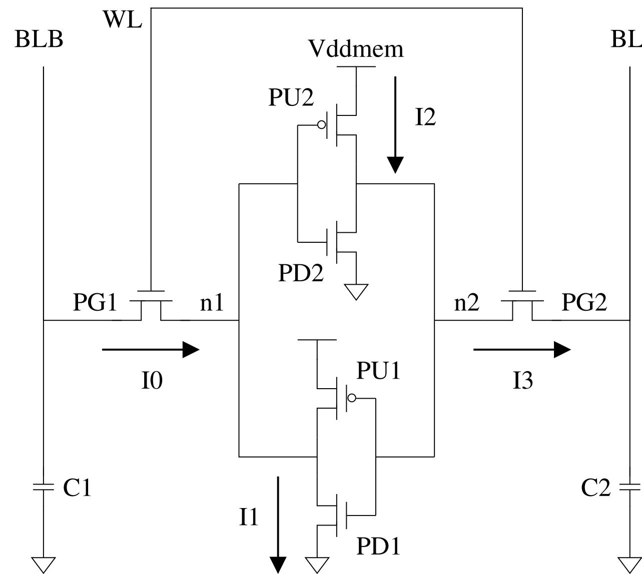

Figure 4.1: 6T SRAM schematic

requirements, including the following:

- Minimizing cell area
- Ensuring read and write stability
- Minimizing supply voltage to reduce power
- High cell read current to minimize access time
- Minimizing leakage current
- Reducing bitline swing to reduce power
- Good soft error immunity

Many of these criteria are conflicting in nature. For example, good cell stability, fast access time, and good soft error immunity would benefit from using larger transistors, but larger transistors result in larger area and increased leakage.

Another example is that increasing the *cell ratio* (CR), defined in (4.1), through the use of a smaller pass-gate transistor (*PG*) improves the SNM, but the smaller *PG* transistor decreases the write margin.

$$Cell\ Ratio\,(CR) \triangleq \frac{W_{pd}/L_{pd}}{W_{pg}/L_{pg}} \tag{4.1}$$

SRAMs must function properly during all of the process, voltage, and temperature (PVT) corners. For read access, the current through *PD1* (I1) must be greater than or equal to the current through *PG1* (I0), i.e.,

$$\mu_n C_{ox}\left(\frac{W_{pd}}{L_{pd}}\right)\left(V_{ddmem} - V_{tn} - \frac{V_{n1}}{2}\right)V_{n1} \geq \frac{\mu_n C_{ox}}{2}\left(\frac{W_{pg}}{L_{pg}}\right)(V_{ddwl} - V_{n1} - V_t)^{\alpha} \tag{4.2}$$

During write, the current through *PG2* (I3) must be greater than or equal to the current through *PU2* (I2), i.e.,

$$\mu_n C_{ox}\left(\frac{W_{pg}}{L_{pg}}\right)\left(V_{ddwl} - V_{bit} - V_{tn} - \frac{V_{n2}}{2}\right)V_{n2} \geq \frac{\mu_p C_{ox}}{2}\left(\frac{W_{pu}}{L_{pu}}\right)(0 - V_{ddmem} - V_{tp})^{\alpha} \tag{4.3}$$

The SRAM cell transistor sizes must satisfy both (4.2) and (4.3). The SRAM cell must also produce the required read access current, I_{read}, which is determined by the *PG* and *PD* transistors. The current I_{read} is especially important because it is usually the determining factor for overall cache speed. Even after simulating the SRAMs for all of the PVT corners, it is still often necessary to obtain silicon data in order to best tune the SRAM cell sizes and layout for a robust design.

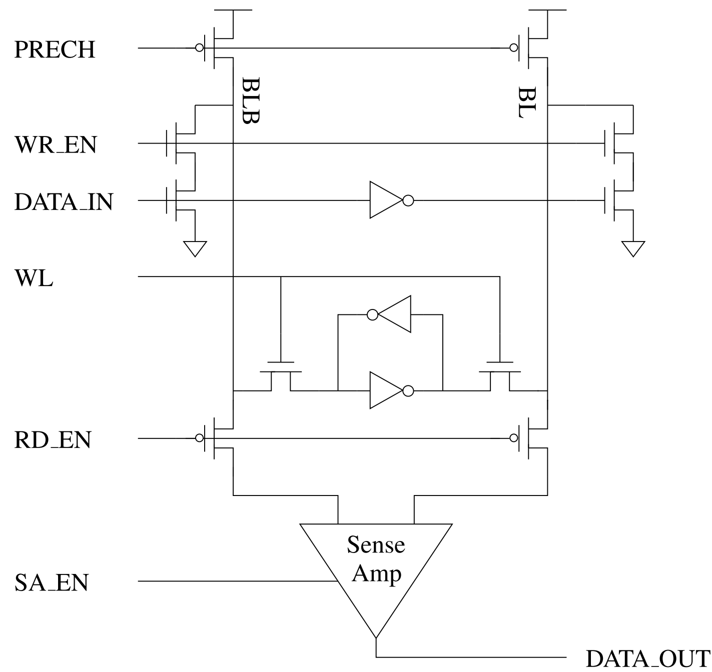

Figure 4.2: SRAM column schematic

4.2 Adaptive SRAM

On-chip memories are designed using a large number of identical SRAM cells. After manufacturing, though, process variations lead to dissimilar on-silicon SRAM cells. This results in the current design methodology, in which all of the SRAM cells must be designed to work with the worst process variation. While general datapath and control logic might be designed for 3σ worth of process variation, because of the larger number of cells, SRAMs typically have 5σ of guard banding in order to achieve acceptable yields. In the next section we show how a guard banded SRAM design leads to a large amount of wasted power.

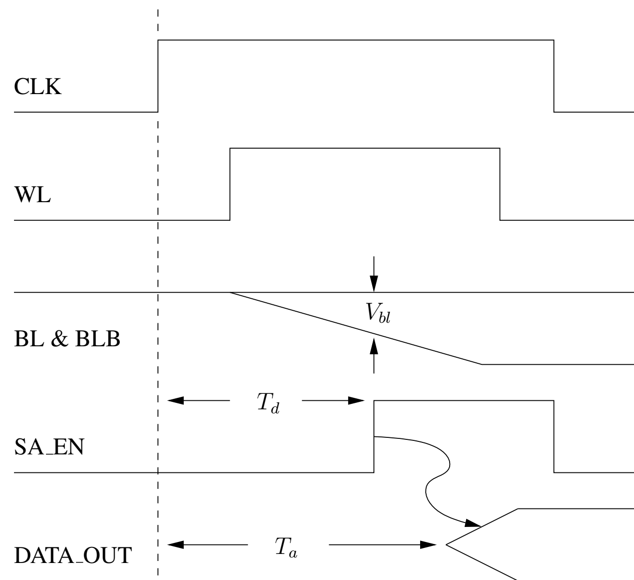

Figure 4.3: Bitline timing diagram. V_{bl} is the amount of seperation between BL and BLB when the SA_EN is asserted.

4.2.1 Memory Operation

Figure 4.2 shows the schematic for one column of an SRAM array with a single SRAM bit, and Figure 4.3 shows the associated timing diagram for reading a single bit from that column. In a memory array, though, the bitlines (BL and BLB) would be connected to multiple SRAM cells. Overall, these bitlines are large contributors to memory bank power [63], so bitline power savings have a direct effect on overall SRAM power. The read operation starts with both bitline (BL) and bitline-bar (BLB) being pre-charged high. When the word line (WL) is asserted, the SRAM cell starts to pull down either BL or BLB via the nmos devices *PD1* or *PD2* (cf. Figure 4.1). After a delay of $\mathrm{T_d}$, the sense amplifier enable signal (*SA_EN*) initiates the sense amplifier reading of the bitlines. The amount of separation be-

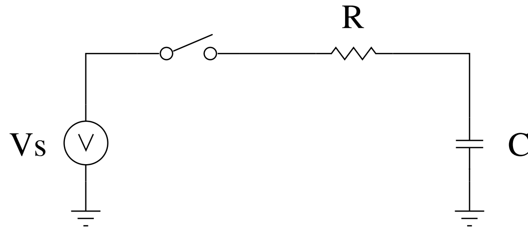

Figure 4.4: RC circuit schematic used to derive the energy expended for precharging and evaluting the bitlines

tween BL and BLB when *SA_EN* is asserted is referred to as *bitline development* (V_{bl}). The sense amplifier is designed to guarantee correct read operation from the SRAM cell when V_{bl} has a certain minimum value.

Figure 4.4 shows the schematic that is used to derive the total energy expended in pre-charging and evaluating the bitlines. Since only one bitline is evaluated during a cycle, C is the total capacitance of one bitline. The resistance R is the equivalent resistance of the additional circuit elements that appear between the voltage supply and the bitline. The total energy for precharging and evaluating the bitline is given by (see Appendix B):

$$\text{Energy}_{bitline} = \text{C V}_{bl}\ \text{V}_{dd} \tag{4.4}$$

From (4.4) it is clear that a larger bitline development leads to more energy consumed. Normally, bitline development is controlled by the difference in time between asserting the wordline and asserting the sense amplifier enable. The longer the wordline pulse stays high, the lower the bitline voltage falls.

Figure 4.5 shows the amount of bitline development for 1,000 SRAM instances considering process variation (the detailed setup is given in Sec. 4.3). For

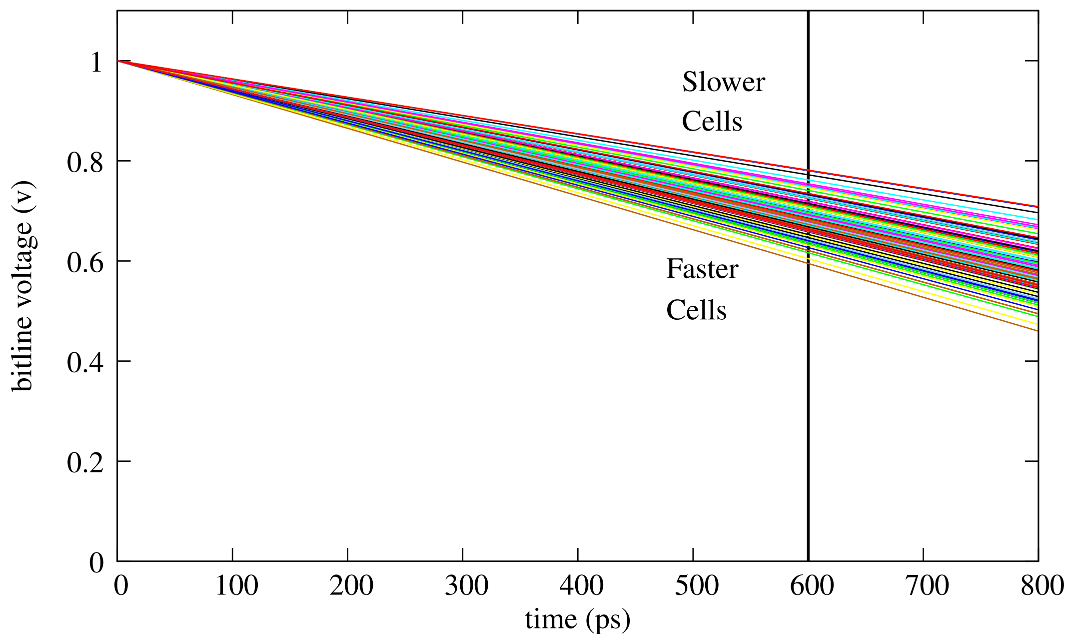

Figure 4.5: 1000 SRAM bitline voltages. The bitline is sampled by the sense amp 600 ps after the word line is asserted (solid line). The SRAM supply voltage is 1.0 V.

this design, the sense amp required 200 mV of bitline development for proper operation. In order for all cells to reach 200 mV of bitline development, *SA_EN* must be asserted at least 600 ps after the wordline is asserted. As shown in Figure 4.5, the slowest cells determine the minimum time to assert *SA_EN*. However, most of the SRAM cells produce larger than required amounts of bitline development, which is a waste of energy.

4.2.2 Adaptive Voltage Supply

Figure 4.6 shows a 4-bit by 4-bit SRAM array that illustrates our adaptive design. Each column of SRAM cells can be assigned to one of four different voltage supplies by enabling one of the power configuration transistors at the top of each column. It is important to note that we are assigning voltage supplies only to the 6T cells, and not to the pre-charge or the word line drivers. By enabling the power configuration transistor that best matches the needs of all of the SRAMs in a column, we are able to tailor the voltage supply to the particular needs of that column.

When designing the power configuration transistor, it is important to consider the current requirements for the column voltage supply. Figure 4.2 shows that the pre-charge transistor *PRECH* pulls up the bitlines. Therefore, the SRAM cell only has to pull down the bitline. From Figure 4.1, we can see that V_{ddmem} is used to drive the nmos gate for *PD1* and *PD2*. In the SRAM array, the wordline ensures that only one SRAM cell in a column is active at one time. At any given time, the SRAM column supply voltage never drives more than a single nmos gate; it never charges the bitline. Therefore, the power configuration transistor only needs to supply a very small amount of current.

Since the current requirement is so small, the power configuration transistors can be very small and dense. Therefore, four minimum size pmos transistors can be used as the power transistors. Then, an electrical fuse can be programmed with the configuration to turn on one of the pmos transistors [26]. The four minimum sized pmos transistors would be 0.12 μm^2 in area. The fuses would use

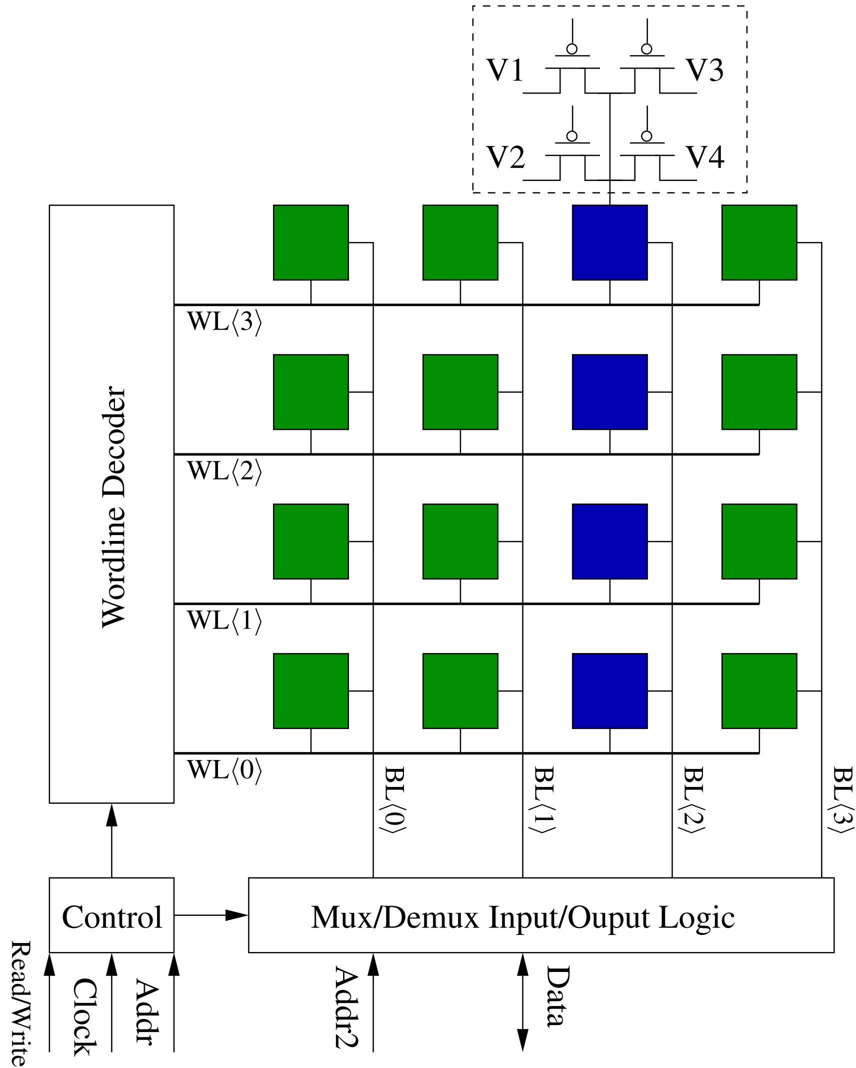

Figure 4.6: 16-bit SRAM array. The dotted box at the top of BL⟨2⟩ is the configurable voltage supply for one column of SRAM cells; the other columns would have a similar structure. V1 through V4 are the four different voltage supplies. By turning on one pmos power transistor, the SRAM voltage for that column can be adjusted. The pmos power transistors are configured using fuses (not shown).

0.46 μm^2. Compared to a column containing 16 SRAM cells, the area overhead would be about 15%. The SRAM columns contribute to about 50% of the total memory array area. Therefore, the adaptive memory for 16 SRAMs per column would be about 8% of the total memory array size.

In addition to the power transistor and configuration, this adaptive memory also requires four voltage supplies. Current designs usually have a Power Management IC (PMIC) that is already capable of generating multiple voltages. Since very little current is used in the SRAM supply voltages, there is no need for a full voltage grid. The SRAM supply voltages can use signal routing from the PMIC to the memory array. We believe that, similar to scan signals, these signals can be routed at the end of the physical design stage using any available metal layers with only small incremental cost.

4.2.3 Voltage Assignment Algorithm

Built-in self test (BIST) is the predominant way to test memory [1]. Testing is done at full speed with an algorithmic pattern generator and cycle-by-cycle response comparison with a pass or fail signature. The BIST engine has a simple interface with input, output, address, enable signal, and pass or fail signal. It also keeps track of the failing addresses for use by the failure analysis and redundancy logic.

Our approach, shown in Figure 4.7, assumes a BIST engine with similar functionality. Essentially, we propose running BIST multiple times, once for each supply voltage. By running BIST for each supply voltage, BIST can determine the

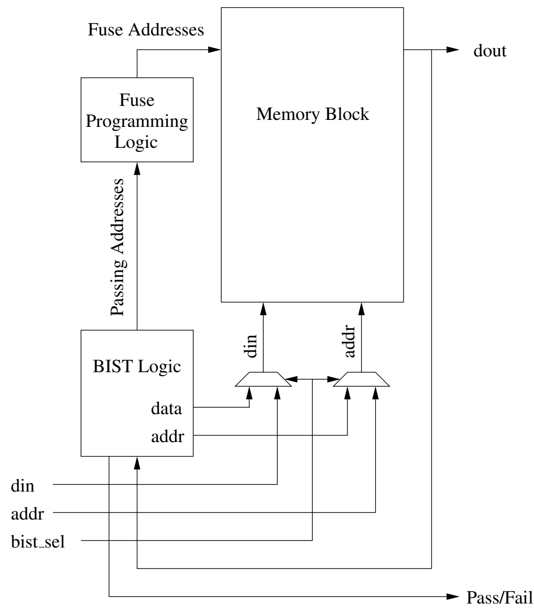

Figure 4.7: BIST block diagram

minimum supply voltage that is needed for a given group of cells. Since BIST is already required for testing, the added overhead for our approach is the wiring from the passing group IDs to the fuses and the time it takes to run BIST during the configuration iterations.

The sequence of testing, shown in Algorithm 4.1, starts by setting all of the SRAM cells to the highest supply voltage. If a group of cells does not pass testing at the highest supply voltage, that group is marked as failing because there is nothing more that can be done for that group. If the group passes testing, the second iteration of BIST testing starts by assigning all of the remaining memory groups to the intermediate high supply voltage. The IDs of the groups that fail testing at the intermediate high supply voltage are sent to the fuse programming logic to be

Algorithm 4.1 Voltage Assignment Algorithm

```
 1: Assign all blocks to use the high supply voltage (V1)
 2: for all SRAM groups that fail testing do
 3:     Mark group as failing
 4: end for
 5: Assign all remaining blocks to use the intermediate high supply voltage (V2)
 6: for all SRAM groups that fail testing do
 7:     Permanently assign group to the high supply voltage (V1)
 8: end for
 9: Assign all remaining blocks to use the intermediate low supply voltage (V3)
10: for all SRAM groups that fail testing do
11:     Permanently assign group to the intermediate high supply voltage (V2)
12: end for
13: Assign all remaining blocks to use the low supply voltage (V4)
14: for all SRAM groups that fail testing do
15:     Permanently assign group to the intermediate low supply voltage (V3)
16: end for
17: for all SRAM groups that pass testing do
18:     Permanently assign group to low supply voltage (V4)
19: end for
```

assigned to the high supply voltage. The remaining memory cells that passed at the intermediate high supply voltage are then tested at the two other voltage levels using this same procedure.

4.3 Experiments

We performed HSPICE simulations for the SRAM from Figure 4.1 using realistic 45nm process data. We created 10,000 instances of the SRAM cell, and each SRAM instantiation had random device parameters.

We measured the resulting amount of bitline development for each instance across a range of supply voltages. With a 1.0 V supply voltage, 13 SRAM instances had less than the required 200 mV of bitline development. The SRAM instance with the weakest bitline development required 1.08 V in order to meet the sense amp requirement. From a power perspective, though, when using a nominal 1.0 V supply voltage, the average SRAM instance produced 110 mV more bitline development than needed. As shown in (4.4), this 110 mV of excessive bitline development results in 55% more bitline energy than is needed by the sense amp.

Next, we created a 1 KB SRAM memory array by selecting SRAM cells with uniform probability from the set of 10,000 instances. The array had 256 columns, and each column contained 32 SRAM cells. Each bitline column had 20 fF of interconnect plus diffusion capacitance.

After the memory array was constructed, we applied the voltage configuration algorithm from the previous section. Each column's bitline was pre-charged

Table 4.1: Array energy

	Average Energy (fJ)		
	32 SRAMs per group	16 SRAMs per group	8 SRAMs per group
configurable supply (4)	1480	1422	1373
configurable supply (∞)	1412	1356	1294
fixed supply	1758	1758	1758

to 1 V. We then determined the amount of development for each SRAM instance in a particular column. The final voltage of the bitline was the average amount of development for all 32 SRAM instances in a column. The average development was then used to measure the energy for each bitline using (4.4). This process was repeated for all 256 columns in the array.

4.3.1 Results

Table 4.1 shows the average array energy for varying supply voltages and varying SRAM configuration group sizes. The first line is our work where each SRAM group can be assigned to four discrete voltage supply values (0.92 V, 0.96 V, 1.00 V, and 1.08 V). The second line is our work where each SRAM group can be assigned to an arbitrary supply voltage; this gives an upper bound on energy savings with our approach. The last line of the table is a conventional design that uses a fixed supply voltage of 1.08 V for all of the SRAMs cells. We also tried varying the size of the SRAM configurable group. The first column has all 32 SRAMs in a bitline column connected to the same supply voltage. The second column has 16 SRAMs in a bitline column connected to the same supply voltage. For the 16 SRAMs per

group case, each bitline column would need to have two sets of configuration fuses and pmos pass transistors. Similarly, 8 SRAMs per group would need to have four sets of configurations fuses and pmos pass transistors.

4.3.2 Key Takeaways

Our adaptive SRAM shows significant advantages compared to a fixed voltage supply design.

- 32 SRAM cells per group with 4 configurable supplies uses 15.2% less energy than a fixed supply design. 16 SRAM cells per group uses 19.1% less energy than a fixed supply, and 8 SRAM cells per group uses 22.5% less energy than a fixed supply. As expected, smaller configuration groups allow for greater energy savings, since the supply voltage is determined by the slowest cell in the group.
- The ability to set group voltages arbitrarily resulted in roughly 5% additional savings compared to 4 configurable supplies. Therefore, the incremental energy savings from trying to use additional supplies would probably be offset by the area overhead associated with more supplies.
- The highest configurable supply voltage could be set to a value larger than the fixed supply voltage because this highest configurable supply is not used to power the entire SRAM array. This could result in greater yield by enabling correct operation from even very slow SRAM cells that are greater than 5σ from nominal.
- While we only presented data for active energy, reducing the supply voltage

should also allow corresponding savings in leakage energy.

Overall, an adaptive SRAM is effective in reducing SRAM array energy.

4.4 Related Work

In addition to finding the right balance for transistor sizing and cell area to achieve the design target, many proposals have addressed the minimum supply voltage requirement and parametric yield loss due to SRAM failure. These proposals can be generally characterized into the following four categories: (1) SRAM cell modification, (2) voltage islands, (3) body and well biasing, and (4) circuit techniques.

As described in Section 4.1, the complex interactions between the SRAM cell parameters, which affect both reading and writing, may create optimization difficulties. Voltage islands have been proposed as a method by which to decouple the cache memory voltage supply from the logic voltage supply, resulting in lower overall voltage supply levels [48][64]. Since voltage has a quadratic relationship to power, any voltage supply reduction has a large effect on both active and leakage power. Voltage islands also enable additional power-saving operating modes, such as standby and sleep. The limitation of voltage islands is that they require multiple power supplies and additional physical design resources. The disadvantage of this approach is that the memory supply still has to support the weakest cell in the entire SRAM array.

Mukhopadhyay *et al.* [40] used nmos body bias and pmos well bias to shift

the threshold voltage higher or lower based on the overall speed of a particular die. A ring oscillator is used to measure the speed of a particular die. Then, a corresponding amount of biasing is applied. The main purpose of Mukhopadhyay's work was to apply body bias to reduce the number of parametric failures due to random doping fluctuation. Lowering V_{th} affects read and hold failures, while raising V_{th} affects access and write failures. Because this approach shifts all nmos transistor threshold voltages the same way, it is not very effective in addressing SNM.

Yabuuchi *et al.* [62] propose special read and write assist circuits to improve SRAM operation. During SRAM read, voltage dividers are used to reduce the wordline voltage, which increases static noise margin. The wordline voltage is reduced by adding contention, which increases power. During SRAM write, dummy bitline capacitance is added to reduce V_{ddmem}, which also increases the write margin. Also, the V_{ddmem} reduction is a result of charge sharing between the V_{ddmem} column and the dummy metal capacitance. In terms of power, the dummy metal capacitance needs to be discharged after each access. Besides the power considerations with Yabuuchi's approach, process variation makes it difficult to balance capacitance in order to reliably reduce V_{ddmem}.

Yamaoka *et al.* [63] increase the write margin by floating V_{ddmem} during writes. This approach works well for low frequencies, but it is limited to only helping the write margin due to the fact that the V_{ddmem} capacitance is comparably big and the discharge path has to go through the pull up transistor *PU* of the 6T cell, which is a very small device.

4.5 Conclusion

We proposed a SRAM architecture that allows us to compensate for variability by powering up devices that, after manufacturing, are slower than designed. Our approach avoids the need for guard banding in the design phase and leads to savings of up to 22% in bitline energy with moderate area overhead.

In the future, we will investigate more efficient ways to implement supply voltage configuration. For instance, floating gate power transistors might allow us to merge the power transistor with the configuration bit. We will also apply the adaptivity principle to overcoming variation in other building blocks such as clock trees.

Chapter 5

TuneInterconnect

5.1 Introduction

As clock frequencies have sped up, it has become necessary to add repeaters to interconnect wires in order to meet delay targets [3]. Then, as the number of transistors on a die increases, the number of repeaters increases as well. Designs might need hundreds of thousands of repeaters [50]. In some cases, repeaters have been predicted to make up 35% of the total cell count for logic blocks in 45 nm [51].

While many synthesis tools will automatically add interconnect repeaters to a design, there are some designs that add repeaters during the design planning phase. For instance, the Itanium processor required repeaters on 85% of its global routes [38]. As a consequence of so many repeaters, the Itanium added dedicated repeater stations to the floorplan. These repeater stations have continued on more recent Intel processors such as the Atom [25].

5.2 Dual Voltage Repeater

Figure 5.1 shows the dual voltage repeater used for TuneInterconnect. This is the same block as in TuneLogic, except that the logic is fixed as a repeater. Also, since there is only one logic block, TuneInterconnect does not use a level converter

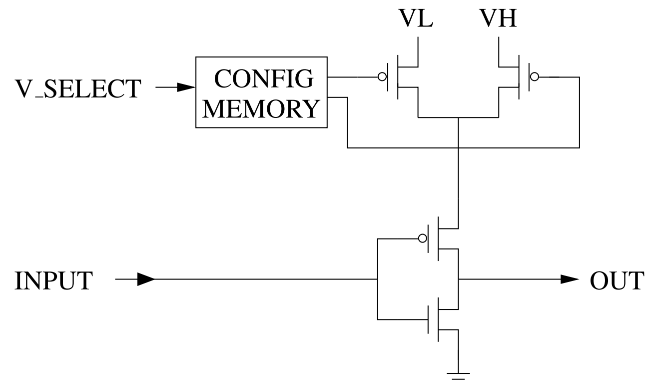

Figure 5.1: Tunable dual-Vdd interconnect repeater with configurable voltage supply PMOS transistors. VH is connected to the high supply voltage grid, and VL is connected to the low supply voltage grid.

in the logic block. For interconnect paths that use a low-Vdd supply, a level converter would be incorporated into the flop at the end of the path.

5.3 Interconnect Characterization

The $\pi 3$ model [47] in Figure 5.2 was used to model the interconnect wire. In order to estimate a reasonable wire length for a long wire, we determined from a die photo of a recent Itanium chip that each core, excluding cache, is about 6.7 mm by 8.6 mm in size [54]. Thus, for this high performance processor, 4 mm to 5 mm would be a reasonable length for a long interconnect. We decided to model a wire of 10 mm in order to determine the benefits of a wire that had even more variation than the estimated 5 mm. Next, k repeaters were inserted into a 10 mm wire, as seen in Figure 5.3. Each segment was of identical length, and each segment had its own repeater. It was determined that using inverters for repeaters results in a faster

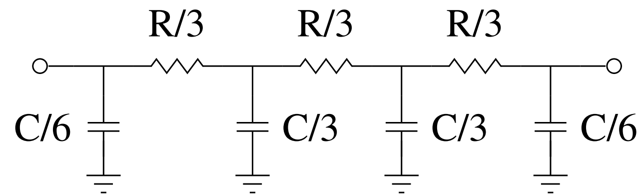

Figure 5.2: 3 segment PI model used for wire simulations

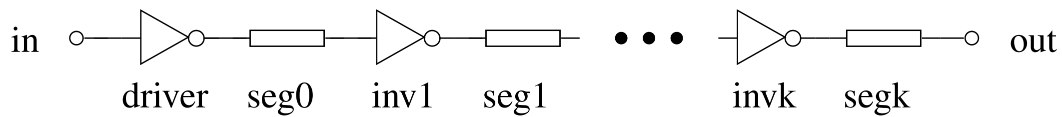

Figure 5.3: k segment wire with k repeaters

interconnect versus using buffers for repeaters. Figure 5.4 shows the interconnect delay for 0 to 20 repeaters with a Vdd of 0.8 V. At each point, the size of the repeater was varied from 10× nominal up to 100× nominal. For this work, the nominal sized inverter was a 0.319 μm width pmos and a 0.150 μm width nmos. The minimum delay for a fixed Vdd of 0.8 V was 1215 ps using 8 repeaters that were sized at 65× nominal. For TuneInterconnect, each power transistor was sized to be 4× the size of the inverter pmos. For a low-Vdd of 0.8 V, TuneInterconnect has a minimum delay of 1263 ps using 8 repeaters that were sized at 70× nominal with power transistors that were sized at 280× nominal. Even using such an extremely large power transistor, TuneInterconnect still had a 4% delay penalty compared to a fixed voltage design.

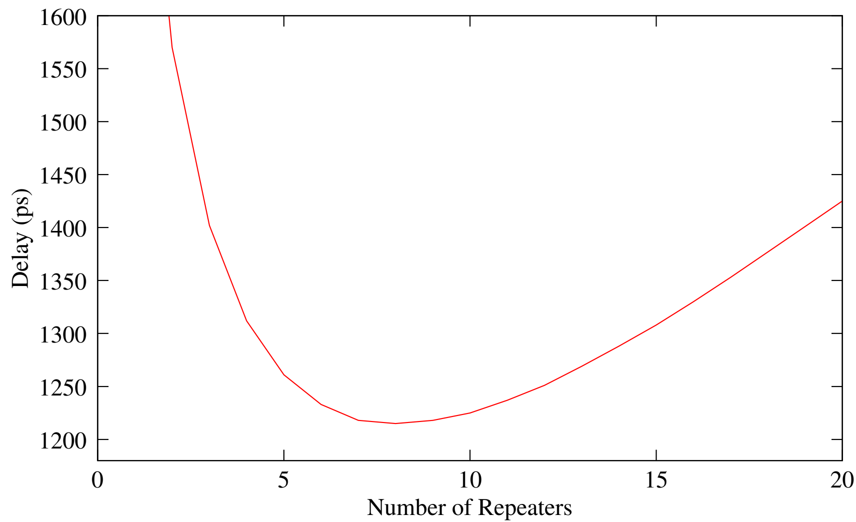

Figure 5.4: Delay for a 10 mm interconnect using various number of repeaters

5.3.1 Methodology

We performed Monte Carlo simulations on the entire interconnect. The wire parameters were set to 449 Ω / mm and 169 fF / mm [65]. For interconnect wires, the resistance is inversely proportional to the capacitance [22]. For instance, the best resistance wire would be a larger wire that would also have the worst capacitance. Conversely, the worst resistance wire would be a smaller wire that would also have the best capacitance. For this work, each wire segment varied the resistance and capacitance the same amount but in opposite directions. The total amount of wire variation was 20% at $3\sigma/\mu$ [22, 59, 33].

As in previous chapters, the 65 nm Berkeley PTM model [12] was used for the SPICE transistor models. For each transistor in the interconnect, both the transistor length and the threshold voltage were modeled as uncorrelated Gaussian

distributions with the $3\sigma/\mu$ variation chosen to be 4%. In previous chapters, the transistor variation was 20%, but the amount of variation is inversely proportional to the square root of the active area [55]. Because the interconnect used predominantly large devices, there should be much less variation in the transistor parameters.

The high voltage supply was set to 1.2 V and the low voltage supply was set to 0.8 V. Since each wire segment was almost 1 mm long, we decided to vary the voltage supply by 10% at $3\sigma/\mu$ in order to model the effects of IR drop in the power grid. The simulation temperature was set to 85 °C.

The delay was measured from the 50% crossing of the input to the 50% crossing of the output for the 10 mm wire shown in Figure 5.3. In order to show the best possible delay, the low-Vdd simulations used 8 repeaters while the high-Vdd simulations used 10 repeaters. The energy is measured from the beginning of the primary input switching until well after the output finishes switching. The leakage energy is then subtracted out of the switching energy in order to present only the active portion of energy. The reported delay and energy values are the average of the rise and fall measurements.

5.3.2 Results

The variation results for two fixed voltages with no power transistors are seen in Figures 5.5 and 5.6. There are some important differences compared to the variations seen in previous chapters. First, the amount of delay variation for low-Vdd is 11 ps, which is much less than was seen in previous chapters. Second, because this interconnect has such a large amount of capacitance, it uses more energy

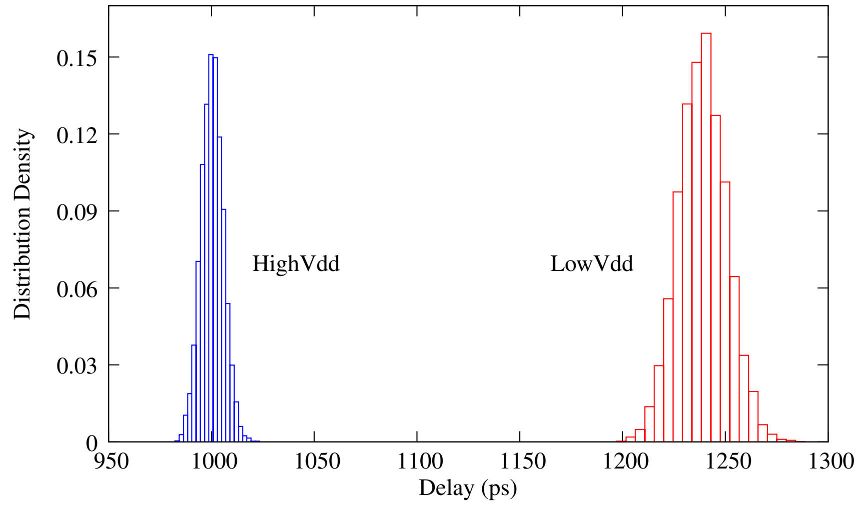

Figure 5.5: Delay distribution for a 10 mm interconnect for two fixed voltages

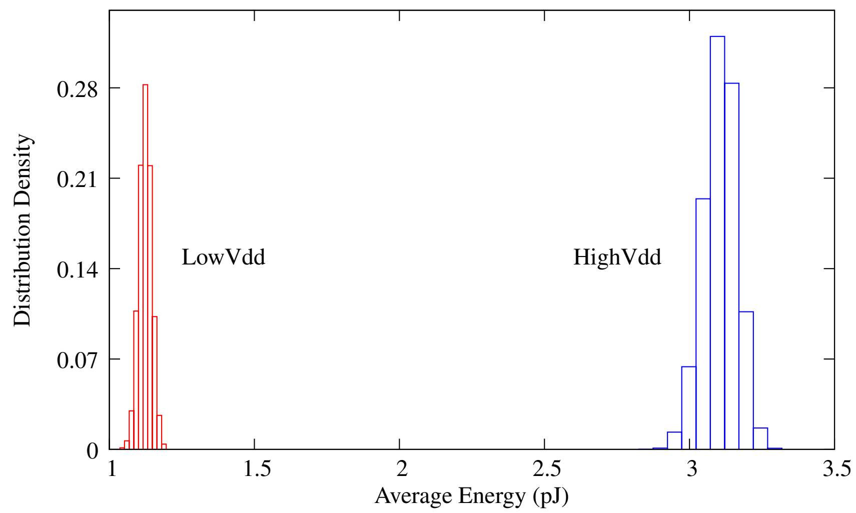

Figure 5.6: Energy distribution for a 10 mm interconnect for two fixed voltages

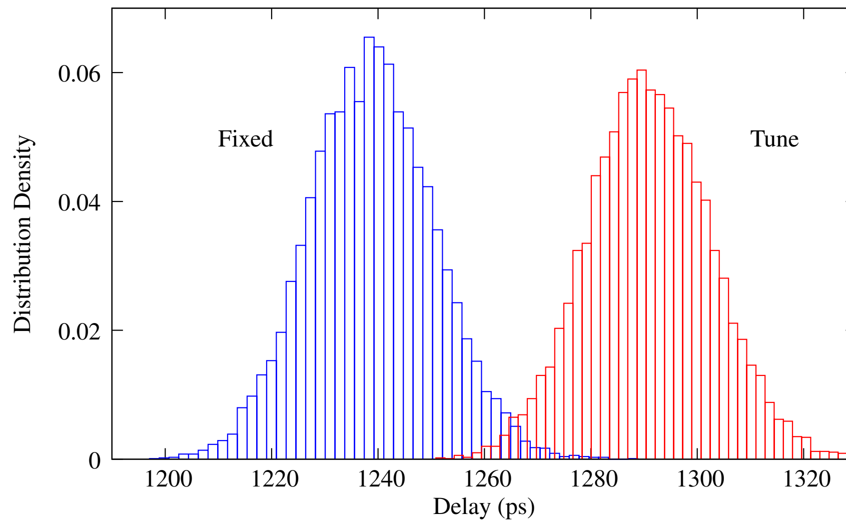

Figure 5.7: Delay distribution for a 10 mm interconnect comparing a fixed voltage design to TuneInterconnect. The bars on the left are a fixed voltage design at 0.8 V, and the bars on the right are a TuneInterconnect design at 0.8 V.

than a TuneLogic block. More importantly though, for each wire segment, there is only one logic path, so the the interconnect repeater is driving all of the capacitance at the same time. In TuneLogic, there were multiple inputs, multiple logic paths, and multiple stages through the block, all of which resulted in TuneLogic being able to spread its current requirements more evenly throughout the cycle. For each TuneInterconnect wire segment, the single repeater is driving all of the capacitance at once, and this results in much higher peak current demands. From a physical design point of view, this higher peak current translates into a larger power transistor than would be needed for a more moderate current requirement.

For a fixed low-Vdd design, the mean delay was 1238 ps and the standard deviation was 11 ps. For the same low-Vdd using TuneInterconnect with power

transistors, the mean delay was 1291 ps with a standard deviation of, again, 11 ps. Figure 5.7 shows that there is very little overlap between a fixed voltage design and TuneInterconnect.

Equation 5.1 shows the minimum propagation delay through an interconnect using the Elmore delay [47].

$$t_{p,min} = \left(1.38 + 1.02\sqrt{1+\gamma}\right)\sqrt{\mathrm{R}_{driver}\mathrm{C}_{driver}\mathrm{R}_{wire}\mathrm{C}_{wire}} \tag{5.1}$$

In this equation, C_{driver} is the driver input capacitance and γ is the ratio of the repeater intrinsic output capacitance to the repeater input capacitance. Because the wire resistance and capacitance were modeled as being inversely proportional, (5.1) predicts that the interconnect-only variation would be offset by changes in resistance R_{wire} and corresponding inverse changes in capacitance C_{wire}.

In order to better account for the cause of variations, the Monte Carlo simulations were run for the following combinations of variation sources: (1) wire variation, (2) wire variation and power supply variation (IR drop), and (3) wire variation, power supply variation, and transistor variation. Table 5.1 shows the interconnect delay statistics for both a fixed voltage design and TuneInterconnect. Since the wire resistance and capacitance were modeled as being inversely proportional, it was expected that in the wire-only variation, the RC delay product would be equally offset between changes in resistance and changes in capacitance. Therefore, the 2 ps standard deviation of wire-only variation would be attributable to the variation in repeater delay due to changes in the capacitance load. For the wire and IR drop variation case, the mean delay is slower than the wire-only variation. This

Table 5.1: Interconnect delay statistics for different combinations of variation sources. The top part of the table is for a fixed voltage design with no power transistor. The bottom part of the table is for TuneInterconnect with a power transistor.

	Low Vdd Delay (ps)				High Vdd Delay (ps)			
Variation Source	Mean	Std	Max	Min	Mean	Std	Max	Min
Fixed Voltage								
Wire	1213	2	1221	1201	991	2	998	982
Wire and IR Drop	1239	7	1274	1214	1000	3	1013	989
Wire, IR, Transistor	1238	11	1288	1197	1000	5	1023	982
TuneInterconnect								
Wire	1264	2	1272	1251	1028	1	1034	1019
Wire and IR Drop	1291	7	1326	1265	1038	3	1052	1027
Wire, IR, Transistor	1291	11	1336	1250	1038	5	1060	1019

is to be expected because IR drop reduces the effective voltage for each repeater, and a reduced voltage would lead to a slower repeater. Even adding in the transistor variation did not substantially increase the delay variation because the large transistors in the repeaters do not vary as much as the typically smaller transistors used in general logic.

5.4 Conclusion

In its present form, TuneInterconnect is not competitive with a fixed voltage design. Because interconnects consist of so much capacitance switching simultaneously, the current demands require an unacceptably large power transistor. If a future process technology were to develop a better power transistor that could switch large amount of currents in a reasonable amount of area, then TuneInterconnect

should be able to provide some power savings.

Still, TuneInterconnect could be beneficial when there are even more variation sources than those presented in this chapter. For instance, crosstalk effects might drastically slow down a signal path. Since crosstalk has a temporal and data dependency, it can be difficult to accurately model the actual amount of crosstalk for a design. If a crosstalk path was missed in the design phase, TuneInterconnect could be used to adjust the delays in a failing design after manufacturing.

Chapter 6

Conclusion

6.1 Conclusion

TuneChip has been shown to increase yield while at the same time decreasing power. TuneChip is able to respond to individual variations specific to each manufactured chip by adapting to variations on multiple different levels. If only small areas of the chip do not meet their design targets, TuneChip can make adjustments just to those areas. Additional, if large sections of the chips are failing, TuneChip can accommodate those large areas as well.

Equally important, TuneChip can be applied to many different areas of a design. Most broadly, TuneChip can be applied to general logic gates by partitioning those gates into blocks. For more particular instances, TuneChip can be configured for use in dedicated blocks such as CLBs in FPGAs and cache SRAM cells found in almost all designs.

TuneChip has been focused on the design aspect of post-silicon tuning because testing, while an important area for future work, is another dissertation topic.

6.2 Future Work

An area for future work is determining what constitutes good voltage supply values. This work used 400 mV as the separation between high voltage and low voltage in order to clearly show the benefit of having two supply voltages. For other combinations of specific designs, associated process technologies, and particular variation sources, though, a different value for voltage separation would most likely yield better results. In addition to separation, setting the voltage supply level would be another area for further exploration. For instance, if a design sought to improve yield, it could be beneficial to set the high voltage supply to a value greater than 1.2 V in order to improve the delay of critical paths. TuneChip could be applied to reduce the power consumption of the resulting faster chip.

An orthogonal area for future work includes investigating the benefits of adding configurability to other transistor parameters, such as threshold voltage or transistor width. For instance, it would be interesting to be able to choose between a low threshold voltage and a nominal threshold voltage transistor after manufacturing. One implementation could arrange a low threshold voltage block in parallel with a high threshold voltage block, with the power transistor selecting which threshold block to use. While having two copies of each block across an entire chip might not be practical, in areas of a design that are interconnect limited, there might be unused die area that could accommodate an additional block with little overhead.

Appendices

Appendix A

Voltage Assignment NP-Complete Proof

We show the Voltage Assignment Problem (VAP) is NP-complete by a reduction from the Weighted Knapsack Problem. First we formalize VAP as follows.

VAP Instance: directed acyclic graph $G = (V, E)$. Source vertices will be referred to as primary inputs, and the set of primary inputs will denoted by PI; sink vertices will be referred to as primary outputs, and the set of primary outputs will be denoted by PO. Internal vertices will be denoted by G; the sets PI, PO, G partition V.

Each $g \in G$ has a low-delay l_g, a high-delay h_g, a low-delay cost c^l_g, and a high-delay cost c^h_g. An assignment α is a Boolean valued function whose domain is G—intuitively, $\alpha(g) = 1$ means that g is assigned high-delay; $\alpha(g) = 0$ means that g is low-delay. Each VAP instance has a cost-bound C and a delay target Δ. All quantities—delays, costs, and bounds—are assumed to be nonnegative integers.

Question: does there exist an assignment function α such that the circuit achieves the delay target Δ, with a cost of no more than C? Formally, does there exist $\alpha \in [G \mapsto \{0, 1\}]$ [1] such that

$$\sum_{g \in G} \Big(\alpha(g) \cdot c^h_g \,+\, (1 - \alpha(g)) \cdot c^l_g \Big) \;\leq\; C$$

[1]Recall that $[A \mapsto B]$ is the set of all functions from A to B.

and for all paths π from PI to PO

$$\sum_{g \in \pi} \Big(\alpha(g) \cdot h_g \; + \; (1 - \alpha(g)) \cdot l_g \Big) \;\; \leq \;\; \Delta?$$

Theorem: VAP is NP-complete.

Proof: Membership of VAP in NP is straightforward, since a candidate assignment α can be checked in polynomial time—the cost constraint is trivial to check, and the timing constraint can be checked by using longest path computation in a DAG, which is known to be polynomial time.

We show that VAP is NP-hard by reduction from the weighted-knapsack problem (WKP), which is known to be NP-complete [24].

WKP Instance: a set of n objects, with values $\{v_1, v_2, \ldots, v_n\}$ and weights $\{w_1, w_2, \ldots, w_n\}$ respectively; weight constraint W; value constraint V.

Question: does there exist a subset of objects $S \subseteq \{1, 2, \ldots, n\}$ such that $\sum_{i \in S} w_i \leq W$ and $\sum_{i \in S} v_i \geq V$?

We reduce WKP to VAP as follows: for any instance of WKP, we create a DAG with n gates $g_1, g_2, \ldots, g_n$, one primary input vertex p, and one primary output vertex q. The edge set consists of $\{(p, g_1)\} \cup \{(g_i, g_{i+1}) : 1 \leq i \leq n-1\} \cup \{(g_n, q)\}$.

Let $\sigma = \sum_{i=1}^{n} v_i$. Take the high-delay of gate i to be σ, and the low delay to be $\sigma - v_i$. The high-delay cost of i is 0, and the low-delay cost of i is w_i.

We claim that there exists a solution to the VAP problem with cost-bound W and delay target $n \cdot \sigma - V$ iff there exists a solution to the original WKP instance.

Proof—first suppose there exists a solution to the WKP, say the subset S^*. Then consider the assignment $\alpha(i) = 0$ iff $i \in S^*$.

This assignment satisfies the cost constraint on VAP, since the gates set to low delay are in one-to-one correspondence with the objects in S^*, and the net cost for both is $\sum_{i \in S^*} w_i$ which, since S^* is a solution to WKP, must be no more than W.

Furthermore, the sum of the values of the objects in S^* is at least V, so the sum of the delays of the gates in G is $\sum_{i \notin S^*} \sigma + \sum_{i \in S^*} (\sigma - v_i)$, which is equal to $n \cdot \sigma - \sum_{i \in S^*} v_i$. Since S^* is a solution for WKP, it must be that $\sum_{i \in S^*} v_i \geq V$, hence the sum of the delays of the gates in G is no more than $n \cdot \sigma - V$, i.e., the delay constraint holds.

Hence given a solution to WKP, we can find a solution to the corresponding instance of VAP.

For the converse, consider any assignment α for the VAP instance derived from a WKP instance such that α satisfies the cost-bound and delay target. Let $S^* = \{i : \alpha(g_i) = 0\}$; then we can show in a manner symmetric to the above that S^* satisfies the knapsack constraints.

The reduction is clearly polynomial (in fact, it is linear), hence VAP is NP-complete.

Appendix B

Bitline Energy Derivation

We used a simple RC circuit, shown in Figure 4.4, to derive the total energy expended in pre-charging and evaluating the bitlines. Since only one bitline is evaluated during a cycle, C is the total capacitance of one bitline. The resistance R is the equivalent resistance of the additional circuit elements that appear between the voltage supply and the bitline. Since this bitline does not fully discharge to 0 V, the energy is calculated for a bitline that starts at voltage V_1 and rises to voltage V_2. The total energy is derived by adding the energy through the capacitor to the energy through the resistor using the following equations:

$$v_{cap}(t) = (V_2 - V_1)\left(1 - e^{-t/RC}\right) + V_1 \tag{B.1}$$

$$i_{cap}(t) = \frac{(V_2 - V_1)}{R}\left(e^{-t/RC}\right) \tag{B.2}$$

$$\text{Energy}_{cap} = \int_0^{\infty} v_{cap}(t)\, i_{cap}(t)\, dt = \frac{1}{2}\, C\, (V_2 - V_1)(V_2 + V_1) \tag{B.3}$$

Notice that (B.1) is the normal capacitor voltage equation except that the voltage range has been shifted to V_2 through V_1 (instead of going all the way to 0). When solving for the energy for the resistor, the voltage supply is set to V_2. Additionally, $i_{res}(t) = i_{cap}(t)$.

$$v_{res}(t) = V_2 - v_{cap}(t) \tag{B.4}$$

$$\text{Energy}_{res} = \int_0^\infty v_{res}(t)\ i_{res}(t)\ dt = \frac{1}{2}\ \text{C}\ (\text{V}_1 - \text{V}_2)^2 \tag{B.5}$$

It is interesting to note that the energy through the resistor is independent of the size of the resistor. And now the total energy:

$$\text{Energy}_{total} = \text{Energy}_{res} + \text{Energy}_{cap} = \text{C}\ (\text{V}_2 - \text{V}_1)\ \text{V}_2 \tag{B.6}$$

Lastly, for this work V_2 is V_{dd} and $(\text{V}_2 - \text{V}_1)$ is V_{bl}, which results in:

$$\text{Energy}_{bitline} = \text{C}\ \text{V}_{bl}\ \text{V}_{dd} \tag{B.7}$$

Bibliography

[1] R. Adams, *High performance memory testing*. Kluwer Academic Boston, 2003.

[2] K. Agarwal and K. Nowka, “Dynamic Power Management by Combination of Dual Static Supply Voltages,” *International Symposium on Quality Electronic Design*, 2007.

[3] C. Alpert and A. Devgan, “Wire Segmenting for Improved Buffer Insertion,” *Design Automation Conference*, 1997.

[4] *”Stratix III FPGAs vs. Xilinx Virtex-5 Devices: Architecture and Performance Comparison”*, Altera Corporation, 2006.

[5] *”Stratix III Device Handbook”*, Altera Corporation, 2007.

[6] AMD, “ATI Radeon HD 3800 Series - GPU Specifications,” http://ati.amd.com/products/radeonhd3800/specs.html, 2008.

[7] A. Asenov, S. Kaya, and J. Davies, “Intrinsic threshold voltage fluctuations in decanano MOSFETs due to local oxide thickness variations,” *IEEE Transactions on Electron Devices*, 2002.

[8] S. Bijansky and A. Aziz, “TuneFPGA: post-silicon tuning of dual-Vdd FPGAs,” *Design Automation Conference*, 2008.

[9] S. Bijansky, S. Lee, and A. Aziz, “Adaptive Voltage Tuning for Dual-Vdd ASICs,” *Austin Conference on Integrated Systems and Circuits*, 2008.

[10] S. Bijansky, S. Lee, and A. Aziz, “TuneLogic: Post-Silicon Tuning of Dual-Vdd Designs,” *International Symposium on Quality Electronic Design*, 2009.

[11] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, “Parameter Variations and Impact on Circuits and Microarchitecture,” *Design Automation Conference*, 2003.

[12] Y. Cao, T. Sato, M. Orshansky, D. Sylvester, and C. Hu, “New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Simulation,” *IEEE Custom Integrated Circuits Conference*, 2000.

[13] H. Chang and S. S. Sapatnekar, “Prediction of leakage power under process uncertainties,” *ACM Transactions on Design Automation of Electronic Systems*, 2007.

[14] C. Chen, C. Chu, and D. Wong, “Fast and Exact Simultaneous Gate and Wire Sizing by Lagrangian Relaxation,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1999.

[15] T. Chen and S. Naffziger, “Comparison of adaptive body bias (ABB) and adaptive supply voltage (ASV) for improving delay and leakage under the presence of process variation,” *IEEE Transactions on Very Large Scale Integration Systems*, 2003.

[16] J. Cong, Y. Ding, and J. Peck, "RASP: A General Logic Synthesis System for SRAM-Based FPGAs," *ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, 1996.

[17] J. Danson, C. Plett, and N. Tait, "Design and characterization of a MEMS capacitive switch for improved RF amplifier circuits," *IEEE Custom Integrated Circuits Conference*, 2005.

[18] S. Dhar, D. Maksimović, and B. Kranzen, "Closed-loop adaptive voltage scaling controller for standard-cell ASICs," *International Symposium on Low Power Electronics and Design*, 2002.

[19] J. Dorsey, S. Searles, M. Ciraula, S. Johnson, N. Bujanos, D. Wu, M. Braganza, S. Meyers, E. Fang, and R. Kumar, "An Integrated Quad-Core Opteron Processor," *IEEE International Solid-State Circuits Conference*, 2007.

[20] M. Elgebaly and M. Sachdev, "Efficient Adaptive Voltage Scaling System Through On-Chip Critical Path Emulation," *International Symposium on Low Power Electronics and Design*, 2004.

[21] D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, N. Kim, and K. Flautner, "Razor: Circuit-Level Correction of Timing Errors for Low-Power Operation," *IEEE Micro*, 2004.

[22] T. Fukuoka, A. Tsuchiya, and H. Onodera, "Worst-case delay analysis considering the variability of transistors and interconnects," *ACM International Symposium on Physical Design*, 2007.

[23] H. Fukutome, T. Aoyama, Y. Momiyama, T. Kubo, Y. Tagawa, and H. Arimoto, "Direct evaluation of gate line edge roughness impact on extension profiles in sub-50nm N-MOSFETs," *IEEE International Electron Devices Meeting*, 2004.

[24] M. R. Garey and D. S. Johnson, *Computers and Intractability*. W. H. Freeman and Co., 1979.

[25] G. Gerosa, S. Curtis, M. D'Addeo, B. Jiang, B. Kuttanna, F. Merchant, B. Patel, M. Taufique, and H. Samarchi, "A Sub-1W to 2W Low-Power IA Processor for Mobile Internet Devices and Ultra-Mobile PCs in 45nm Hi-K Metal Gate CMOS," *IEEE International Solid-State Circuits Conference*, 2008.

[26] A. Hoefler, C. Henson, C.-N. Li, and D.-G. Lin, "Analysis of a novel electrically programmable active fuse for advanced cmos soi one-time programmable memory applications," *Solid-State Device Research Conference*, 2006.

[27] K. Katsuki, M. Kotani, K. Kobayashi, and H. Onodera, "A Yield and Speed Enhancement Scheme under Within-Die Variations on 90nm LUT Array," *IEEE Custom Integrated Circuits Conference*, 2005.

[28] K. Kuhn, "Reducing variation in advanced logic technologies: Approaches to process and design for manufacturability of nanoscale cmos," *IEEE International Electron Devices Meeting*, 2007.

[29] S. Kulkarni and D. Sylvester, "High performance level conversion for Dual Vdd design," *IEEE Transactions on Very Large Scale Integration Systems*,

2004.

[30] I. Kuon and J. Rose, “Measuring the Gap Between FPGAs and ASICs,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, 2007.

[31] F. Li, Y. Lin, and L. He, “Field Programmability of Supply Voltages for FPGA Power Reduction,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2007.

[32] X. Liang, G. Wei, and D. Brooks, “ReVIVaL: A variation tolerant architecture using voltage interpolation and variable latency,” *International Symposium on Computer Architecture*, 2008.

[33] X. Lin, “Design for Manufacturing with Increasing Variability,” *International Conference On ASIC*, 2005.

[34] Y. Lin, F. Li, and L. He, “Circuits and Architectures for Field Programmable Gate Array with Configurable Supply Voltage,” *IEEE Transactions on Very Large Scale Integration Systems*, 2005.

[35] H. Mahmoodi, S. Mukhopadhyay, and K. Roy, “Estimation of delay variations due to random-dopant fluctuations in nanoscale CMOS circuits,” *IEEE Journal of Solid-State Circuits*, 2005.

[36] J. Massey, “NBTI: What we know and what we need to know—A tutorial addressing the current understanding and challenges for the future,” *IEEE International Integrated Reliability Workshop*, 2004.

[37] Y. Matsumoto, M. Hioki, T. Kawanami, T. Tsutsumi, T. Nakagawa, T. Sekigawa, and H. Koike, "Performance and Yield Enhancement of FPGAs with Within-Die Variation using Multiple Configurations," *ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, 2007.

[38] R. McInerney, K. Leeper, T. Hill, H. Chan, B. Basaran, and L. McQuiddy, "Methodology for repeater insertion management in the RTL, layout, floorplan and fullchip timing databases of the Itanium microprocessor," *ACM International Symposium on Physical Design*, 2000.

[39] B. Mohammad, S. Bijansky, A. Aziz, and J. Abraham, "Adaptive SRAM-based Memory for Low Power and High Yield," *IEEE International Conference on Computer Design*, 2008.

[40] S. Mukhopadhyay, K. Kim, H. Mahmoodi, and K. Roy, "Design of a Process Variation Tolerant Self-Repairing SRAM for Yield Enhancement in Nanoscaled CMOS," *IEEE Journal of Solid-State Circuits*, 2007.

[41] G. Nabaa, N. Azizi, and F. Najm, "An adaptive FPGA architecture with process variation compensation and reduced leakage," *Design Automation Conference*, 2006.

[42] S. Narendra and A. Chandrakasan, *Leakage in Nanometer CMOS Technologies*. Springer, 2006.

[43] S. Nassif, "Modeling and Analysis of Manufacturing Variations," *IEEE Custom Integrated Circuits Conference*, 2001.

[44] M. Orshansky, S. Nassif, and D. Boning, *Design for Manufacturability and Statistical Design: A Constructive Approach*. Springer, 2008.

[45] K. Parhi, *VLSI Digital Signal Processing Systems*. Wiley New York, 1999.

[46] L. Pileggi, H. Schmit, A. Strojwas, P. Gopalakrishnan, V. Kheterpal, A. Koorapaty, C. Patel, V. Rovner, and K. Tong, "Exploring Regular Fabrics to Optimize the Performance-Cost Trade-Off," *Design Automation Conference*, 2003.

[47] J. Rabaey, A. Chandrakasan, and B. Nikolić, *Digital Integrated Circuits: A Design Perspective*, 2003.

[48] P. Royannez, H. Dahan, F. Wagner, M. Streeter, M. Bouetel, L. Blasquez, J. Clasen, H. Semino, G. Scott, D. Pitts *et al.*, "90nm low leakage SoC design techniques for wireless applications," *IEEE International Solid-State Circuits Conference*, 2005.

[49] S. Rusu, S. Tam, H. Muljono, D. Ayers, J. Chang, B. Cherkauer, J. Stinson, J. Benoit, R. Varada, J. Leung *et al.*, "A 65-nm Dual-Core Multithreaded Xeon Processor With 16-MB L3 Cache," *IEEE Journal of Solid-State Circuits*, 2007.

[50] P. Saxena, "The scaling of interconnect buffer needs," *International Workshop on System-Level Interconnect Prediction*, 2006.

[51] P. Saxena, N. Menezes, P. Cocchini, and D. Kirkpatrick, "Repeater Scaling and Its Impact on CAD," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2004.

[52] N. Sherwani, *Algorithms for VLSI Physical Design Automation*. Kluwer Academic Publishers Norwell, MA, USA, 1995.

[53] A. Srivastava, S. Shah, K. Agarwal, D. Sylvester, D. Blaauw, and S. Director, "Accurate and Efficient Gate-Level Parametric Yield Estimation Considering Correlated Variations in Leakage Power and Performance," *Design Automation Conference*, 2005.

[54] B. Stackhouse, B. Cherkauer, M. Gowan, P. Gronowski, and C. Lyles, "A 65nm 2-billion-transistor quad-core itanium processor," *IEEE International Solid-State Circuits Conference*, 2008.

[55] P. Stolk, F. Widdershoven, and D. Klaassen, "Modeling statistical dopant fluctuations in MOS transistors," *IEEE Transactions on Electron Devices*, 1998.

[56] C. Stroud, *A Designer's Guide to Built-in Self-Test*. Kluwer Academic Publishers, 2002.

[57] T. Tuan, S. Kao, A. Rahman, S. Das, and S. Trimberger, "A 90nm low-power FPGA for battery-powered applications," *ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, 2006.

[58] O. Unsal, J. Tschanz, K. Bowman, V. De, X. Vera, A. González, and O. Ergin, "Impact of Parameter Variations on Circuits and Microarchitecture," *IEEE Micro*, 2006.

[59] V. Venkatraman and W. Burleson, "Robust Multi-Level Current-Mode On-Chip Interconnect Signaling in the Presence of Process Variations," *International Symposium on Quality Electronic Design*, 2005.

[60] C. Visweswariah, "Death, Taxes and Failing Chips," *Design Automation Conference*, 2003.

[61] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*. Pearson/Addison-Wesley, advance copy, 2009.

[62] M. Yabuuchi, K. Nii, Y. Tsukamoto, S. Ohbayashi, S. Imaoka, H. Makino, Y. Yamagami, S. Ishikura, T. Terano, T. Oashi *et al.*, "A 45nm Low-Standby-Power Embedded SRAM with Improved Immunity Against Process and Temperature Variations," *IEEE International Solid-State Circuits Conference*, 2007.

[63] M. Yamaoka, N. Maeda, Y. Shinozaki, Y. Shimazaki, K. Nii, S. Shimada, K. Yanagisawa, and T. Kawahara, "90-nm Process-Variation Adaptive Embedded SRAM Modules With Power-Line-Floating Write Technique," *IEEE Journal of Solid-State Circuits*, 2006.

[64] K. Zhang, U. Bhattacharya, Z. Chen, F. Hamzaoglu, D. Murray, N. Vallepalli, Y. Wang, B. Zheng, and M. Bohr, "A 3-GHz 70mb SRAM in 65nm CMOS technology with integrated column-based dynamic power supply," *IEEE International Solid-State Circuits Conference*, 2005.

[65] W. Zhao and Y. Cao, "New Generation of Predictive Technology Model for

Sub-45nm Design Exploration," *International Symposium on Quality Electronic Design*, 2006.

Vita

Stephen Bijansky, Jr., attended William Penn High School in New Castle, Delaware. In 1993, he entered the University of Delaware in Newark, Delaware. He was the first engineer in the history of the University to receive an Honors Bachelor of Electrical Engineering in May 1997. He then enrolled at Carnegie Mellon University, where he received a Masters of Science in Electrical and Computer Engineering in December 1998. From 1999 until 2002, he was employed as a storage systems architect at Dell in Round Rock, Texas. In 2002, he entered the University of Texas at Austin. While a graduate student at the University of Texas, he held interships at Magma Design Automation, Pyxis Technology, and Qualcomm. After graduation, he will continue working as a circuit designer at Qualcomm in Austin, Texas.

Permanent address: 2000 Cullen Avenue Unit 12
Austin, Texas 78757

This dissertation was typed by the author.