Circuit Techniques for Subthreshold Leakage Reduction in a Deep Sub-Micron Process

by

Benton Highsmith Calhoun

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2002

© Massachusetts Institute of Technology 2002. All rights reserved.

Author . . . .
Department of Electrical Engineering and Computer Science
May 24, 2002

Certified by . . . .
Anantha P. Chandrakasan
Associate Professor
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Circuit Techniques for Subthreshold Leakage Reduction in a Deep Sub-Micron Process

by

Benton Highsmith Calhoun

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2002, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

The trend of process scaling for CMOS technology has made subthreshold leakage reduction a growing concern for submicron circuit designers. Power consumption has become a principle design consideration as device sizes decrease and many more devices fit on a single chip. Since switching power is proportional to V_{DD}^2 new processes are tailored for lower supply voltages. The decrease in V_{DD} slows down devices which requires that the threshold voltage, V_T, must be lowered to maintain performance. This reduction of V_T produces the exponential increase of subthreshold leakage currents that has become well known.

Field Programmable Gate Arrays (FPGAs) are one type of chip that could benefit from subthreshold leakage reduction techniques. The programmable nature of FPGA designs is amenable to standby mode leakage reduction since some of the logic blocks might be unused. Leakage reduction for programmed blocks would make FPGAs more attractive to the designers of battery-operated devices.

Our research demonstrates circuit techniques to reduce subthreshold leakage for a state-of-the-art 0.13μm process. Although applicable to generic CMOS circuits in this process, the techniques in this work are specifically intended for use in low power FPGA design.

A testchip uses Multi-Threshold CMOS (MTCMOS) style logic design to implement a new type of FPGA architecture for distributed arithmetic. The circuits are designed to provide maximum reduction in standby leakage current without degrading performance of the circuits by more than 10% relative to an all low V_T design. Sections of the FPGA which are not used in a given configuration are automatically placed in a low leakage state. We exhibit the new Leakage Feedback Flip-flop (LFBFF) [1] that retains its state in standby mode. We also introduce supply voltage scaling during standby mode to reduce power consumption while retaining state.

Thesis Supervisor: Anantha P. Chandrakasan
Title: Associate Professor

Acknowledgments

[28]Do you not know?
Have you not heard?
The LORD is the everlasting God,
the Creator of the ends of the earth.
He will not grow tired or weary,
and his understanding no one can fathom.
[29]He gives strength to the weary
and increases the power of the weak.
[30]Even youths grow tired and weary,
and young men stumble and fall;
[31]but those who hope in the LORD
will renew their strength.
They will soar on wings like eagles;
they will run and not grow weary,
they will walk and not be faint.
Isaiah 40:28-31

[10]The fear of the LORD is the
beginning of wisdom,
and knowledge of the Holy One
is understanding.
Proverbs 9:10

I first thank my Lord Jesus Christ for graciously giving me his strength and guidance throughout my life. I owe him everything, and I attribute all praise and glory to him.

I cannot thank Professor Anantha Chandrakasan enough for his role in this project. Thank you, Anantha, for giving me the opportunity to join your group and to tackle this research in the first place. Your ideas, advice, and support made this work possible. I owe a lot of gratitude to Frank Honore and Theodoros Kontantakopoulos for their help on the testchip. Since Frank I and worked on the same part of the chip, his collaboration and his long hours in the lab were incredibly valuable. I surely could not have done it alone. I also thank Cypress Semiconductor for providing us with the tools to build the testchip. Alan Refalo especially gave a lot of his time and energy to support us. Thanks also to everyone else in *ananthagroup* for fun times and technical discussions. I especially thank Travis Simpkins for his advice and for sharing the joys and woes of chip design. I am also grateful to MARCO, Texas Instruments, and Infineon for providing funding at different points during the project.

Finally, I thank my family. My parents have supported me, as always, with their thoughts, generosity, and prayers. I especially thank my wife, Mary Kathryn. She has been so generous and wonderful for the last two years. Mare, I know I could not do this without you. Thanks to you and to Anna Grace for motivating me and for loving me so selflessly. I love you both so much.

Contents

1 Introduction and Background **17**

1.1 Introduction 17

1.2 Motivation 17

1.3 Existing Techniques for Subthreshold Leakage Reduction 20

1.3.1 Switched Source Impedance 20

1.3.2 Stack Effect 21

1.3.3 Body Biasing 22

1.3.4 Dual-Threshold CMOS 22

1.3.5 MTCMOS 23

1.3.6 BGMOS and SCCMOS 24

1.4 Leakage Feedback Flip-flop 24

1.4.1 MTCMOS Sequential Circuits 24

1.4.2 Leakage Feedback Gate 25

1.4.3 LFBFF Design 27

1.5 Contributions of Thesis 28

2 Characterizing Leakage in a State-of-the-Art 0.13μm CMOS Process **29**

2.1 Introduction 29

2.2 Leakage Models 29

2.3 Subthreshold Current Versus V_{GS} 30

2.3.1 Subthreshold Slope 31

2.4 V_{DS} and the DIBL Coefficient 34

2.5 Temperature 36

2.6 Subthreshold Leakage Versus Device Size 37

3 Voltage Scaling During Standby Mode 41

3.1 Introduction 41

3.2 Voltage Scaling Concept 41

3.3 Varying Supply Voltage and the LFBFF 42

3.3.1 Storing a Logic '1' 43

3.3.2 Storing a Logic '0' 47

3.3.3 Power Savings Using Voltage Scaling 52

3.3.4 Levels of Standby 53

3.4 Adaptively Scaling Supply Voltage 55

3.4.1 Architecture for Adaptive Flip-Flop Voltage Scaling 55

3.4.2 Designing Flip-Flops for Failure 56

3.4.3 Effect of Variation on Flip-Flops 58

4 Testchip Design and Architecture 63

4.1 Introduction 63

4.2 Subthreshold Leakage and FPGAs 63

4.3 Test Chip Circuits 64

4.3.1 Top Level Design 65

4.3.2 Core 65

4.3.3 Data Registers 72

4.3.4 Peripheral Components 73

5 Testchip Implementation 79

5.1 Introduction 79

5.2 Multiple Power Regions 79

5.3 Sleep Mode Specifics 81

5.3.1 Multiple Regions for Sleep in a CLB 81

5.3.2 Fine Granularity 87

5.3.3 Sizing Sleep Devices 88

5.3.4 Sleep, FFs, and Clock 88
5.3.5 Energy Overhead for Sleep Mode 89
5.4 Stopping Sneak Leakage Paths 92
5.4.1 Causes of Sneak Leakage Paths 93
5.4.2 Sneak Leakage Paths Eliminated from the Testchip 94
5.4.3 Guidelines for Preventing Sneak Leakage Paths 101
5.5 Layout 102
5.5.1 General Strategy 102
5.5.2 Clock Signals 103
5.5.3 Power Signals 104
5.5.4 Critical Path Implementation 104
5.5.5 Layout Screenshots 106

6 Conclusion 109

6.1 Introduction 109
6.2 Summary and Results 109
6.2.1 Flip-Flop Voltage Scaling 109
6.2.2 Low Power FPGA Testchip 110
6.2.3 CLB Power Savings Using Voltage Scaling 112
6.3 Contributions 113
6.4 Future Work 114
6.4.1 Voltage Scaling 114
6.4.2 Low Power FPGA Testchip 114

List of Figures

1-1 Sub-threshold (SD) Leakage 18
1-2 Trend of Increasing Leakage Power 19
1-3 Leakage Power for Future Chips 19
1-4 Switched-Source-Impedence 20
1-5 Artificial Creation of Stack Effect 22
1-6 MTCMOS Schematic 23
1-7 MTCMOS Balloon Flip-Flop 25
1-8 Leakage Feedback Gate 26
1-9 Leakage Feedback Flip-Flop 27

2-1 Subthreshold Current vs V_{GS} 31
2-2 Model versus Simulation for Varying V_{GS} 32
2-3 Simulation of Subthreshold Leakage versus V_{DS} 34
2-4 Model Versus Simulation for Varying V_{DS} 35
2-5 Simulation of Subthreshold Leakage versus Temperature 36
2-6 Model Versus Simulation for Varying Temperature 36
2-7 Threshold Voltage Variation with Device Size 37
2-8 Threshold Voltage Variation with Device Size 38

3-1 LFBFF in Sleep Mode 42
3-2 LFBFF Holding a ’1’ 44
3-3 Output Node Changing Value for Stored ’1’ 45
3-4 Internal Node Changing Value for Stored ’1’ 46
3-5 LFBFF Storing ’1’ at V_{DD} = 80mV 47

3-6 LFBFF Holding a '0' 48
3-7 High V_T Inverter Output for Varying Widths 48
3-8 Output Node Changing Value for Stored '0' 50
3-9 Internal Node Changing Value for Stored '0' 50
3-10 LFBFF Storing '0' at $V_{DD} = 80$mV 51
3-11 Power Reduction from Voltage Scaling 52
3-12 Power Reduction Factor from Voltage Scaling 53
3-13 Adjustable Voltage Scaling Architecture 55
3-14 LFBFF Storing '1': Sized to Fail at 130mV 57
3-15 LFBFF Storing '0': Sized to Fail at 130mV 58
3-16 Effect of Temperature on FF Failure 59
3-17 Effect of Process Corners on FF Failure 60

4-1 Test Chip Setup 65
4-2 Configurable Logic Block Used on the Test Chip 66
4-3 Slice Diagram 68
4-4 Programmable Transmission Gate for Interconnect 68
4-5 Top Level Switching Diagram 70
4-6 Slices Implementing an 8-tap, 16-bit FIR Filter 71
4-7 Data Register Diagram 72
4-8 Input Generation Hardware and Operation 74
4-9 Timing Diagram for the Capturing Filter Output 76
4-10 Clock Gating System 76

5-1 Sleep Regions within the CLB 82
5-2 Sleep Signal Selection for LUT 82
5-3 Short Circuit Current due to Poor Sleep Region Interface 84
5-4 Steady-State Current Change Entering Sleep 86
5-5 Clock Buffers to Flip-Flops During Sleep Mode 89
5-6 CLB Current and Energy Transitioning to/from Sleep Mode 90
5-7 Break-Even Time for Entering Sleep Mode 91

5-8 Effect of Sneak Leakage Paths on Current Reduction 93
5-9 High V_T Latch Used in LUTs 94
5-10 Sneak Leakage Paths in 16-bit LUT 94
5-11 Final Design for 16-bit LUT 96
5-12 Sneak Leakage Path through Multiplexor 97
5-13 Sneak Leakage Path Eliminated through Multiplexor 97
5-14 Sneak Leakage Path Due to Missing Sleep Device 98
5-15 Subtle Sneak Leakage Path Due to Shared Sleep FETs 99
5-16 Isolating LFBFF Input Removes Sneak Path 100
5-17 Layout Placement for a Slice (with 4 CLBs) 105
5-18 Layout of Full Adder 106
5-19 Layout of CLB 107
5-20 Layout of Chip 108

6-1 Leakage Current Reduction Entering Sleep 111

List of Tables

3.1 LFBFF State for Stored '1' 43

3.2 LFBFF State for Stored '0' 47

3.3 Possible Set of Standby States 54

4.1 Signals Multiplexed by Configuration Bits 67

4.2 Mapping of Slice Signals to Buses 69

4.3 Shift Registers on the Testchip 73

5.1 Power Supply Regions 80

5.2 Configuration Bit Constraints 85

5.3 Steady-State Current Reduction Using Sleep Regions 85

6.1 Breakdown of CLB Power 111

Chapter 1

Introduction and Background

1.1 Introduction

This chapter motivates the need for subthreshold leakage reduction techniques and discusses existing methods for providing such reduction.

1.2 Motivation

The trend of process scaling for CMOS technology has made subthreshold leakage reduction a growing concern for submicron circuit designers. Power consumption has become a principle design consideration as device sizes decrease and many more devices fit on a single chip. Since switching power is proportional to $C_L V_{DD}^2$ new processes are tailored for lower supply voltages. Since the decrease in V_{DD} slows down logic gates, the threshold voltage, V_T, must be lowered to maintain performance. This reduction of V_T produces the problematic increase of subthreshold leakage currents according to the well-known equation

$$I_{leak} = I_o e^{\frac{V_{GS}-V_T}{nV_{th}}} \left(1 - e^{-\frac{V_{DS}}{V_{th}}}\right) \tag{1.1}$$

where $I_o = \mu_0 C_{ox} \frac{W}{L} V_{th}^2 e^{1.8}$. This equation can be approximated in the equally well-known form

$$I_{leak} \cong \frac{I_o}{W_o} W 10^{\frac{V_{GS}-V_T}{S}} \quad (1.2)$$

where $S = nV_{th} \ln 10$. As these equations show, leakage currents increase exponentially as V_T scales down with technology.

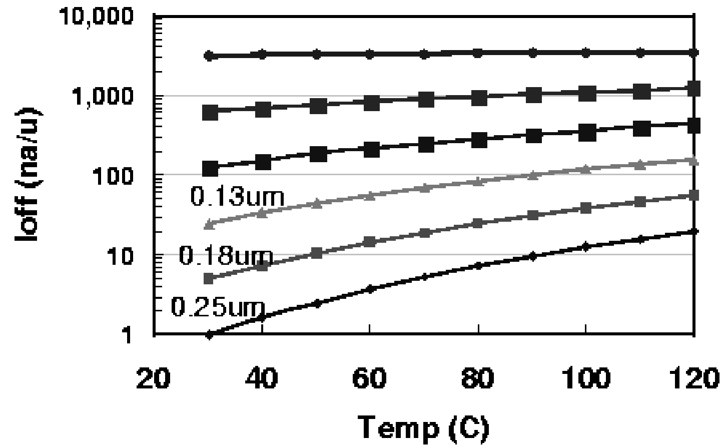


Figure 1-1: Sub-threshold (SD) Leakage

The graph in Figure 1-1 shows how subthreshold leakage scales with technology[2]. The figure shows that subthreshold leakage has increased by about 5X per generation for the last three generations. The numbers in the plot are normalized to 1nA/μm for the 0.25μm technology at room temperature. Clearly, this trend could cause extremely high subthreshold currents for future technologies.

In fact, Figure 1-2 shows research from Intel that predicts leakage power consuming over half of a chip's power at operating temperatures in a deep submicron process. Intel also shows in Figure 1-3 how the increase in I_{off} will produce exponential rises in leakage power for digital chips in future technologies. These plots from Intel give a representative picture of the concern about leakage current trends across the industry.

If unaddressed, this leakage power will exist in both active and standby modes. Although some new techniques address active mode leakage reduction, most techniques target standby leakage reduction. Many low power devices operate primarily

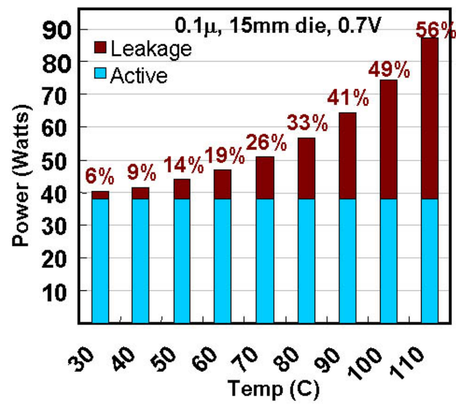


Figure 1-2: Trend of Increasing Leakage Power

in bursts of activity amid significant periods of inactivity. Standby leakage reduction techniques can reduce the power dissipation of such devices during these standby periods.

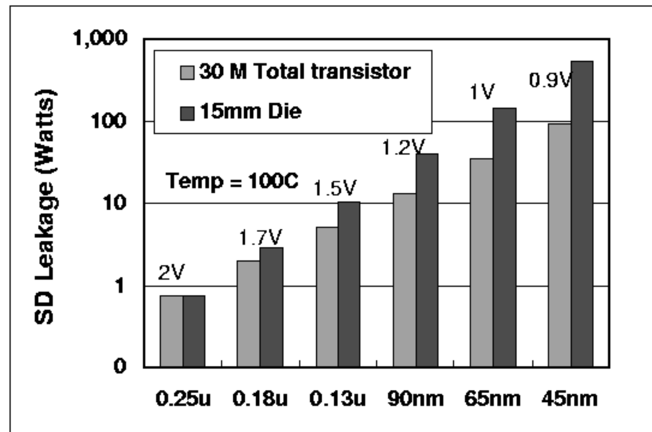


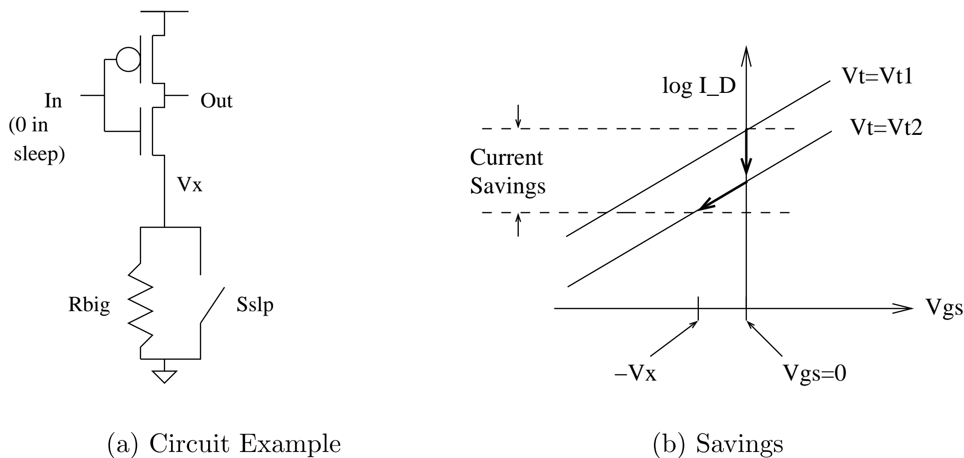
Figure 1-3: Leakage Power for Future Chips

1.3 Existing Techniques for Subthreshold Leakage Reduction

Equation 1.2 shows that exponential decreases in I_{leak} are possible by either increasing the threshold voltage or by decreasing V_{GS}. Existing techniques explore both of these options.

1.3.1 Switched Source Impedance

One obvious method for reducing leakage current is to decrease V_{GS} in standby mode by raising the source voltage through switched-source-impedence CMOS [3]. In this approach, a passive resistor is switched in between a gate's source nodes and ground during standby mode. During active mode, the resistor is shorted out so that there is no additional delay penalty.



(a) Circuit Example (b) Savings

Figure 1-4: Switched-Source-Impedence

The presence of the resistor in standby mode introduces a positive voltage at V_X due to the leakage current through R_{BIG} (see Figure 1-4(a)). The voltage V_X reduces leakage current in two ways. First, the increased source voltage of the NMOS device

causes an increase in the threshold voltage due to the well-known equation:

$$V_T = V_{t0} + \gamma \left(\sqrt{V_{SB} + 2\phi_F} - \sqrt{2\phi_F} \right)$$

This higher V_T causes a shift in the current-V_{GS} curve in Figure 1-4(b) from $V_T = V_{T1}$ to $V_T = V_{T2}$. Furthermore, the increased V_X reduces V_{GS} of the NMOS device, causing the operating point to shift downward along the current-V_{GS} curve. The combination of these two effects lowers the standby leakage current. In practice, switched-source-impedence circuits are rare because the resistor must be large and resistor placement depends on knowing the standby state of every gate. Since there is no perfect switch, the resistance of the switch often replaces the explicit resistor for this technique.

1.3.2 Stack Effect

A similar and more practical technique for subthreshold leakage reduction takes advantage of the stack effect. The term *stack effect* refers to the reduction of subthreshold leakage by stacking multiple FETs in series. Figure 1-5 shows a case of artificially introducing the stack effect to an inverter, but stacks of transistors occur naturally in many logic gates (*i.e.* - NAND). The stack effect provides leakage reduction in the same way as switched-source-impedence. The last device in the stack essentially appears as an impedence to any leakage current, so the voltage V_X becomes non-zero. At that point, the effects shown in Figure 1-4(b) occur as previously described. Notice that the imposed stack effect approach is essentially the switched-source-impedence approach without the resistor. This approach just uses the resistance of the non-ideal switching FET in an off state.

Designers take advantage of the stack effect in two ways. First, one can exploit natural stacks in a circuit by determining an input vector at design time that turns off stacks of devices. Several algorithms exist for applying test vectors to a circuit at design time to find the vector that generates the lowest leakage current (*e.g.* - [4] [5]). When the circuit enters standby mode, this low-leakage vector is muxed into the circuit inputs.

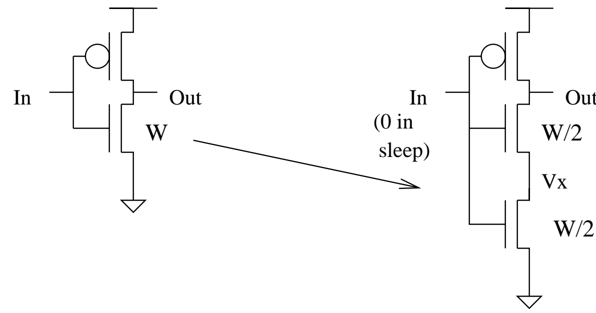


Figure 1-5: Artificial Creation of Stack Effect

Artificial introduction of stacks to a circuit [6] can enhance the gains from the vector-finding method of leakage reduction. In this approach, stacks are introduced to gates off of the critical path (Figure 1-5). While slower itself, the new gate presents the same capacitive load at its input, so it does not slow down previous gates. This method trades off speed on non-critical paths and design time for reduced leakage. A new model for determining the savings introduced by forcing stacks allows designers to determine the usefulness of the approach for a given technology [6].

1.3.3 Body Biasing

A entirely different approach to reducing subthreshold leakage biases the body voltage to change V_T. There are many angles taken to use this technique, but they all require isolation of the bulk for different FETs (as in a triple well process). This research will not focus on body biasing.

1.3.4 Dual-Threshold CMOS

Dual-threshold CMOS techniques use FETs having two thresholds to reduce leakage currents. The most straightforward application of dual-threshold CMOS uses all low V_T FETs on the critical path and high V_T FETs off of the critical path. The high V_T devices placed at design time reduce leakage while increasing delay off of the critical path. The placement of these devices can become complicated since the high V_T devices can slow down parts of the circuit enough to create new critical paths.

Nevertheless, several algorithms exist for that purpose (*eg* - [7]). One advantage of dual-threshold CMOS is that the leakage savings occur in both active and standby modes. Disadvantages include more complicated design time and the inability to stop critical path leakage during standby mode.

Dual-threshold CMOS is particularly useful for some special logic styles. For example, domino logic can use high V_T devices on the precharge path [8]. All of the pull-down paths remain fast because of the low V_T devices, but each path between the rails contains at least one high V_T device. This allows the tradeoff of precharge time with reduced leakage.

1.3.5 MTCMOS

Multi-Threshold CMOS (MTCMOS) refers to a circuit technique that uses a high V_T footer and/or header FET to sever a circuit from the power rails (Figure 1-6) [9]. Similar approaches have been applied to caches [10, 11] and to DRAMs [12]. In active

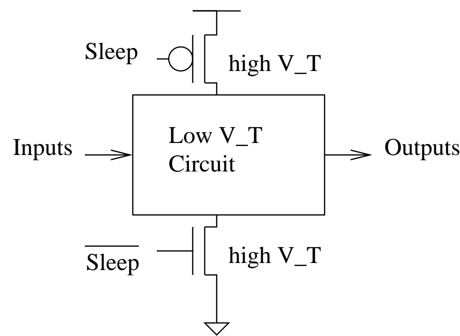


Figure 1-6: MTCMOS Schematic

mode, the high V_T switches are on to enable regular circuit operation. Turning off the high V_T devices in standby mode places at least one high V_T FET on each path from power to ground, thereby reducing the leakage current significantly.

Sizing the sleep devices is no easy task. Small sleep devices further reduce the leakage current in standby mode but slow down transitions in active mode. The loss of performance occurs because smaller sleep devices sink less current through to ground.

This exacerbates the bouncing created at the virtual rails and effectively pinches V_{DS} for the low V_T circuit. Furthermore, this rail bounce depends on the discharge pattern of the circuit, so it can be difficult to predict worst-case performance.

Combinational logic blocks only need a sleep device on one rail, but sequential MTCMOS circuits are more complicated. Simply disconnecting a circuit from the rails in standby destroys any state previously stored in that circuit. Special care is necessary to retain state in MTCMOS sequential circuits.

Despite some of the design difficulties, MTCMOS techniques appear to be viable for reducing leakage currents in new technologies.

1.3.6 BGMOS and SCCMOS

Two additional proposals for reducing leakage basically increase the effectiveness of MTCMOS sleep transistors. The Boosted Gate MOS (BGMOS) approach [13] suggests that overdriving the gate voltage of sleep devices in active mode can reduce the area devoted to sleep devices. The overdriven (above V_{DD}) gate voltage enables a smaller device to sink the required amount of current in active mode while still effectively stemming the leakage current in standby mode. A similar approach called Super Cut-off CMOS (SCCMOS) advocates underdriving gate voltages in standby mode [14]. For example, an NMOS sleep device would be turned off in standby mode with a negative gate voltage rather than a gate voltage of 0 volts. From equation 1.2 we see that the negative gate voltage provides exponentially decreasing leakage current through the sleep device.

1.4 Leakage Feedback Flip-flop

1.4.1 MTCMOS Sequential Circuits

Straightforward MTCMOS design prevents the use of sequential logic because it disconnects internal nodes from the power rails. Without an active source to hold a stored value, any sequential gate could lose its state due to floating nodes.

Several strategies exist for implementing sequential circuits using the MTCMOS approach. For example, high V_T devices placed in parallel with low V_T devices can hold the state of storage nodes while the low V_T inverters are disconnected from the rails[9]. This approach adds the additional capacitive load of the high V_T device on the critical path of the flip-flop. A high V_T "balloon" circuit consisting of back to

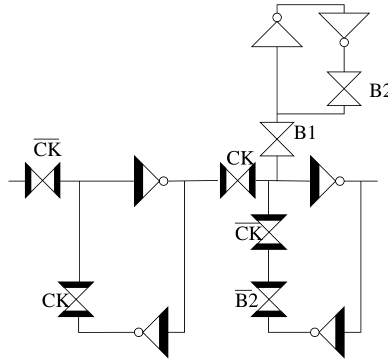


Figure 1-7: MTCMOS Balloon Flip-Flop

back inverters can hold state at a given node [15, 16, 17]. Figure 1-7 shows one design using the balloon circuit. All of the low V_T inverters in the figure have implicit high V_T sleep devices at both rails. This approach increases the load on the critical path. It also can require complicated timing to store the data into the balloon and to retrieve data from the balloon. Another approach uses the output of a flip-flop to control the sleep state of the device conditionally [18]. This permits one rail to drive the output actively while conditionally disconnecting the output node from the other rail. This implementation cannot retain state correctly if the inputs to the flip-flop change.

This project focuses on the Leakage Feedback Flip-flop (LFBFF) proposed in [1]. The next two sections describe this flip-flop in more detail.

1.4.2 Leakage Feedback Gate

A leakage feedback gate holds its state during standby mode by maintaining an active path to one rail contigent on the output. Figure 1-8 shows the concept at work. When the gate enters sleep, devices M1 and M3 turn off. One of either M2 or M4 also turns

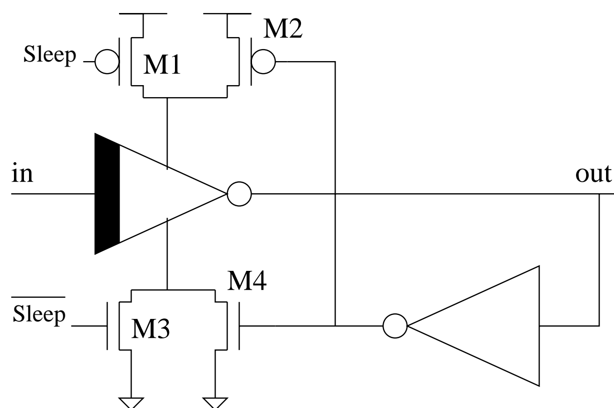


Figure 1-8: Leakage Feedback Gate

off depending on the output of the gate at the time that the sleep signal rises. For example, suppose that *out* is '1' when sleep is asserted. This means that the gates of M2 and M4 are '0', so M4 is off and M2 remains on. After the sleep signal turns off devices M1 and M3, the output node connects actively to the V_{DD} rail through M2. The turned-off pair of M3 and M4 disconnect the output node from ground, effectively gating all subthreshold leakage paths with high V_T devices. The opposite occurs when *out* is '0'.

This gate has the interesting property of maintaining its output even when the input node floats. Take the case of the example above. When the gate enters sleep, node *in* is '0'. Suppose that *in* floats to an intermediate value or even to '1' once in sleep. The feedback path at the output continues to keep M2 on and M4 off. Node *in* changing causes the NMOS inside the inverter to turn on and the PMOS in the inverter to turn off. Now, the low V_T PMOS device in the inverter buffers the output node from V_{DD}, but the high V_T devices M3 and M4 buffer it from ground. The order of magnitude higher leakage current through the low V_T device will continue to hold the output at the correct logical value of '1'. The opposite occurs if the output was originally '0'.

Even though mismatched leakage currents can hold the output at the correct voltage when the inputs float, the lack of any active connection at the output node makes it particularly susceptible to capacitive coupling. A strong coupling with a nearby

node that switches could overcome the mismatch in leakage currents to produce an incorrect value at the output. Designers must position leakage feedback gates (and flip-flops) to prevent such capacitive coupling. Strong capacitive coupling often is eliminated inherently because the leakage feedback gate (or flip-flop) lies in the midst of an entire region of devices in sleep state. This means the surrounding devices will not be switching as long as the region remains in standby mode.

The properties of the leakage feedback gate make it useful for sequential logic design. The gate also can provide an interface between MTCMOS and CMOS gates. The floating nodes generated in the MTCMOS region during sleep mode do not affect the state of the LFB gate. Thus, it maintains an actively driven input to the CMOS logic even when the MTCMOS circuits are in standby mode.

1.4.3 LFBFF Design

A leakage feedback flip-flop (LFBFF) [1, 19] applies the principles of the leakage feedback gate to a traditional master-slave flip-flop. The inherent design of the latches inside the flip-flop allow this implementation to avoid adding any capacitive load to the nodes on the critical path. Instead, the high V_T inverters that feedback to hold state during the active mode also provide the feedback for the conditional sleep gating in standby mode. Figure 1-9 shows the implementation of the LFBFF.

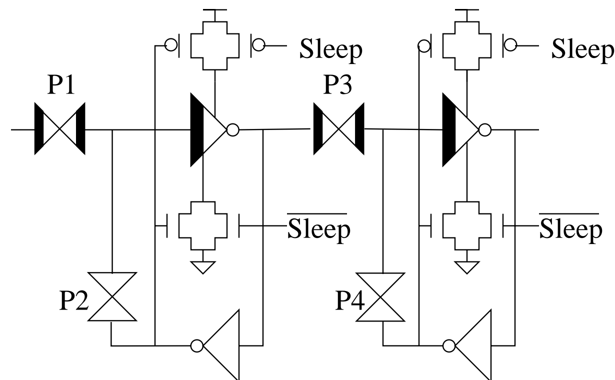


Figure 1-9: Leakage Feedback Flip-Flop

The critical path through the LFBFF consists solely of low V_T devices. As pre-

viously mentioned, it sees no extra loading from the feedback mechanism beyond the load inherent to the latches. During sleep mode, passgates P1 and P4 are off, and passgates P2 and P3 are on. Thus, the master stage stores the data value and propagates it through the low V_T inverter in the slave stage to the output. The sleep devices attached to the sleep signal all turn off. The other sleep devices (minimum sized) conditionally turn off to maintain the correct values at the internal and external storage nodes. Simulations have shown that the LFBFF compares very favorably in terms of speed and power to other MTCMOS sequential circuits [1, 19]. This project provides the first test of this flip-flop design in silicon.

1.5 Contributions of Thesis

This thesis makes several contributions in the area of subthreshold leakage reduction. First, it introduces supply voltage scaling as a method to reduce power consumption during standby mode while retaining state information. Analysis of the LFBFF shows the source of the limits of supply voltage scaling. Understanding the mechanism of failure of the LFBFF leads to the design of a second LFBFF that fails at a higher voltage than the LFBFF of interest. This permits an adaptive method for scaling the supply voltage.

A testchip applies MTCMOS leakage reduction techniques to a new FPGA architecture. The design demonstrates the LFBFF in silicon for the first time. It also introduces configurable sleep regions that provide reductions in leakage power during the active mode. An analysis of the sleep region interfaces and of sneak leakage paths provides additional characterization that can make MTCMOS designs more effective.

Chapter 2

Characterizing Leakage in a State-of-the-Art 0.13μm CMOS Process

2.1 Introduction

This section discusses the technology used for the project. Through simulation, it explores the effects that different parameters have on subthreshold leakage. It also compares the accuracy of the BSIM2 MOS transistor model[20] to the simulated results. The comparison provides confidence in the models for designing in this process. The results are normalized to protect proprietary information about the process.

2.2 Leakage Models

The previous section introduced the problem of subthreshold leakage and gave two equivalent equations for modeling leakage. The equations are:

$$I_{leak} = I_o e^{\frac{V_{GS}-V_T}{nV_{th}}} \left(1 - e^{-\frac{V_{DS}}{V_{th}}}\right)$$

where

$$I_o = \mu_0 C_{ox} \frac{W}{L} V_{th}^2 e^{1.8}$$

and

$$I_{leak} \cong \frac{I_o}{W_o} W 10^{\frac{V_{GS}-V_T}{S}}$$

where

$$S = nV_{th} \ln 10.$$

The BSIM2 models are similar to these equations, but they add a few higher order elements. For example, these first order equations neglect the Drain Induced Barrier Loading (DIBL) effect. Equation 2.1 gives the BSIM2 model for subthreshold leakage:

$$I_{leak} = I_o e^{\frac{V_G - V_S - V_{T0} - \gamma V_S + \eta V_{DS}}{nV_{th}}} \left(1 - e^{-\frac{V_{DS}}{V_{th}}}\right) \tag{2.1}$$

where, again,

$$I_o = \mu_0 C_{ox} \frac{W_{eff}}{L_{eff}} V_{th}^2 e^{1.8}$$

To maintain the naming conventions for this paper, V_T is the threshold voltage, and V_{th} is the thermal voltage. Thus, V_{T0} is the zero bias threshold voltage, γ is the linearized body effect coefficient, and η is the Drain Induced Barrier Loading (DIBL) coefficient. As before, the parameter n is related to the subthreshold slope by the equation

$$S = nV_{th} \ln 10. \tag{2.2}$$

The following sections compare this BSIM2 equation to HSPICE simulations of the leakage current under various conditions.

2.3 Subthreshold Current Versus V_{GS}

Figure 2-1 shows the simulated subthreshold current versus V_{GS} for the four FETs used from the 0.13μm process. The HSPICE simulation that produced this data operated at a temperature of 25C. Each device has the minimum length of 0.13μm

and a set width. The supply voltage for the simulation was 1.0V. The current through these devices was measured over a range from $V_{GS} = 0$ to $V_{GS} = 300$mV. The 300mV number approximates the upper edge of the subthreshold region for the devices.

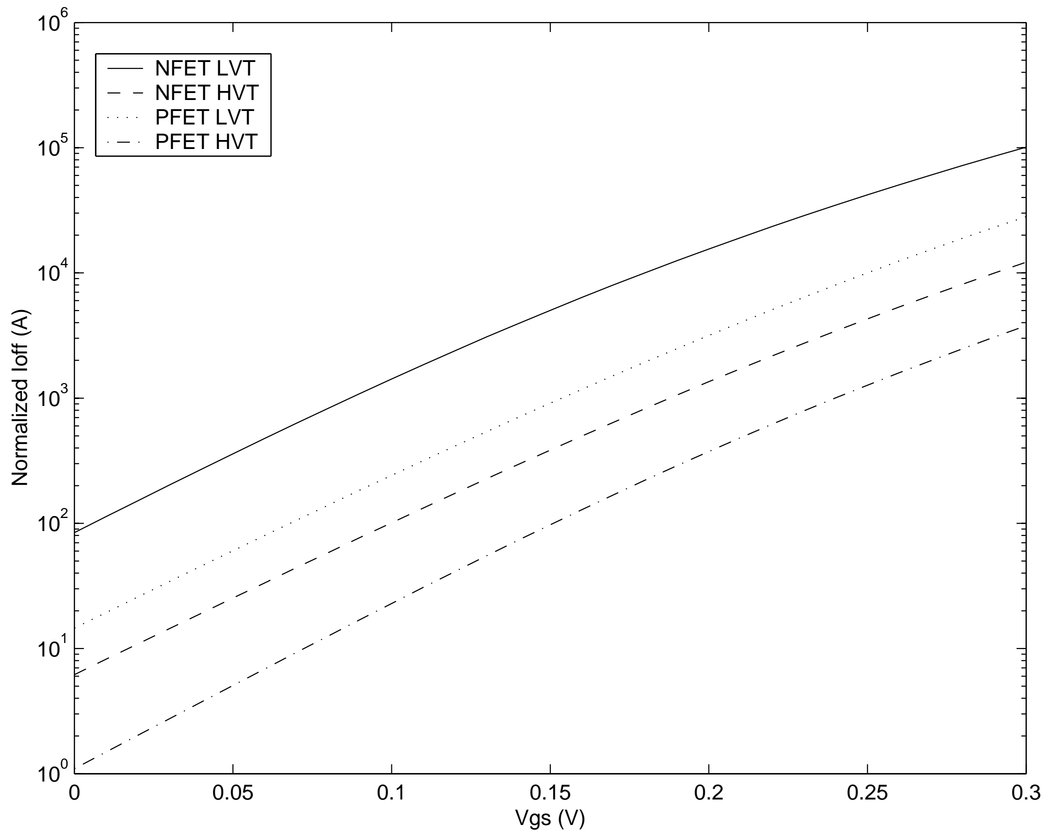


Figure 2-1: Subthreshold Current vs V_{GS}

The normalized plot shows that both PMOS and NMOS low V_T devices have subthreshold currents more than an order of magnitude greater than their high V_T counterparts at $V_{GS} = 0$. For both NMOS and PMOS, this ratio rolls off slightly closer to the threshold voltage.

2.3.1 Subthreshold Slope

Several parameters are necessary for comparing the BSIM2 model to the simulated results. The parameters that are not readily available are γ, η, and n. This device characterization uses single devices, so the body effect does not come into play. We

describe in a following section how we estimated the value of η. Extracting the subthreshold slope from the simulations allows us to find a valid n for use in the model.

The tailing off of current in Figure 2-1 indicates that the subthreshold current does not increase as rapidly with gate voltage as V_{GS} nears the threshold voltage. In other words, the subthreshold slope increases along with V_{GS}. Taking the derivative of the curves in Figure 2-1 and inverting the result provides an instantaneous estimate of the subthreshold slope at each value of V_{GS}.

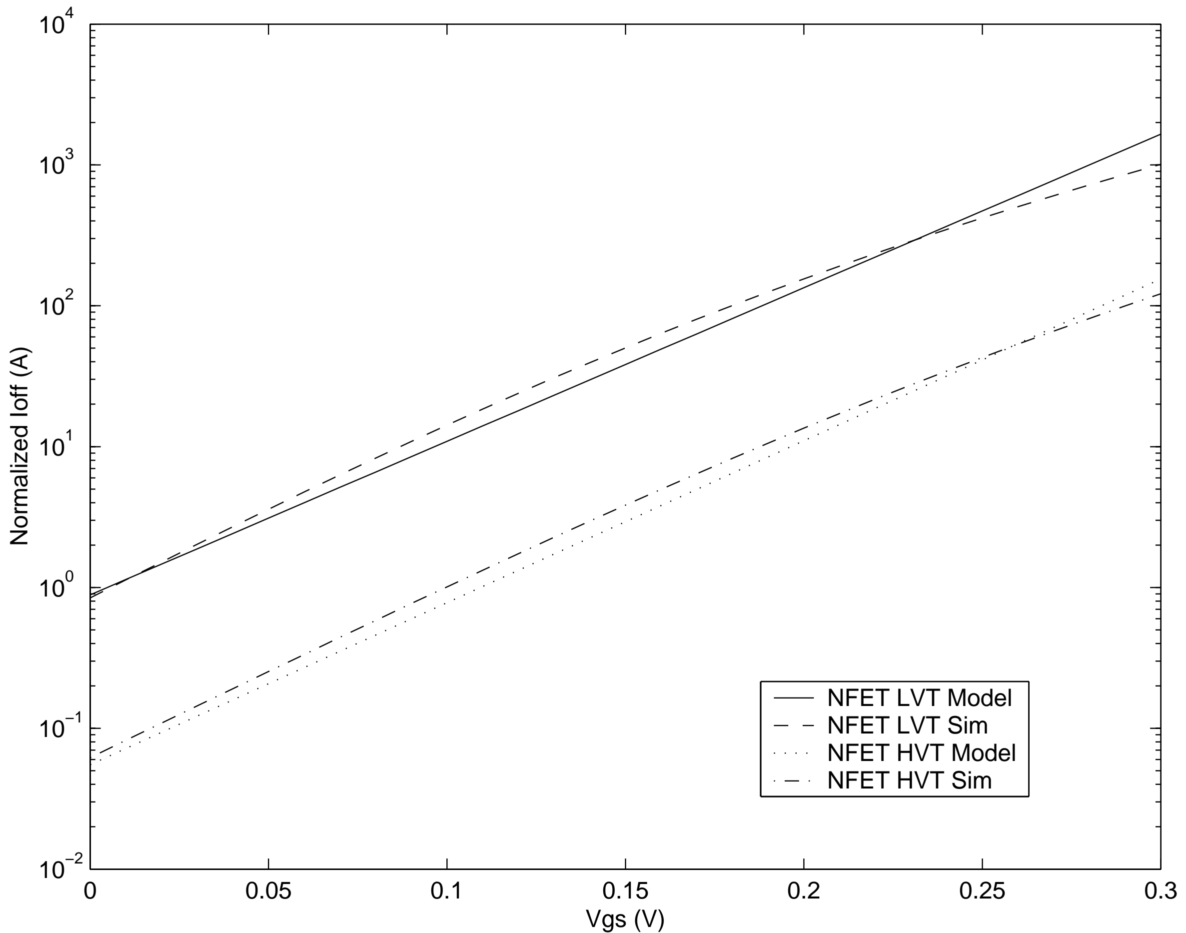


Figure 2-2: Model versus Simulation for Varying V_{GS}

The "instantaneous" value of n can be derived by applying Equation 2.2 to the subthreshold slope data. Selection of n makes a considerable difference in the model

since Equation 2.1 depends exponentially on n. The value of n varies rather widely across the V_{GS} range. Experimentation showed that a value of n from lower V_{GS} values provided a close match between the model and the simulated results. This value of n also is the mean value of the data for n over the lower range of V_{GS}. Neglecting the data from the higher gate voltages presumably gave a more accurate estimate for the value of n because the neglected range of n corresponds to the devices beginning to leave the subthreshold region.

Figure 2-2 shows that the normalized BSIM model for subthreshold leakage matches well with the simulated results for the NFETs. The PFET models showed similar matching accuracy. The models used the value of n determined using the above method. The next section describes the process for determining η.

2.4 V_{DS} and the DIBL Coefficient

Equation 2.1 shows that the leakage current of a FET in subthreshold depends on the supply voltage in two ways. First, the leakage current rolls off exponentially with V_{DS} due to the term in parentheses, $1 - e^{-\frac{V_{DS}}{V_{th}}}$. This term only becomes noticable at low values of V_{DS}. The leakage current also increases exponentially based on V_{DS}. The coefficient that weights the exponent is the DIBL facter. Drain Induced Barrier Loading (DIBL) refers to the effective reduction of threshold voltage with an increase of V_{DS}. This occurs in short channel devices because the depletion regions around the source and the drain junctions can be reasonably large relative to the channel length. Increasing V_{DS} causes an increase in the depletion regions at the source and drain which allows a smaller V_{GS} to invert the channel.

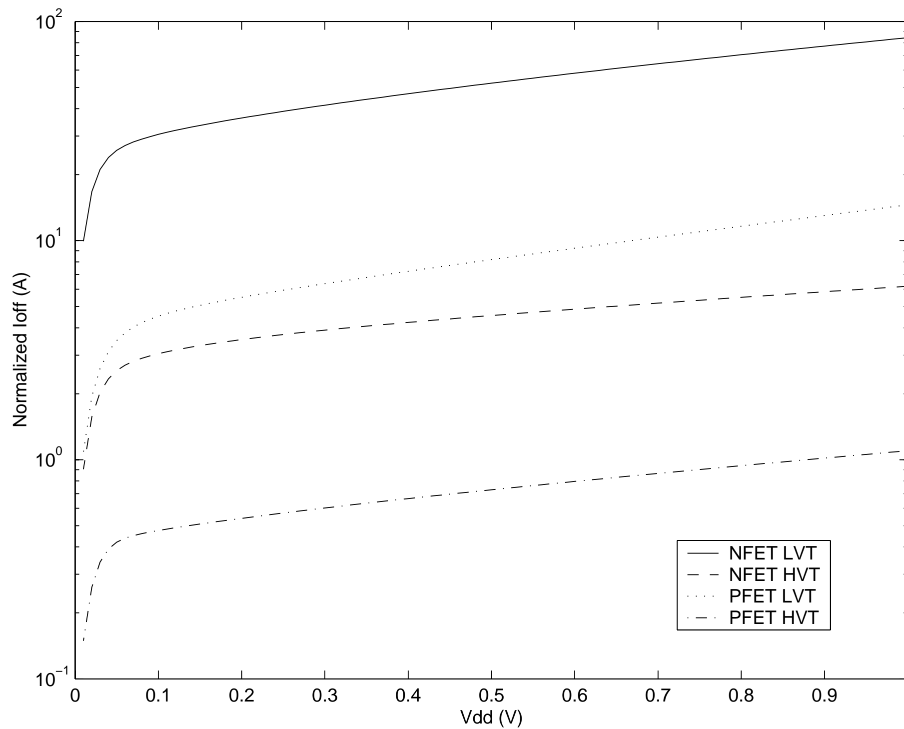


Figure 2-3: Simulation of Subthreshold Leakage versus V_{DS}

Figure 2-3 shows simulated data for all four types of FETs. Sweeping the V_{DS} from a small value to the full value of 1V shows the dependence of the leakage currents

on V_{DS}. The parenthetical term from the equation mentioned previously models the rolloff of the current that appears at extremely low voltages. The slope of the current over the rest of the voltage range arises from the DIBL effect.

Once all other parameters in the BSIM2 model were determined, sweeping η provided an easy method for matching the DIBL coefficient. Experimentation allowed a choice of η that made the slope of the curve for the model match the simulation data. Figure 2-4 shows the results for the NFETs. The data for the PFETs was similar. The figure shows that a careful selection of η permits a close match in the slopes of the model and the simulation data. The slight offset for the high V_T device occurs because the model did not match as well at the $V_{GS} = 0$ and $V_{DS} = 1$V point (see Figure 2-2).

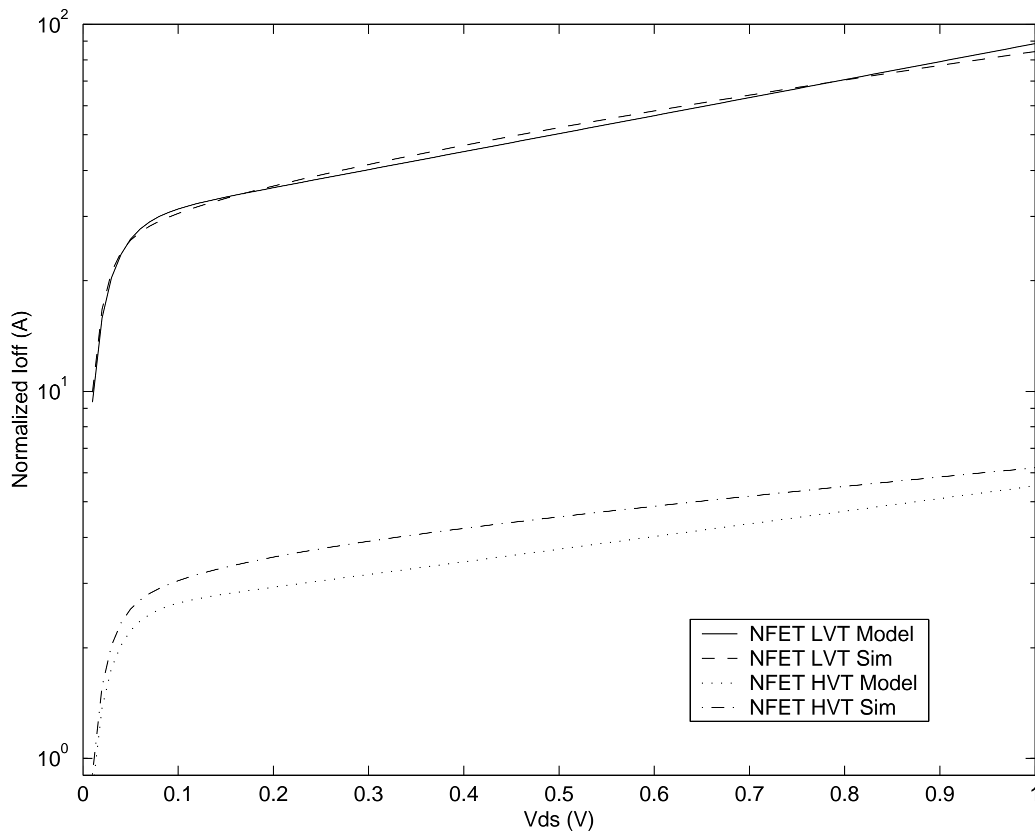


Figure 2-4: Model Versus Simulation for Varying V_{DS}

2.5 Temperature

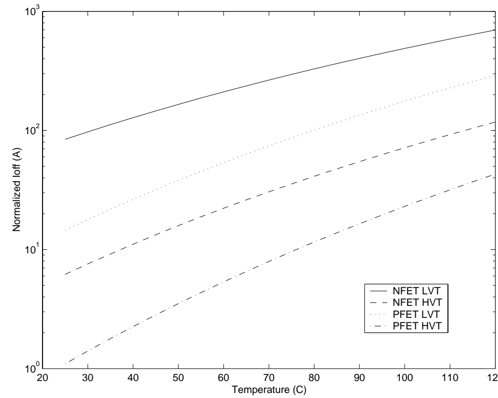


Figure 2-5: Simulation of Subthreshold Leakage versus Temperature

Figure 2-5 shows simulated data for leakage current versus temperature for all four types of FET. The models required no special attention to portray the effects of temperature on subthreshold current accurately. As expected, the subthreshold leakage increases exponentially with temperature. The exponential increase comes from the V_{th} in the denominator of the exponent in Equation 2.1, where $V_{th} = \frac{kT}{q}$. The exponential rate of increase with temperature increases further due to the V_{th}^2 in the expression for I_o. Figure 2-6 shows the models versus simulations for the NFETs.

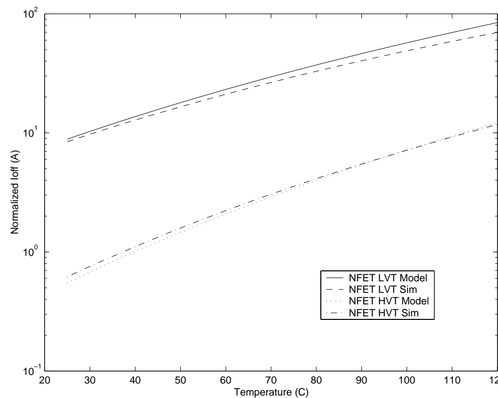


Figure 2-6: Model Versus Simulation for Varying Temperature

2.6 Subthreshold Leakage Versus Device Size

Both the first order model in Equation 1.1 and the BSIM2 model in Equation 2.1 show a simple linear relationship between device size and subthreshold leakage current. The equations predict that the leakage current will increase in proportion with device width. Similarly, the leakage current is inversely proportional to device length. Extensive research has shown the inability of these lower order equations to model device behavior for short and/or narrow devices.

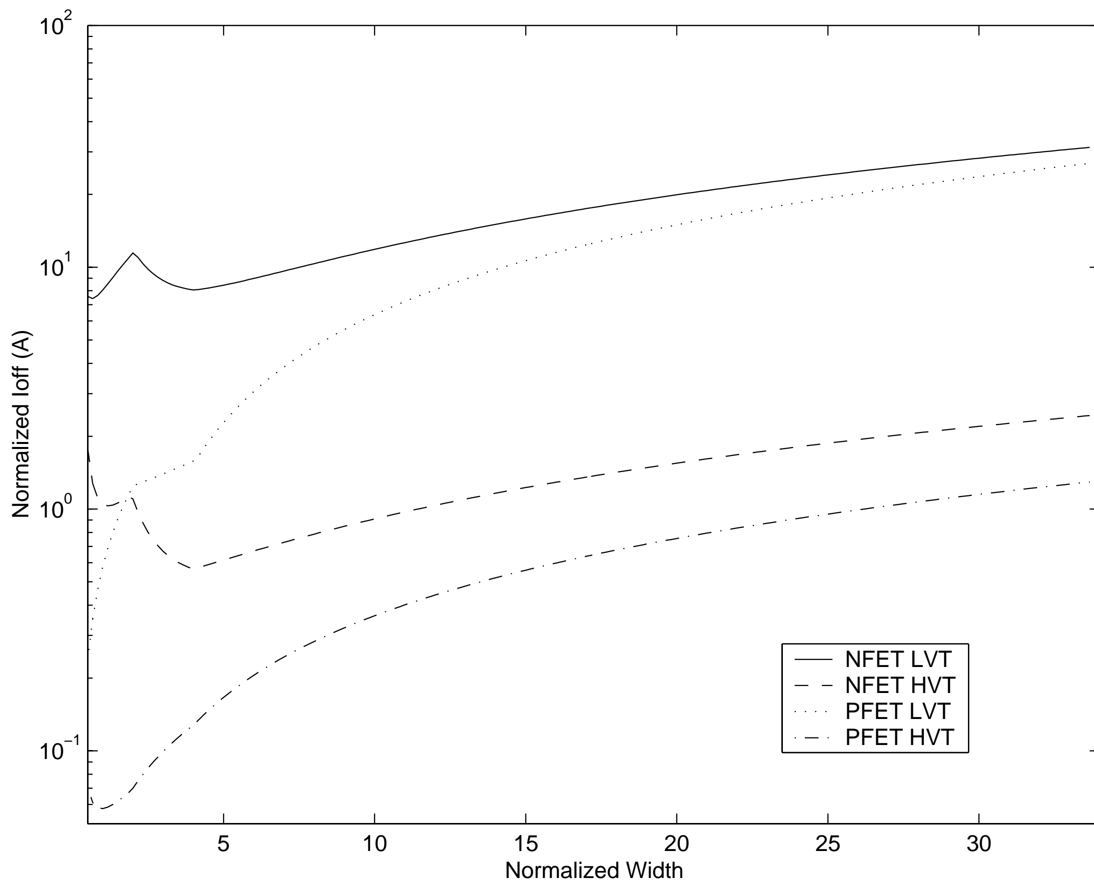


Figure 2-7: Threshold Voltage Variation with Device Size

Figure 2-7 shows the simulated leakage current varying with device width in the 0.13μm CMOS process. The first order, proportional relationship between width and leakage current begins to emerge only when the width becomes relatively large. The relationship at lower widths clearly is not linear. Instead, lowering width produces

a brief rise in current followed by a more dramatic tail-off. These two effects can produce a pronounced peak as in the case of the low V_T NFET, or they can appear more as ripples in the current plot as in the case of the PFETs. In either case, the higher order effects on leakage current can cause headaches for circuit designers.

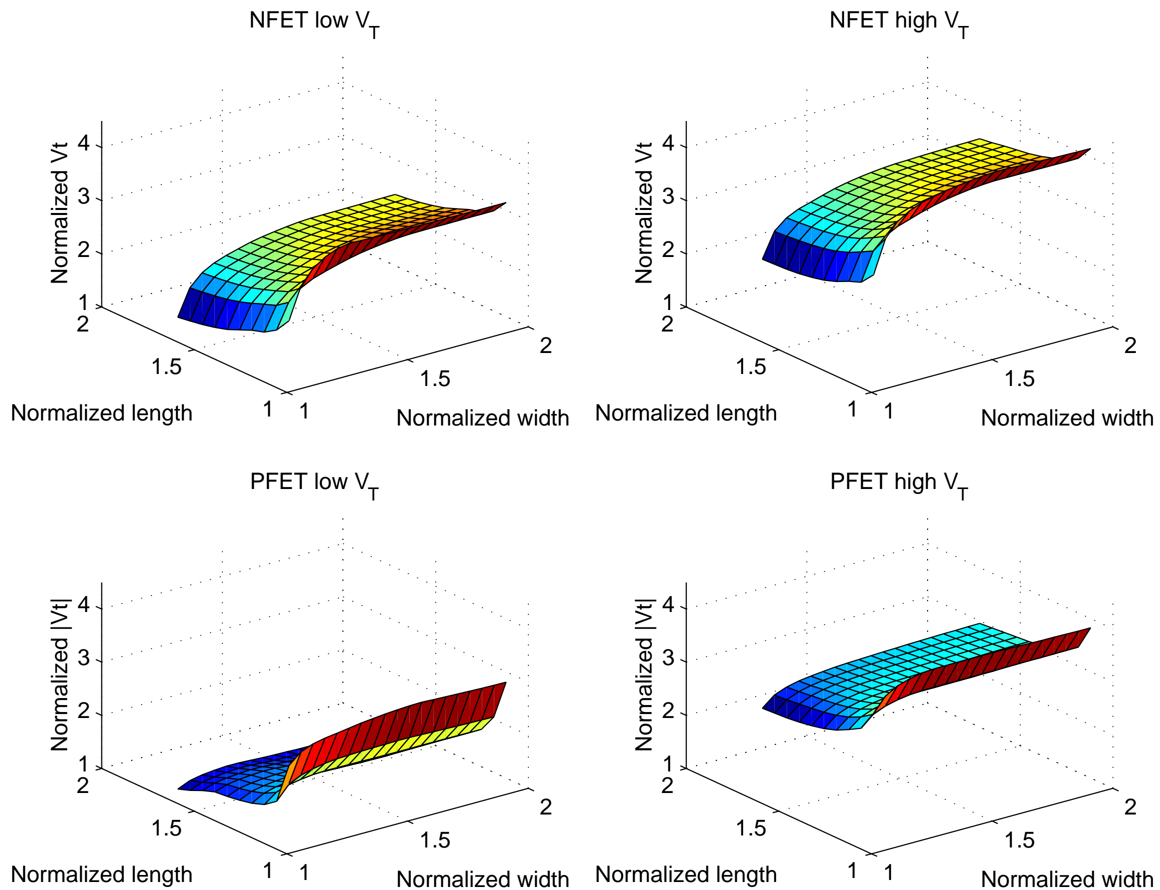


Figure 2-8: Threshold Voltage Variation with Device Size

Device research has characterized these effects for short, narrow devices. The leakage current at narrow widths changes because the effective threshold voltage changes. Figure 2-8 shows the simulated variation of threshold voltage with device size. The axes in the four plots are the same to facilitate comparison. For every one of the devices, the threshold voltage increases as length decreases, and V_T decreases as width decreases.

The increase in V_T with decreasing length opposes the idea of Short Channel

Effect that suggests that V_T will decrease for a shorter length device. Appropriately, the relationship between V_T and length exhibited in Figure 2-8 is called the Reverse Short Channel Effect (RSCE). Several methods exist for modeling the RSCE (*i.e.* - [21, 22]). The RSCE occurs because of a pile-up of dopant material (*e.g.* boron) along the edges of the source and drain. As the channel shortens, the pile-ups of dopant at the channel edges begin to overlap. This increases the effective channel doping value and leads to the increase in threshold voltage.

The decrease in V_T with decreasing width is called the Reverse Narrow Channel Effect (RNCE). Existing work attributes the RNCE to transient enhanced diffusion (TED) that redistributes dopant material (*e.g.* boron) away from the isolation oxide along the edges of the channel [23]. When the device width decreases, the lower concentration of dopant along the edges of the channel extends across more of its width. This reduces the effective doping of the channel and reduces V_T. The TED is the same mechanism that causes the dopant pile-up at the edges of the source and drain to cause the RSCE.

The RNCE explains the abnormalities that appear in the leakage current plot in Figure 2-7. As device width lowers, the leakage currents decrease with width but increase due to a lowering V_T. The combination of these effects produces the simulation results in the figure.

These variations in subthreshold current with device size have at least one interesting practical consequence for circuit design in this process. The minimum sized high V_T device is no longer minimum sized in the traditional sense. Figure 2-7 shows that the minimum leakage current for the high V_T NFET occurs at a width of about 4 times minimum size. A high V_T NFET of this size has a larger on-current than a minimum sized device, but its off-current is several times less. For this reason, the "minimum sized" high V_T devices in the testchip are actually about 4X the minimum physical size.

Chapter 3

Voltage Scaling During Standby Mode

3.1 Introduction

This chapter proposes scaling the supply voltage during standby mode. MTCMOS circuits using LFBFFs could save power while retaining state in standby mode by using a reduced supply voltage. Lowering the voltage supplied to the sequential logic reduces power directly (P=VI). It also reduces the leakage currents drawn by the flip-flops. This chapter examines the limits of supply voltage scaling for LFBFFs. It also proposes a method for adaptively adjusting the supply voltage to sequential logic during standby mode.

3.2 Voltage Scaling Concept

The leakage feedback flip-flop (LFBFF) proposed in [19] has the ability to hold its state in standby mode, using a mismatch in leakage currents, even when its inputs float. This mechanism for memory retention should continue to function correctly even with lower supply voltages.

MTCMOS circuits using LFBFFs could save power while retaining state in standby mode by using a reduced supply voltage. Different regions of a circuit permit differ-

ent depths of voltage scaling. For example, the power supplied to combinational logic can be disconnected entirely. Sequential logic needs to remain powered to hold state. Lowering the voltage supplied to the sequential logic reduces power directly (P=VI) and by reducing leakage currents. These operations would eliminate subthreshold leakage in the combinational logic and dramatically reduce standby leakage power in the sequential logic.

3.3 Varying Supply Voltage and the LFBFF

A given implementation of LFBFFs has a lower bound on the supply voltage below which the device cannot hold state. Figure 3-1 shows the LFBFF in sleep mode with

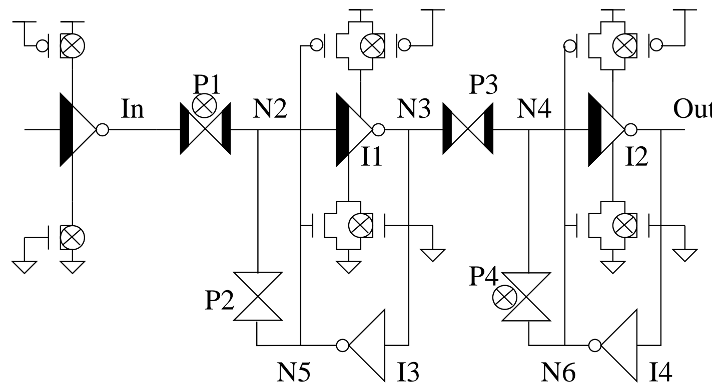


Figure 3-1: LFBFF in Sleep Mode

an input inverter. The following two sections describe the LFBFF holding state with lower supply voltage. The sections detail the mechanism of failure for each storage value.

Figure 3-1 implicitly includes a few signals that deserve mention. The sleep devices connected to the *sleep* and $\overline{sleep}$ signals appear in the figure with their gates tied to power and ground, respectively. The discussion in this chapter assumes that the sleep signal remains at the full V_{DD} value of one volt throughout any voltage scaling. This is true for all of the simulations described in the chapter. The sleep voltage remains high to simplify analysis since the high sleep signal drives the PMOS sleep

devices with a negative V_{SG} when the supply voltage decreases. This decreases the subthreshold current in those devices to a negligible value. Since the sleep signal is routed with its own power supply on the testchip (see Chapter 5), the sleep signal can scale along with the flip-flop voltage. The analysis of power savings in the conclusion shows that scaling the sleep power supply degrades the ability of the flip-flops to hold state at a low voltage but increases overall power savings. Figure 3-1 also shows the on or off state of the passgates. The *clock* and $\overline{clock}$ signals determine the state of the passgates. Since the testchip employs MTCMOS gates to drive the clock signals, the logical high *clock* will take on the reduced supply voltage value of the flip-flop supply during voltage scaling. The In node also scales with the flip-flop power supply because the input buffer lies in the flip-flop power region.

3.3.1 Storing a Logic '1'

The flip-flop maintains the correct state as long as the nodes in the circuit all maintain the correct logical values. Table 3.1 shows the state of the nodes in the flip-flop for a stored logic '1'. The nodes that should retain a '0' value have no trouble doing so as

Table 3.1: LFBFF State for Stored '1'

Node Name	Value
N2	'1'
N3	'0'
N4	'0'
N5	'1'
N6	'0'
Out	'1'

V_{DD} scales so long as the logical '1' nodes remain correct. The logical '1' nodes are the initial point of failure for the flip-flop as the power supply lowers.

As Table 3.1 shows, there are two key nodes for storing a '1' that need to retain the high value. These are the N2/N5 node and the output node. Figure 3-2 shows the schematic of the flip-flop holding this state. This initial analysis assumes that

MP1, MP2, and inverters I3 and I4 are all minimum sized. Sizing these devices for minimum subthreshold leakage makes sense because they do not have to be large for active mode operation. This analysis will show, however, that their sizes can dictate the minimum voltage supply during voltage scaling.

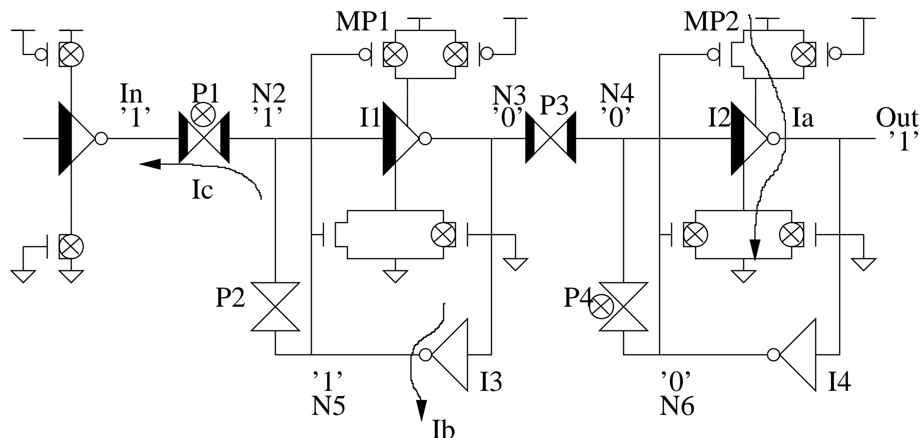


Figure 3-2: LFBFF Holding a '1'

First consider the output node holding a '1'. The output will remain a logical '1' as long as MP2 can supply all of the current drawn through the two NMOS sleep devices with $V_{SD}(MP2) \approx 0$. Note that the current through the other PMOS sleep device is negligable because its V_{SG} becomes strongly negative as V_{DD} scales and *sleep* remains at 1 volt. The combined width of the NMOS sleep devices in the slave latch increases their subthreshold current relative to the current supplied through the minimum sized MP2. The source to gate voltage of MP2 also decreases with the power supply voltage. As the voltage supply drops, MP2 eventually transitions from the linear region to the subthreshold region. The combination of these effects eventually decreases the amount of current MP2 can support (with $V_{SD} = 0$) below the current supported by the NMOS devices. Since I_a in Figure 3-2 goes through both MP2 and the NMOS sleep FETs, MP2 must increase its V_{SD} to maintain the current balance. Once this happens, the voltage at the output node begins to fall at a faster rate than the decreasing supply voltage.

Figure 3-3 shows the failure of the output node to hold a '1'. This failure occurs

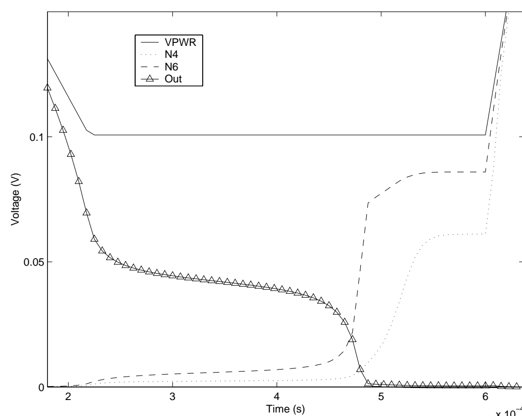


Figure 3-3: Output Node Changing Value for Stored '1'

with a supply voltage of just above 100mV. At this value of V_{DD}, the circuit just barely fails, so it takes some time for the voltage values to settle before they cause the output node to change its logical value. The figure shows that node N6 begins to increase its voltage value well before the latch actually fails. As the output node voltage (input to I4) begins to drop below the V_{DD} value, the PMOS in inverter I4 begins to increase its subthreshold current as the NMOS in I4 loses some of its current drive. Node N6 must rise above zero to balance the current in the inverter. The increase of N6 aggravates the situation with I_a by reducing the gate drive for MP2 and increasing the gate drive of the NMOS sleep device. This feedback adds to the inability of MP2 to sustain I_a without the output voltage dropping even more.

A similar situation causes the logical '1' at node N5 to transition to '0'. Figure 3-2 shows two currents, I_b and I_c, that act to pull node N5/N2 toward ground. Current I_b occurs because of the inability of the minimum sized PMOS inside inverter I3 to sustain the leakage current drawn by the NMOS device in that inverter without an increased V_{SD}. Similar to the process described above, N5 drifts below the V_{DD} value. Another current mismatch in the sleep devices for the input buffer causes node IN to begin to fall below V_{DD}. This falling voltage pulls node N2 toward ground by the large leakage current, I_c, through the low V_T passgate P1. The combined effects of I_b and I_c cause node N2/N5 to transition to a logical '0'. Figure 3-4 shows this process

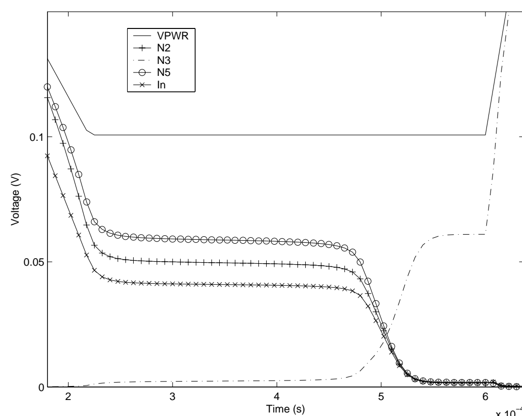


Figure 3-4: Internal Node Changing Value for Stored '1'

occurring during the same simulation that produced Figure 3-3.

The combinations of the effects in these two figures result in the failure of the flip-flop to maintain a logical '1' below $\sim$ 100mV. However, the mechanisms of failure suggest that the performance of the flip-flop can improve. For example, increasing the width of MP2 above minimum size could allow that FET to sustain the large leakage current I_a without requiring a larger V_{SD}. Additionally, increasing the size of the PMOS device in inverter I3 would hold N5 at a logical '1' for lower supplies. The sleep devices for the input buffer were sized for speed reasons during active mode, so they cannot be changed. Figure 3-5 shows the re-sized LFBFF correctly holding a logical '1' at a supply voltage of 80mV. Note that the output node and all the internal voltages settle to a steady state at the lower supply voltage and then return to the proper levels when the supply voltage rises again. The steady state values of the logical '1' nodes actually are well below V_{DD} at the lower supply voltage. Nevertheless, the nodes all return to the correct logical values when the supply voltage returns to higher values. This plot shows that careful sizing of the devices mentioned above can improve the ability to store a '1' at lower supply voltages. The next section will show that tradeoffs with storing a logical '0' prevent the improvements from going much below $\sim$ 80mV in this process and for the flip-flop sizing chosen for the testchip.

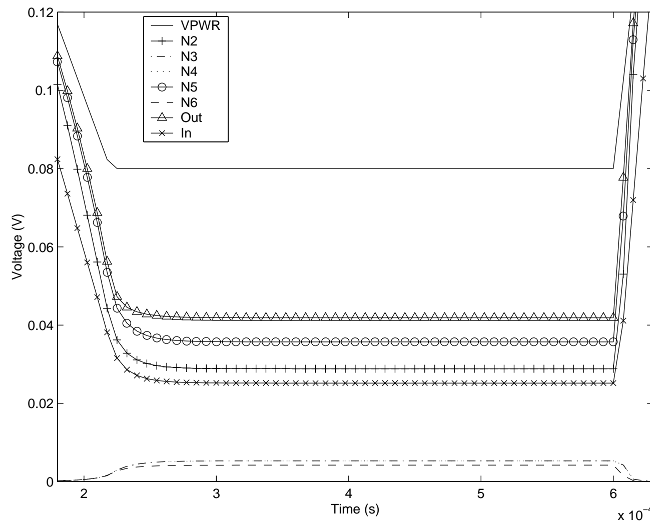


Figure 3-5: LFBFF Storing '1' at $V_{DD} = 80$mV

3.3.2 Storing a Logic '0'

The LFBFF consisting of minimum sized MP1, MP2, I3, and I4 stores a '0' successfully at extremely low V_{DD} values. Modifications made to improve the storage of a logical '1' degrade the ability of the flip-flop to store a logical '0'. These modifications consisted of upsizing MP2 and the PMOS device inside inverter I3. These changes directly address the causes of failure described in the previous section. Table 3.2 shows the state of the nodes in the flip-flop for a stored logic '0'. Again, the nodes

Table 3.2: LFBFF State for Stored '0'

Node Name	Value
N2	'0'
N3	'1'
N4	'1'
N5	'0'
N6	'1'
Out	'0'

in the flip-flop that deserve the most attention are those holding a logic '1'.

Figure 3-6 shows the LFBFF holding a logical '0'. As the supply voltage begins

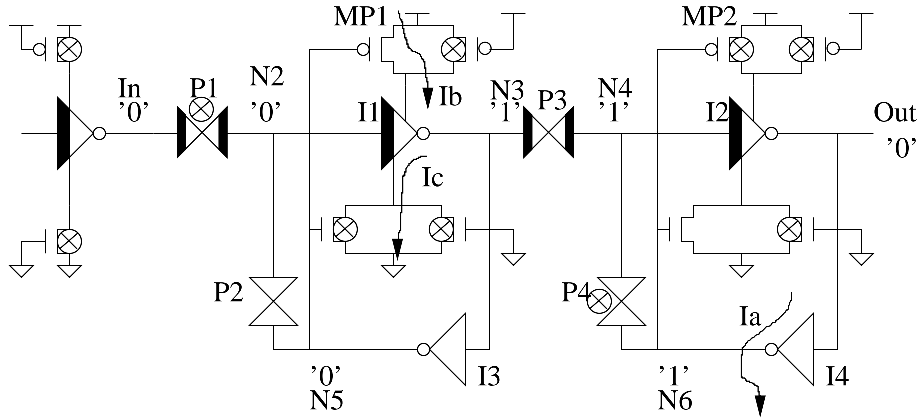


Figure 3-6: LFBFF Holding a '0'

to fall, the voltage at node N6 also falls. The current I_a in the figure shows the mechanism that causes the N6 voltage to fall below V_{DD}. As previously discussed, the PMOS inside inverter I4 cannot support the leakage current through the NMOS device at lower supply voltages without increasing its V_{SD}.

Figure 3-7 shows how the size of the PMOS in I4 affects the steady-state voltage

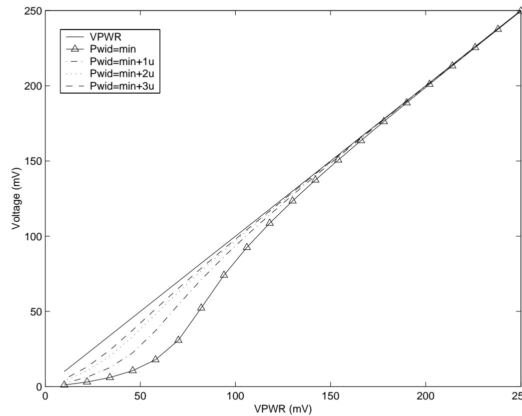


Figure 3-7: High V_T Inverter Output for Varying Widths

of N6. The figure shows the voltage at N6 for a standalone high V_T inverter with a few different widths for the PMOS. The input of the inverter is grounded explicitly for each case, and the outputs show the DC output of the inverter as V_{DD} scales toward 0. The plot indicates that the output of inverter I4 will begin to have a voltage

lower than the supply voltage once V_{DD} decreases beyond $\sim$ 100mV. A smaller width PMOS in the inverter creates a much wider gap between the output voltage (N6) and the supply voltage.

Once the voltage at N6 begins to drop, MP2 will see increased gate drive, and the NMOS sleep device will lose gate drive. As a result, the voltage at the output will rise above zero to maintain the match between the PMOS and NMOS currents. As shown in the previous section, the rising output voltage aggravates the effect shown in Figure 3-7 (where the input was grounded). The resulting feedback continues to drive N6 to an even lower voltage. Now it is apparent that increasing the size of MP2 to help the flip-flop store a logical '1' (as in the previous section) will cause the flip-flop to fail earlier while storing a '0'. This tradeoff requires that MP2 be sized with care so that the flip-flop fails for both a '1' and '0' at approximately the same voltage.

Figure 3-8 shows this failure occurring for a stored '0'. The plot shows waveforms for a LFBFF sized with minimum width I4 and MP1 but with increased MP2 and I3 PMOS to help with the store '1' case. The plot shows node N6 falling below the low supply voltage value of $\sim$ 80mV. As it falls, the output voltage begins to rise slightly. The voltages continue to settle, and N6 eventually dives for 0 while the output node gets pulled high. When the supply voltage returns to a higher value, the output node clearly tracks it as a logical '1'. Node N4 in the plot also contributes to flipping the state of the output. As N4 falls (the reason for this follows shortly), the low V_T PMOS inside inverter I2 increases its subthreshold current, and the NMOS in that inverter supports less current. This only helps MP2 to charge up the output node.

The other node that will cause failure for the LFBFF storing a '0' is N3. The same effects that caused the output node to fail for a storing '1' case will cause N3 to change to a logic '0'. We summarize those reasons again for this case. The voltage at node N3 falls along with the supply voltage. It actually begins to fall below the supply voltage because of the current mismatch that develops between I_b and I_c. In other words, voltage N3 must drop below V_{DD} in order for I_b to equal I_c. When it gets low enough, it begins to turn on the PMOS in inverter I3 in the master stage,

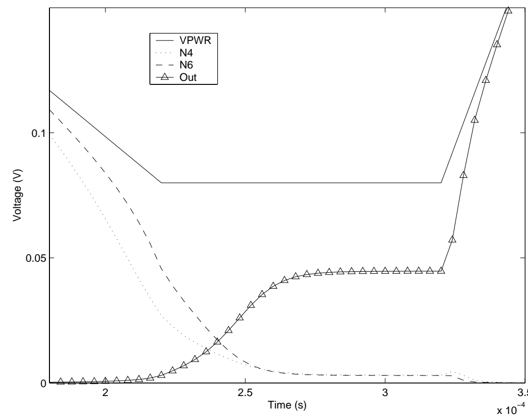


Figure 3-8: Output Node Changing Value for Stored '0'

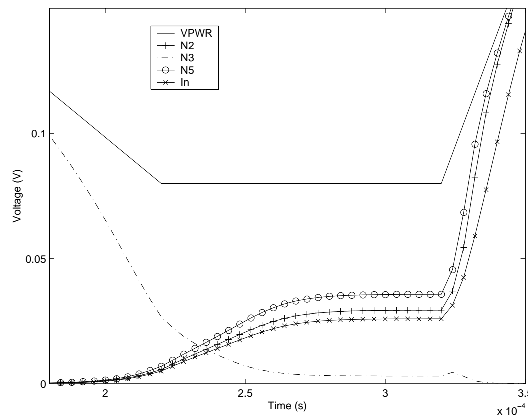


Figure 3-9: Internal Node Changing Value for Stored '0'

causing N5 to increase slightly in voltage. As N5 rises, it reduces gate drive for MP1 and increases gate drive in the NMOS sleep device. This feedback mechanism causes failure when the supply voltage lowers to the point that the current mismatch between I_b and I_c cannot maintain a '1' at N3. Node N3 must drop to a low value in order for I_b to match I_c.

Figure 3-9 shows this type of failure for the LFBFF storing a '0'. The plot comes from the same simulation that produced Figure 3-8. The plot shows the voltage at N3 dropping towards zero so that I_b remains matched with I_c. This change in voltage forces N5 to increase its voltage to maintain the current balance through inverter I3.

The rising N5 pulls N2 high through the high V_T passgate that is on. The input node also gets pulled high by the leakage current through the low V_T passgate P1.

Some simple sizing changes could improve the ability of the LFBFF to hold a '0' at low supply voltages. For example, increasing the size of MP1 would prolong the time that N3 would stay high as V_{DD} decreased. Also, increasing the size of the PMOS device in inverter I4 would help to maintain N6 at a logical '1'. The tradeoffs involved with these sizing changes are readily apparent. Table 3.1 and Table 3.2 show that the nodes inside the flip-flop hold opposite values for the two different states. This means that any sizing change that helps to maintain one state will degrade the ability of the flip-flop to hold the other state. Essentially, this tradeoff requires careful sizing to adjust the failure points for each state to be the same.

Figure 3-10 shows the flip-flop successfully holding a '0' at $V_{DD} = 80$mV. This implementation of the LFBFF also successfully stored a '1' at $V_{DD} = 80$mV, as shown in Figure 3-5. Notice in Figure 3-10 demonstrates that the output voltage and the

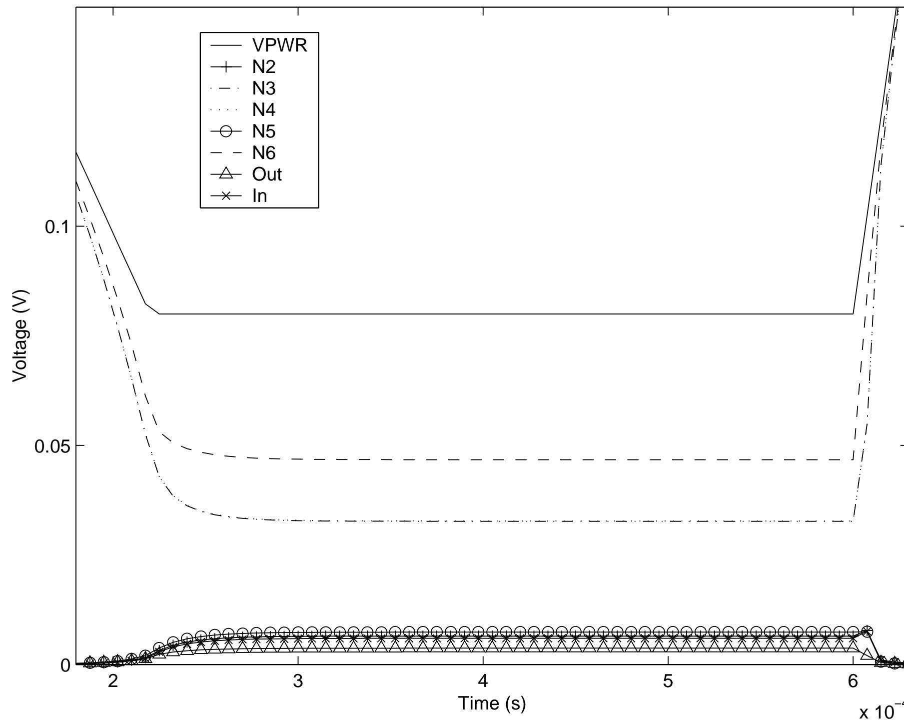


Figure 3-10: LFBFF Storing '0' at $V_{DD} = 80$mV

internal voltages all retain their correct values when the supply voltage begins to rise. The LFBFF shows the ability to retain either state with a supply voltage of 80mV in this process. The flip-flop can perform fairly robustly at a supply voltage of 80mV. Although capable of operating below this voltage, any lower operation becomes extremely susceptible to coupling noise, etc. For example, the rate at which V_{DD} rises can dramatically affect functionality for lower supply voltages.

3.3.3 Power Savings Using Voltage Scaling

Scaling the supply voltage during standby mode provides power savings in several ways. First, the power dissipated by the flip-flops decreases linearly with the sup-

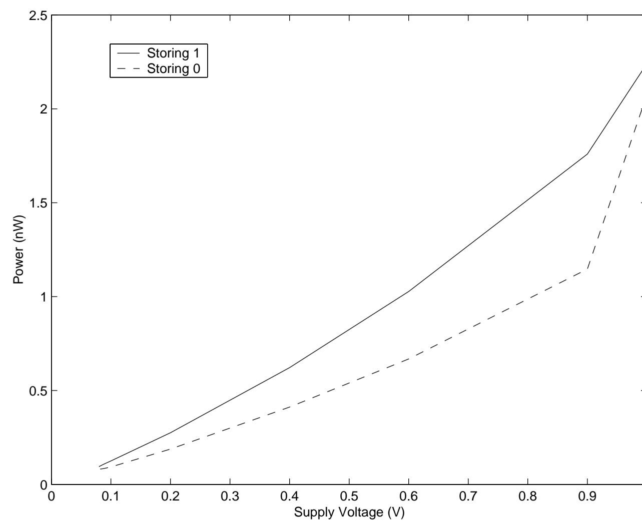


Figure 3-11: Power Reduction from Voltage Scaling

ply voltage. Furthermore, the subthreshold current leaking through the flip-flops decreases exponentially with supply voltage according to the DIBL effect. As V_{DD} becomes quite small, the effect of the exponential term in parentheses in the BSIM2 model reduces subthreshold current even further. Figure 3-11 shows the power decreasing as the supply voltage scales down. A simulation for a single LFBFF sized for the testchip application produced this plot. The figure shows power reduction for both storage states. The minimum point on this plot occurs at 80mV, which is the

lowest voltage where the flip-flops from the testchip can store both states successfully.

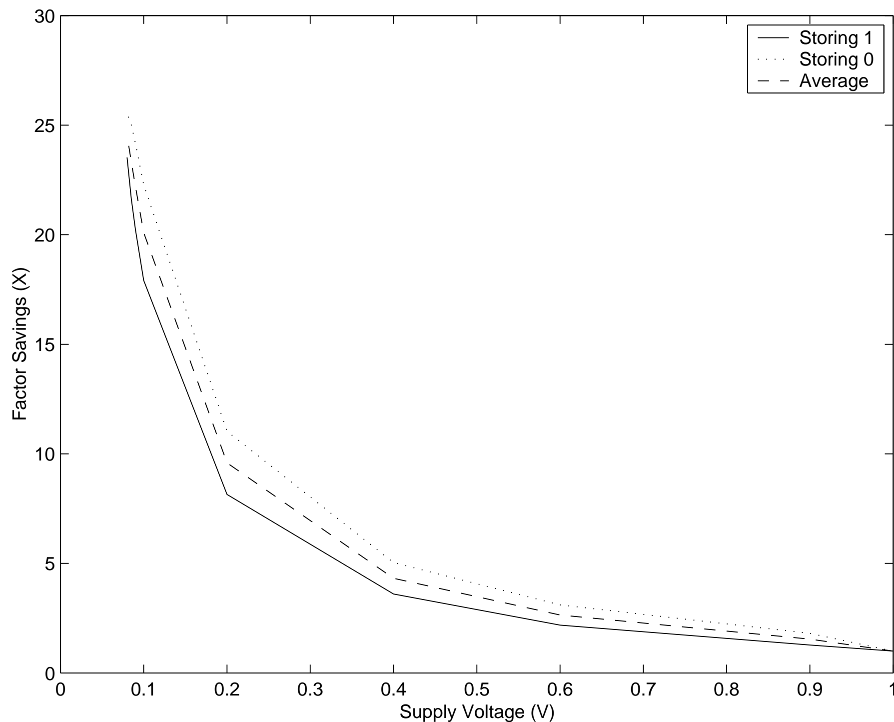


Figure 3-12: Power Reduction Factor from Voltage Scaling

The reduced power shown in Figure 3-11 translates into fairly significant reductions in power consumption. The Power Reduction Factor is the ratio $\frac{Power at V_{DD}=1.0V}{Power at lower V_{DD}}$. Figure 3-12 shows the Power Reduction Factor for the flip-flops on the testchip. The plot demonstrates an average power reduction of $\sim$ 25X at V_{DD} = 80mV. The steepness of the curve at lower supply voltages indicates that savings increase dramatically once V_{DD} can scale below about 200mV.

3.3.4 Levels of Standby

The addition of supply voltage variation to the traditional MTCMOS bag of tricks creates several modes of standby operation. Table 3.3 provides an example set of standby states using the techniques described. Clearly, a number of variations exist for defining standby states. Entering the deeper standby states requires more effort

Table 3.3: Possible Set of Standby States

Standby Level	Description	Holds State
Active	Normal Active Operation of Circuits	✓
Standby 1	Clock Gated and No Sleep	✓
Standby 2	Clock Gated and Sleep Asserted	✓
Standby 3	V_{DD} Scaled to $<$ 100mV for FFs, and $V_{DD} = 0$ for logic	✓
Standby 4	All V_{DD} scaled to 0	

in terms of control signals and complexity. It also has higher overhead power costs. However, the deeper states provide greater power savings.

For example, simply gating the clock eliminates dynamic power consumption but does not address subthreshold leakage. Asserting the sleep signal reduces the subthreshold leakage of the circuit. This mode requires more time in standby so that the leakage currents can settle to their lower values. Asserting sleep also uses some overhead energy when the sleep devices are switching and as some of the gated nodes charge up to V_{DD}. Voltage scaling can provide significant power savings, but it carries an even greater overhead cost. For example, all of the nodes in the combinational logic get discharged to zero when the logic supply is disconnected. The decision to enter this deep sleep state must account for the energy required to recharge the combinational logic when exiting sleep mode. Also, the overhead circuitry associated with varying the power supplies expends extra energy. Energy savings only occur when the circuit spends enough time in deep sleep to save more energy than it costs to enter the mode in the first place. Obviously, the deepest sleep state simply disconnects all of the circuit power supplies. The flip-flops cannot save their state in this mode.

The cost of entering different levels of sleep deserves special attention in a design using voltage scaling. The overhead costs depend strongly on the size of the particular circuit (number of combinational and sequential gates) and on the variable power supply itself. Consideration of the overhead costs can allow the control circuitry to maximize the lifetime of a circuit by minimizing its power.

3.4 Adaptively Scaling Supply Voltage

The previous sections show that scaling the supply voltage to flip-flops while disconnecting the combinational power can provide substantial power savings. The power savings increase rapidly when V_{DD} reaches very low ($<$ 100mV) levels. Unfortunately, the ability of the LFBFF to retain its value correctly becomes questionable at lower supply voltages. Even though the LFBFF may show correct behavior in simulation, many other factors could cause failure on an actual chip. For example, changes in temperature, process variation, power supply noise, and other similar variations could cause the flip-flops to fail at low voltage. These considerations suggest that a cautious higher supply voltage is advisable. On the other hand, the power supply ideally should scale as low as possible to provide the maximum savings.

3.4.1 Architecture for Adaptive Flip-Flop Voltage Scaling

Figure 3-13 shows an architecture that adjusts the supply voltage adaptively to maximize power savings while ensuring correct functionality. The system in the figure

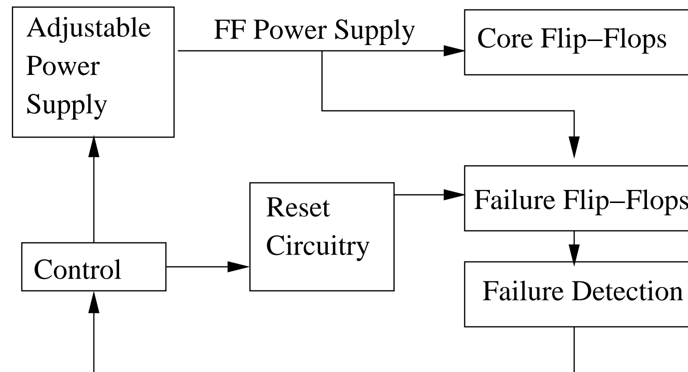


Figure 3-13: Adjustable Voltage Scaling Architecture

relies on a group of "failure flip-flops" as an indication of the lower bound on supply voltage. The system works as follows. Once the core flip-flops are in sleep mode, the control block signals the adjustable power supply to begin lowering the FF supply voltage. As the voltage scales, the control block monitors the failure flip-flops for a

failure. When a failure occurs, the control block re-adjusts the power supply to a higher value and resets the failure flip-flops. The system depends on the fact that the failure flip-flops will fail at a higher supply voltage than the core flip-flops. That way, the control logic block can adjust the supply voltage to a point that nears the failure point of the core flip-flops without actually causing them to fail.

The use of LFBFFs designed to fail at higher voltages than the core flip-flops enables the voltage scaling process to adapt to variations in process or environment. Any process variation that affects the core flip-flops should affect the failure flip-flops in the same way. Also, any variations in temperature or power supply will affect both types of flip-flops. The difference between the failure voltage of the failure flip-flops and the failure point of the core flip-flops is adjustable by design. A larger difference provides a more robust system in terms of functionality at the cost of power savings. Reducing the difference in failure voltages provides additional power savings, but it increases the risk of core failure. The next section discusses the design of the failure flip-flops.

3.4.2 Designing Flip-Flops for Failure

The adaptive voltage scaling system requires flip-flops that fail consistantly at a higher voltage than the flip-flops in the core circuit. The previous analysis of mechanisms of failure for the LFBFF facilitates design of the failure flip-flops. First consider the LFBFF holding a logical '1'. The two critical nodes in Figure 3-2 are the output node and N5/N2. The flip-flop fails when the PMOS devices that supply current to these nodes cannot maintain the current drawn by the NMOS devices without reducing the node voltage toward zero. This analysis previously supplied a method for sizing the non-critical PMOS devices in the circuit to increase the functional range. Reversing the lessons applied in that case gives a method for sizing a LFBFF to fail while holding a logical '1'.

Specifically, the PMOS device in inverter I3 and MP2 hold the '1' nodes high. Making these devices minimum size minimizes their ability to maintain the logical '1' voltages. Conversely, the PMOS device in inverter I4 and MP1 act to pull the '0'

nodes high. Increasing the size of these devices causes the flip-flop to fail at higher voltages. Figure 3-14 shows the simulation of a LFBFF sized to fail at 130mV. The figure shows the critical nodes in the flip-flop at 140mV, 130mV, and 120mV. The flip-flop's output voltage transitions incorrectly to a logical '0' for supply voltages of

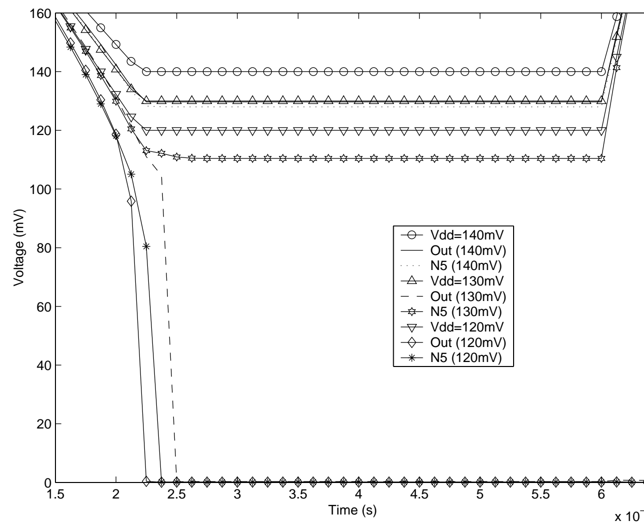


Figure 3-14: LFBFF Storing '1': Sized to Fail at 130mV

130mV and 120mV. The internal node N5 transitions incorrectly at 120mV. All of the flip-flop nodes behave correctly at V_{DD} values of 140mV and above. This analysis assumes that the adjustable power supply has a reliable granularity of 10mV. The figure confirms that the LFBFF used in the testchip can be sized to fail at 130mV, which is 50mV higher than the core flip-flop failure point.

The analysis for a LFBFF holding a logical '0' is similar. In this case, the critical nodes are N3/N4 and N6 from Figure 3-6. These nodes have a logical '1' value that transitions to zero when the node fails. The same four PMOS devices from the previous case can change the failure point for a flip-flop holding '0'. In this case, however, the PMOS device in inverter I4 and MP1 are minimum sized. The sizes of the PMOS device in inverter I3 and MP2 are increased to alter the failure point. Figure 3-15 shows a LFBFF sized to fail at 130mV while holding a logical '0'. Again, the plot shows the critical nodes in the flip-flop at 140mV, 130mV, and 120mV. Both

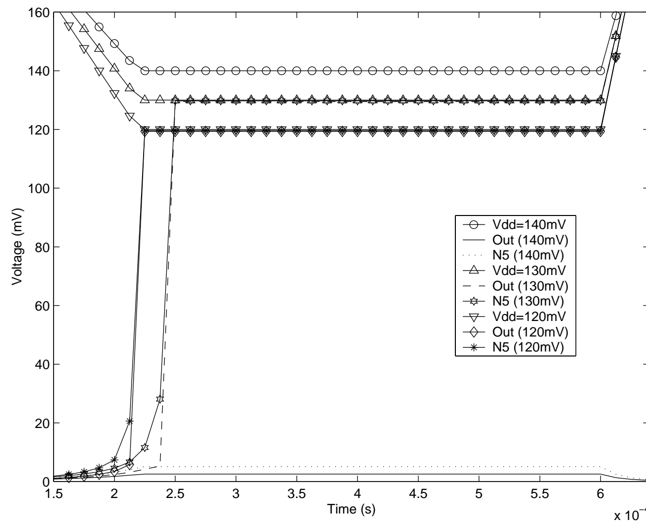


Figure 3-15: LFBFF Storing '0': Sized to Fail at 130mV

the output node and the critical internal node transition for supply voltages of 130mV and lower. The FF correctly holds a '0' at 140mV.

The two failure flip-flops that produced the above plots are sized identically to the core LFBFFs except for the PMOS device in inverter I4, the PMOS device in inverter I3, MP1, and MP2. These four PMOS devices do not affect the flip-flop's speed in the active mode. Simply resizing these four devices can increase the failure voltage of the core LFBFF by over 50mV. The size of the gap between failure voltages in the core device and the failure devices is a design decision that depends largely on the accuracy of the adjustable voltage supply. Resizing these four PMOS devices provides a decent range of values for the gap. The failure flip-flops can provide an arbitrary gap size, however, by resizing other devices in the flip-flop. The analysis of LFBFF failure maps out which devices need resizing to increase the failure voltage in a given state.

3.4.3 Effect of Variation on Flip-Flops

As previously mentioned, adaptive supply voltage scaling depends on the ability of the failure flip-flops to fail before the core flip-flops fail to maintain its reliability. The

previous section showed failure flip-flops designed at the TT corner to fail $\sim$ 50mV higher than the core flip-flop for both states. These failure flip-flops can only be used in practice if they continue to fail at a higher supply voltage than the core flip-flops in the face of variation. This section demonstrates that the failure flip-flops continue to fail before the core flip-flops under temperature and process variation.

Figure 3-16 shows the effect of temperature on the failure voltage for the core and failure flip-flops. In both cases, higher temperatures cause the flip-flops to fail at a higher supply voltage. The failure point of the failure flip-flops increases faster with temperature than for the core flip-flops. This occurs because the failure flip-flops have a larger disparity in the sizes of devices inside the flip-flop. The increasing temperature magnifies the offset of currents between devices of different size more than between devices of roughly the same size. The figure shows that the failure

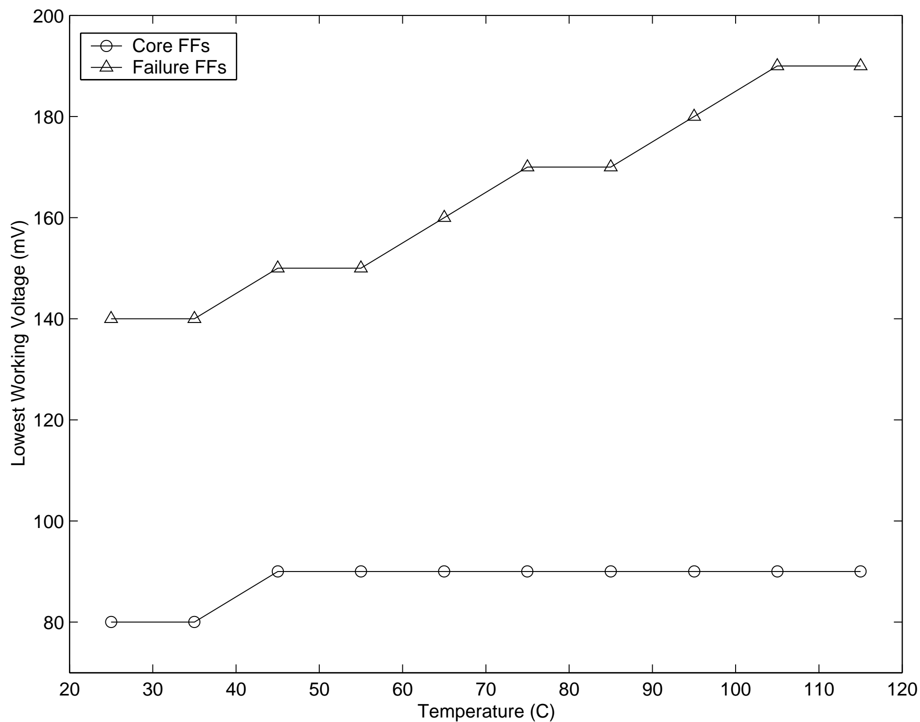


Figure 3-16: Effect of Temperature on FF Failure

flip-flops continue to perform correctly at higher temperatures. The devices used for the simulation are the same designs that appear earlier in the chapter. The points

on the plot show the lowest supply voltage (with a 10mV granularity) for which the flip-flops correctly hold their state.

Process variations also could alter the behavior of the failure flip-flops and the core flip-flops. The flip-flop designs described earlier in the chapter were sized at the TT process corner. Figure 3-17 shows the lowest supply voltage at each corner for which the flip-flops hold their state. Again, the failure flip-flops fail at a higher supply

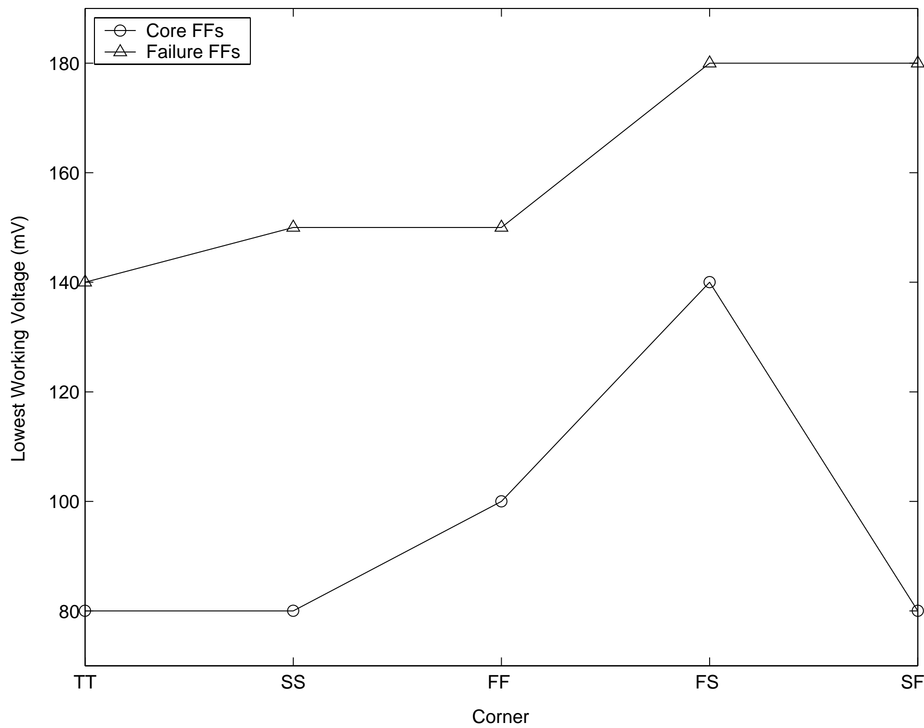


Figure 3-17: Effect of Process Corners on FF Failure

voltage than the core flip-flops at every corner.

These two examples show that a feedback system based on failure flip-flops can continue to function robustly in the presence of temperature and process variation. The difference in failure voltages should remain acceptable as long as variations are common to both types of flip-flop. For example, power supply noise should not cause a problem because both flip-flop types share the same power supply. The failure flip-flops might not maintain the buffer between failure voltages in the presence of irregular variations. For example, if process variations within a single die affected failure flip-

flops and core flip-flops differently, there could be problems. The plot shows that the core flip-flop at the FS corner fails at the same voltage as the failure flip-flop at the TT corner. Positioning the failure flip-flops in close proximity to the core flip-flops reduces the likelihood of negative effects from irregular process variation. A comprehensive understanding of the sources of variation in a given application would allow designers to size the failure flip-flops to provide a sufficient cushion for functionality despite irregular variations.

Chapter 4

Testchip Design and Architecture

4.1 Introduction

This chapter provides a top-level description of the testchip. It details the architecture of the design and the functionality of the circuit under test. This chapter intends to establish the design of the testchip along with high-level architectural decisions rather than to explain technical details of the design. The subsequent chapter describes interesting technical issues in more detail.

4.2 Subthreshold Leakage and FPGAs

This thesis project is incorporated with two other projects whose collective goal is the design of a low power Field Programmable Gate Array (FPGA) architecture. While other group members explore new bus coding techniques and low power logic block architectures, this thesis applies subthreshold leakage reduction methods to FPGA logic.

FPGAs traditionally appear in systems that do not have stringent power constraints. Some of the traditional benefits of FPGAs suggest that they would be a strong competitor for inclusion in battery operated devices. For example, the low cost of an FPGA makes it ideal for a small or medium volume project. An FPGA's programmability places design time well below that of an ASIC and thus reduces the

time to market. Programmability also allows for easy upgrades in the field. These advantages would make FPGAs an attractive option for the designers of various battery operated devices except for the traditionally high power consumption of an FPGA. Existing work has examined the active power dissipation of FPGAs, but the standby power dissipation has largely escaped attention. FPGA designs suffer the same effects as other logic when implemented in a "leaky" process such as 0.13μm. Since many handheld communications devices spend significant time in standby mode, a low power FPGA design stands to benefit from subthreshold leakage reduction techniques.

As opposed to other leakage reduction techniques that depend on unconventional processes or on multiple power supplies, MTCMOS only requires a process having two or more threshold voltages. Since multithreshold processes are becoming more common, MTCMOS appears to be a commercially viable option for reducing leakage power. This work applies MTCMOS techniques to reduce leakage currents in a state-of-the-art 0.13μm multi-threshold process.

4.3 Test Chip Circuits

The testchip designed for this thesis contains circuits for testing and demonstrating the subthreshold leakage reduction techniques previously mentioned. It applies these techniques in a cutting edge commercial process (0.13μm).

The circuit under test is a simplified version of an FPGA. The FPGA arcitecture groups 12 Configurable Logic Blocks (CLBs) into 3 slices. The CLBs inside these slices can be connected in a number of ways based on bits used to configure the interconnect circuitry. This small FPGA architecture can provide a number of different functions, but the function of primary interest in this project is an 8-bit filter with 16 bits of precision. The testchip includes test circuitry that generates inputs for the filtering operation and that captures the output. All configuration bits and computational results will be transferred to and from the chip using a JTAG interface.

4.3.1 Top Level Design

Figure 4-1 shows the top level of the test chip setup including the JTAG interface. The JTAG control signals will come from off chip to manage the shifting of data to and from the chip. The CORE of the testchip design contains the three slices and interconnect logic. This CORE block performs all of the computation in the filter. The leakage reduction techniques under examination are used inside the CORE block. The DREGISTERS block and the PERIPHERALS block contain the overhead circuitry necessary for configuring and testing the CORE. These overhead circuits operate on their own power supply (at the same voltage as the CORE) so that the core power consumption can be measured separately. The following sections provide

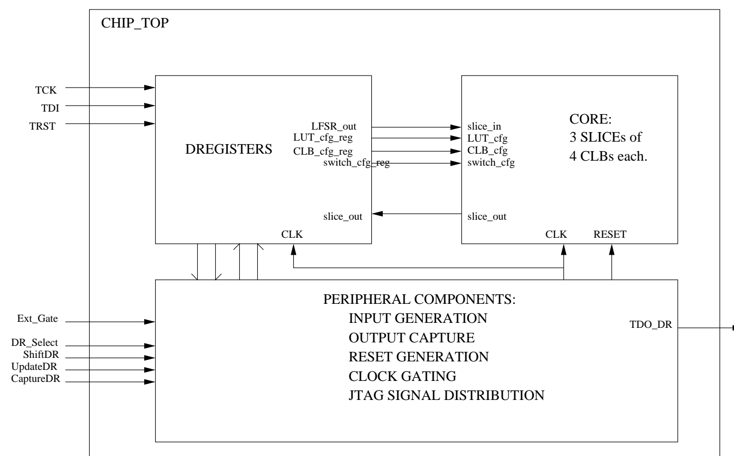


Figure 4-1: Test Chip Setup

more detail about the architecture of each component on the testchip.

4.3.2 Core

Configurable Logic Block

The actual CLB architecture is a design focused on providing filtering and/or arithmetic operations for low power FPGAs using Distributed Arithmetic (see Figure 4-2). This new CLB design was developed by Frank Honore.

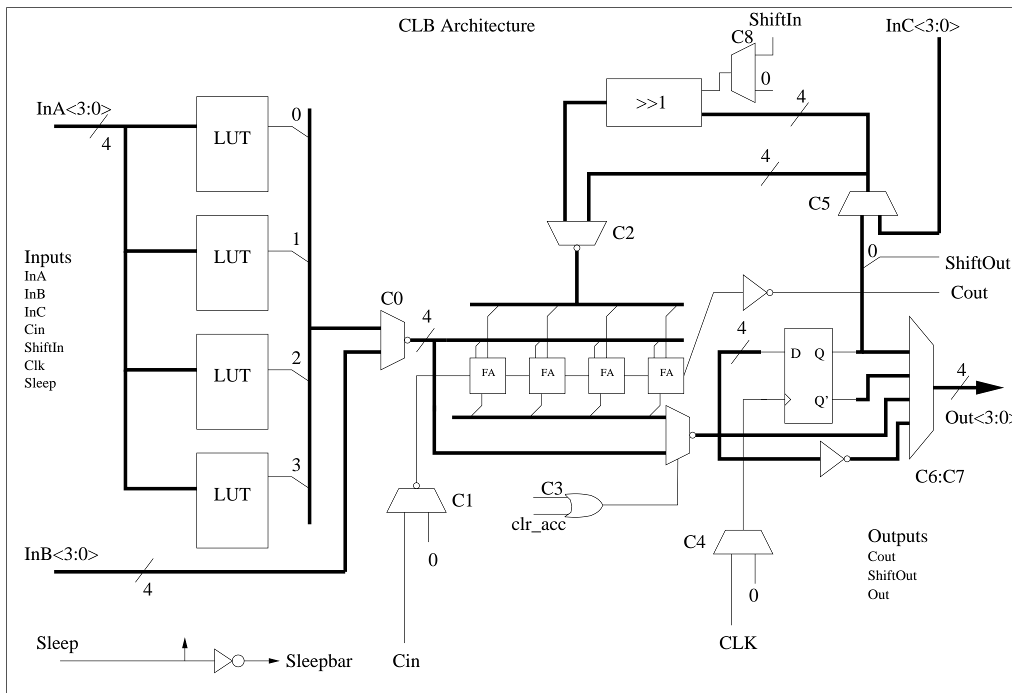


Figure 4-2: Configurable Logic Block Used on the Test Chip

The CLB consists of three primary components. These are the four 16-bit Look-Up Tables (LUTs), the 4-bit adder, and the 4-bit register. The outputs of the register can feedback either directly or in shifted form to the input of the adder. This feedback loop together with the LUTs enables a single CLB to perform a multiply-accumulate (MAC) operation with four bits of precision. The CLB has three different inputs consisting of four bits each. *InA* inputs directly to the LUTs. In a MAC operation, the LUTs hold the result of a multiplication with the input, *InA*. *InB* provides a route for bypassing the LUTs and accessing the adder directly. *InC* provides a second input that can reach the adder while bypassing the register. Clearly, the adder also can be bypassed with the correct configuration bits.

The configuration of the CLB controls its function. The configuration bits do not change in value except during a reprogram. Reprogramming the CLBs on the testchip entails shifting in a new vector of CLB values through the JTAG interface and then latching that vector into the configuration registers associated with each CLB. The configuration bits act as the inputs to multiplexors with one exception. The

Table 4.1: Signals Multiplexed by Configuration Bits

Configuration Bit	InputA	InputB	Bits Switched
0	LUTout	InB	4
1	Carry In	0	1
2	Feedback	Shifted Feedback	4
3*	Adder Sum	Adder Ain	4
4	Clk	0	1
5	Reg Out	InC	4
6,7	Q, Qbar	FFin, FFinbar	4
8	ShiftIn	0	1

*bit is OR-ed with a clear signal - see text

multiplexor that selects between the adder output and the LUT output is controlled by the logical OR of a configuration bit and a clear signal (*clr_acc*). This clear signal pulses high at the appropriate time to clear the accumulator during a filtering operation (see Figure 4-9). Table 4.1 shows the function of the configuration bits.

Slice

In the testchip design a slice contains four CLBs. Figure 4-3 shows the interconnections of CLBs within a slice. Most of the inputs to the CLBs are either shared (*i.e.* - CLK) or unrelated (*i.e.* - *InA* for CLB<0>and *InA* for CLB<1>). Maintaining the independence of the InX inputs to the CLBs permits interconnect above the slice level to provide any desirable connectivity. For convenience sake in this testchip, the carry and shift signals were hardwired at the slice level to allow for 16-bit addition and shifting within the slice. In a final FPGA design, the routing of these signals would also most likely be programmable.

As the figure shows, the outputs of the CLBs are buffered at the slice level. This buffering provides the additional drive strength necessary to propagate output signals through the switchable interconnect and into the inputs of other slices. Not shown in the figure are buffers at the *InA* inputs on the slice level.

The slice includes a 36-bit configuration register that holds the 9-bits of configuration required for each of the four CLBs. This configuration register consists of

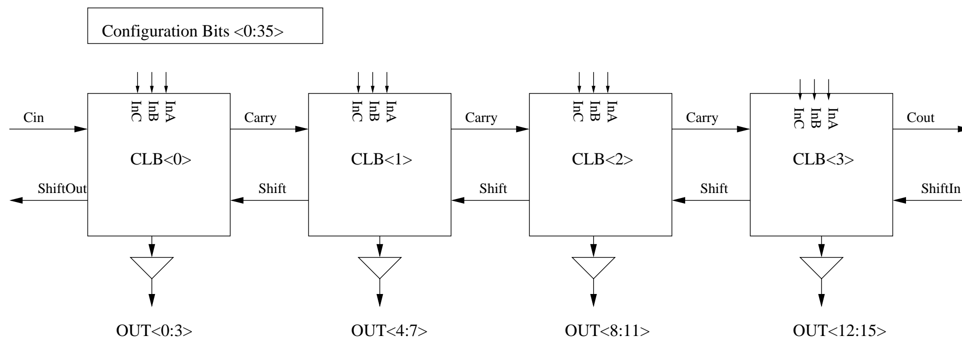


Figure 4-3: Slice Diagram

simple latches. The latches do not require significant drive strength since they will not change their value during the course of normal circuit operation.

Interconnect

One of the most critical elements of FPGA design is the interconnect between logic blocks. Interconnect must be programmable to provide the desired connectivity for different logic implemented on the FPGA. For this reason, the interconnect must be switchable. Many designs of FPGA interconnect have appeared over the years, but this testchip employs a straightforward transmission gate approach for the sake of simplicity.

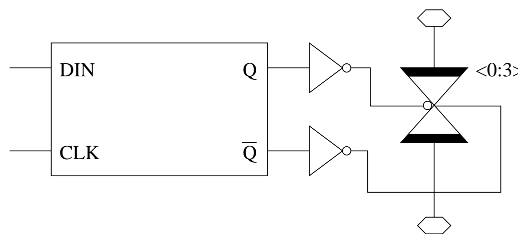


Figure 4-4: Programmable Transmission Gate for Interconnect

In this approach, an open transmission gate forms the connection between two wires while closed transmission gates prevent other wires from driving those lines. Figure 4-4 shows how a simple latch associated with each set of four transmission gates

provides the programmability for changing connectivity. In a final FPGA architecture, an SRAM bit would likely replace the latch to reduce area and power consumption. The latch suffices in this simple design. One advantage to the latch approach is its relatively small power consumption due to subthreshold leakage. The latch itself can consist of minimum sized, high V_T devices since speed is not an issue. Some of the transmission gates, on the other hand, will lie on the critical path if signals are routed between slices. For this reason, the transmission gate uses low V_T devices to increase speed. Despite the significantly greater subthreshold leakage in low V_T devices, the low V_T transmission gates do not actually increase the total subthreshold leakage for the core because their sources and drains are not directly connected to power or ground. So long as the drivers that connect to the gated lines are properly disconnected from the power rails in standby mode the transmission gates cause no increase in subthreshold leakage current (see Section 5.4.2).

Table 4.2: Mapping of Slice Signals to Buses

Slice Signal	Bus Mapping
InA0	hbus0<0:3>
InA1	hbus0<4:7>
InA2	hbus0<8:11>
InA3	hbus0<12:15>
InB0	hbus0<0:3>
InB1	hbus0<4:7>
InB2	hbus0<8:11>
InB3	hbus0<12:15>
InC0	hbus1<16:19>
InC1	hbus1<20:23>
InC2	hbus1<24:27>
InC3	hbus1<28:31>
Out0	hbus2<32:35>
Out1	hbus2<36:39>
Out2	hbus2<40:43>
Out3	hbus2<44:47>

Table 4.2 shows the mapping from slice signals to interconnect buses. Note that the *InA* and *InB* inputs are tied together because only one can be used at a time. All

of the signals from a slice appear in three 16-bit buses: hbus0, hbus1, and hbus2.

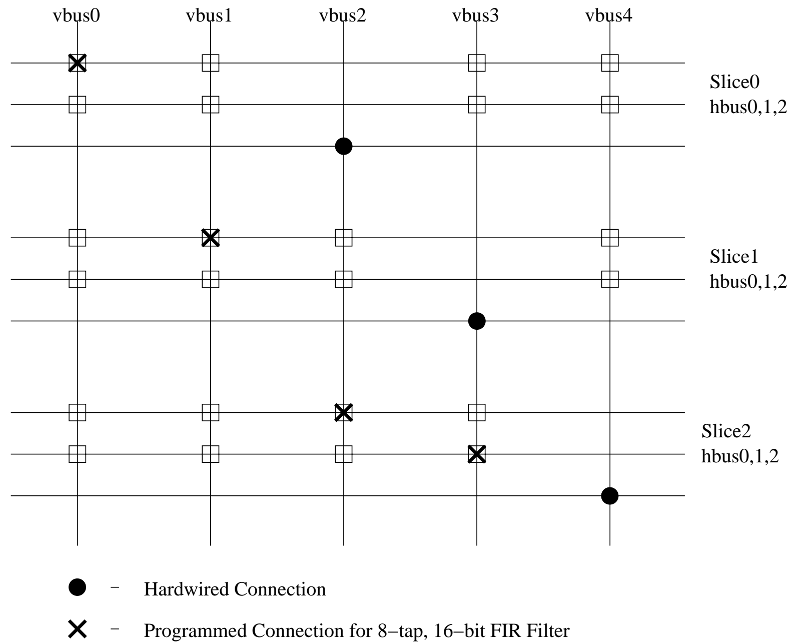


Figure 4-5: Top Level Switching Diagram

Figure 4-5 shows the interconnection among the three slices. The nine total hbus lines cross with five, 16-bit vertical buses. For simplicity, each of the outputs (hbus2) from the slices is hardwired to its own vertical bus (denoted by a dot in the figure). The boxes in the figure show the location of interconnect switches. Each of these interconnect switches can be programmed, using 4 bits, to connect the two 16-bit buses in 4-bit segments. In other words, hbus0<0:3>can connect to vbus0<0:3>while hbus0<4:7>remains disconnected from vbus0<4:7>. This interconnection scheme provides essentially full connectivity among the slice outputs and inputs.

The bold **X**'s in Figure 4-5 show the switches that are connected for the application of particular interest in this testchip. The next section provides some more detail about this application.

Core Operating as 8-tap, 16-bit Filter

As previously mentioned, the application chosen for testing the FPGA core is an FIR filter. The slices implement this filter using distributed arithmetic. The filter has 8 taps (or 8 consecutive, non-zero samples), and the final accuracy of the filter is 16-bits. Figure 4-6 demonstrates the connectivity of the slices to produce the filtering operation.

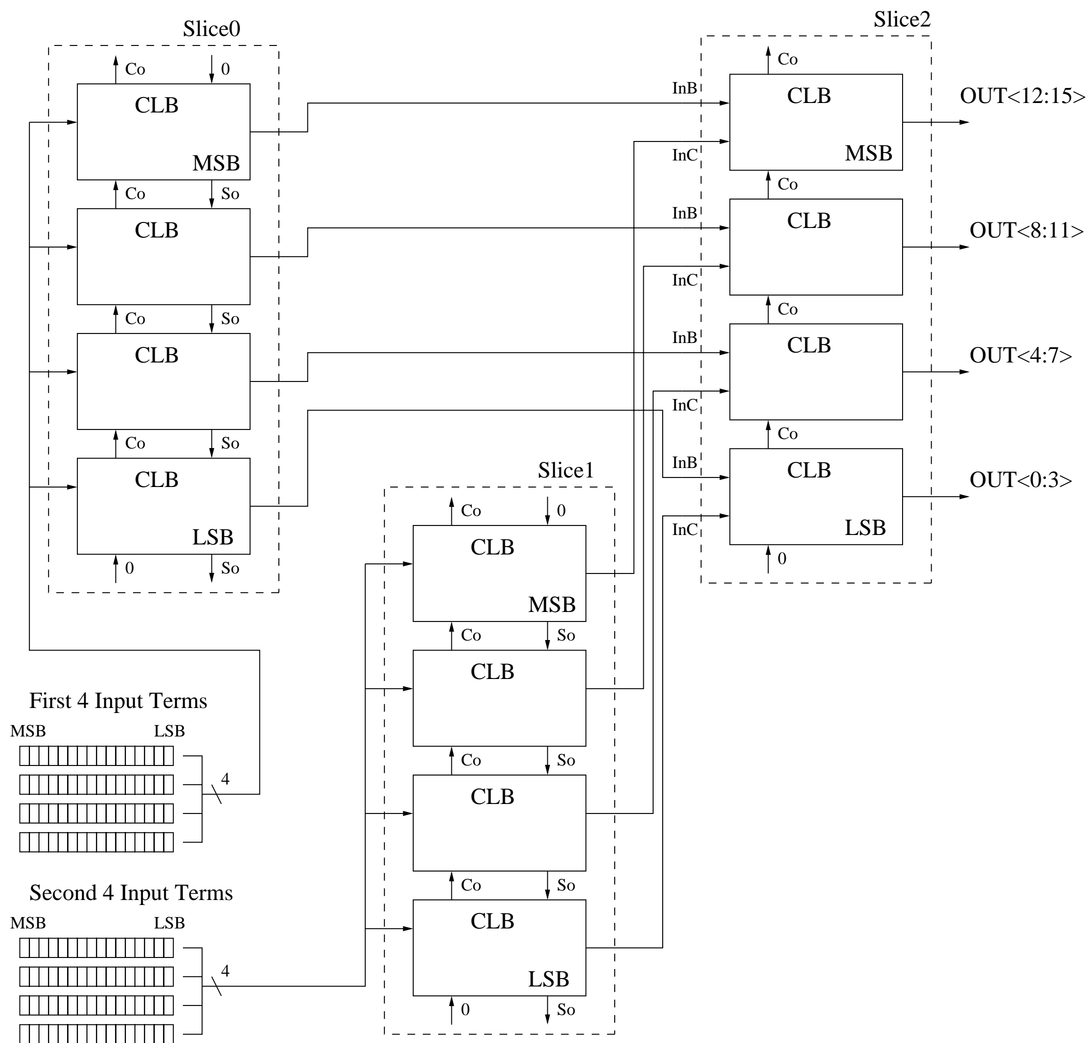


Figure 4-6: Slices Implementing an 8-tap, 16-bit FIR Filter

The interconnect configuration marked by **X**'s in Figure 4-5 shows how this connectivity is achieved in the testchip design. Basically, the outputs of the first two

slices are routed to the *InA* and *InB* inputs of the third slice, whose CLBs are configured to perform a 16-bit addition. The 16 bits of the eight input terms are applied as *InA* to the first two slices as shown in Figure 4-6 over the course of 16 cycles of the clock. On the seventeenth cycle, the final sum is the 16-bit result of the filtering operation. The section on the peripheral blocks describes the input generation and output capturing techniques in greater detail.

4.3.3 Data Registers

As previously mentioned, the testchip uses a JTAG interface to transfer data to and from the chip. The JTAG protocol uses a serial link to shift data into on-chip shift registers. Once the shift registers contain the data, other registers on the chip can load it to hold for later use. Table 4.3 shows the shift registers that are used on the testchip. A check in the input column indicates that the register can be loaded from offchip, and a check in the output column indicates that the register's contents can be shifted offchip. These registers are abstracted in the DREGISTERS block in Figure 4-1.

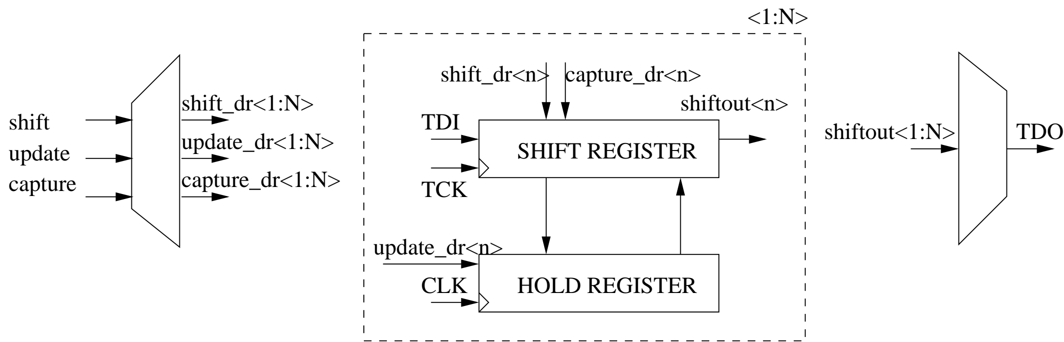


Figure 4-7: Data Register Diagram

Figure 4-7 shows the basic setup for each of the registers in Table 4.3. The serial JTAG input *TDI* connects to the shift-in port of every shift register. The signals that control shifting and latching are demultiplexed to the correct shift register using a data register address that off-chip control logic generates. The same address selects

Table 4.3: Shift Registers on the Testchip

Register Name	Size (bits)	Input	Output
LUT configuration	768	✓	✓
Switch configuration	96	✓	✓
CLB configuration	108	✓	✓
Cycle Count	9	✓	✓
Filter Input	113		✓
Filter Result	16		✓

the correct output bit from the shift registers for transfer off-chip. The *shift* signal tells the shift register to shift its contents at each clock edge. As Figure 4-7 indicates, the shift registers operate on a different clock than the testchip core. *TCK* can be rather low frequency since it does not perform any function during active operation. Its only use is loading the shift registers. The *update* signal tells a holding register to latch the contents of the shift register once it holds the correct value. Since the majority of the shift registers in the testchip design operate on a different clock than their associated holding register, very few *update* signals are used. The *capture* signal tells the shift register to capture the data from its holding register prior to the onset of shifting. This ensures that the bits shifting off of the chip come from the core. For example, the Filter Result register will capture the actual result from the core into the shift register before shifting.

4.3.4 Peripheral Components

The testchip includes circuitry that facilitates testing its functionality. As previously mentioned, the intended function for test on the chip is an 8-tap, 16-bit FIR filter. The top-level diagram of the testchip (Figure 4-1) shows these peripheral components that will enable easy testing. This section discusses those peripheral components whose purpose is to enable testing of the testchip.

Input Generation Block

An 8-tap FIR filter operates on eight input terms simultaneously. These eight inputs are multiplied by the tap values and then added together. The distributed arithmetic algorithm accomplishes this computation over 16 cycles for 16-bit precision samples. After shifting the 16 bits of each term in as input to the slices for the first sample, the inputs terms are shifted, and the computation for the next sample begins. Figure 4-8 shows the hardware used for this task and provides snapshots of the process at several different times.

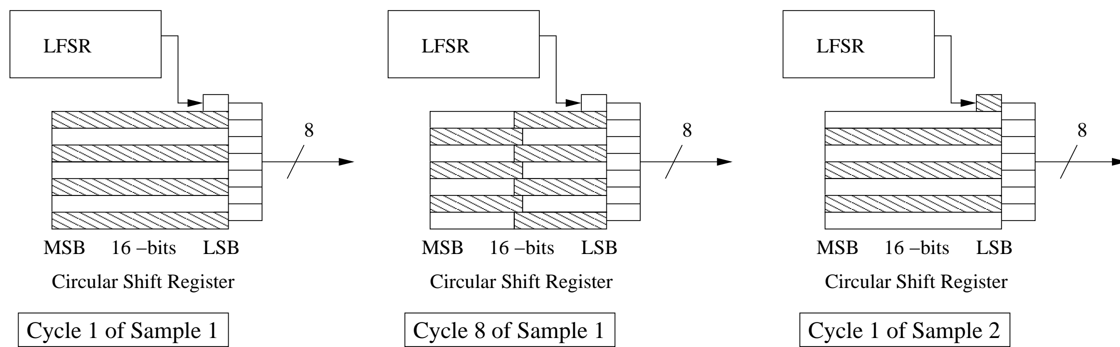


Figure 4-8: Input Generation Hardware and Operation

Linear Feedback Shift Register (LFSR) A Linear Feedback Shift Register (LFSR) is a component that cycles through a given number of unique states before returning to the same state. In other words, it is a counter that counts in a semi-random order. The 8-bit LFSR used in this design cycles through the 256 8-bit words before repeating the cycle. The order of appearance for the 8-bit words is semi-random but deterministic (the same sequence always appears for a given starting state). The testchip uses one bit of the LFSR as a semi-random input to provide the bits of the input terms to the filter. As shown in the figure, the LFSR inputs this bit to the Circular Shift Register.

Circular Shift Register The circular shift register provides the function of storing the 16-bit input terms and shifting them in the correct manner. Figure 4-8

shows that the circular shift register consists of 8 rows of 16 bits each (but the first row has only a few bits since the LFSR produces its input). Note that alternate rows are crosshatched for clarity. The 16 bits in each row shift from left to right to produce the 8-bit signal that inputs to the filter core. When a bit at the right end of the row shifts past the LSB position, it moves to the MSB position of the next row. The center snapshot shows the state of the input generation circuitry during the eighth cycle of filtering. Each of the eight terms has shifted to the right by 8 bits, so those bits have entered the filter as inputs and then shifted down to the next row. The rightmost snapshot shows the state of the circular shift register at the beginning of the filtering operation for the next sample. By this time, the first seven, 16-bit terms from the first sample have shifted down a row. The original eighth term has shifted completely out of the register, and the LFSR is producing a new first term. This hardware produces the inputs necessary for the filtering operation. Of course, the inputs also can serve as pseudo-random 8-bit inputs for any other configuration of the core as well.

Output Capture Block

Since the filtering operation of interest has 16-bit precision, a new 16-bit output sample is produced every sixteen cycles. The output capture block provides a means for latching this 16-bit result at the correct time. It also generates a clear signal that resets the contents of the accumulators in the CLBs performing the MAC operations. Figure 4-9 shows the timing of the signals required to perform these tasks. The *clear_accumulator* signal pulses high on the first cycle of filtering for each new output sample. This ensures that the accumulator adds its first result to zero, discarding the data from the previous output sample. The *latch_results* signal pulses high on the second cycle of each new output sample. This one cycle delay accounts for the cycle required for the third slice to add the outputs of the first two slices (see Figure 4-6). The pulses still occur every 16 cycles to latch the final 16-bit sum for each output sample.

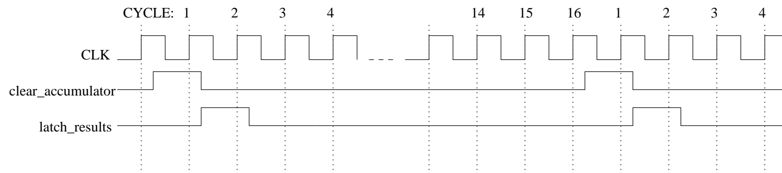


Figure 4-9: Timing Diagram for the Capturing Filter Output

Clock Gating

The input generation block and the output capture block use the core clock and operate at relatively high frequency. Rather than designing the system to drive the final output samples through the pads at the higher frequency, we implemented a method of stalling the core filtering operation. The JTAG circuitry previously mentioned can easily shift out the filter result once the core has stalled. Figure 4-10 shows this stalling system. The threshold register holds a value that is set using the JTAG interface. Once the counter reaches this threshold, the *match* signal goes high. This causes the internal clock (core clock) to gate on the following cycle. Clearly this one cycle delay means that the threshold should be set to one below the desired number of running cycles.

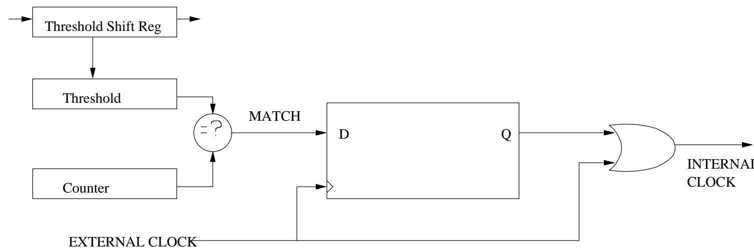


Figure 4-10: Clock Gating System

A separate reset signal applied to the counter allows the process to restart for a new set of cycles. This design permits easy verification of testchip functionality according to the following steps:

1. Set the threshold to 15 using the JTAG interface.

2. Start the core clock and remove the reset signal.

3. Once the clock gates, shift out the filter result using JTAG to get the value of the first output sample.

4. Reset the counter using its own reset.

5. Repeat steps 3-5 to view subsequent output samples.

The flip-flop between the match signal and the internal clock ensures that the clock gating always occurs synchronously with the internal clock. In other words, the internal clock stalls high only after a rising edge, and it exits its stalled state cleanly on a falling edge. Figure 4-10 presents a simplified version of the actual implementation. The final circuit includes an enable bit used to disable the counter for continous operation. It also has an external gate signal for stalling the internal clock to the core.

Chapter 5

Testchip Implementation

5.1 Introduction

This chapter provides a more detailed look at the testchip implementation. The design and implementation considerations relating to reducing standby power consumption receive particular attention. Some of the topics in this chapter address design decisions made specifically to attack subthreshold leakage in standby mode. Other topics cover practical issues that arose during implementation.

5.2 Multiple Power Regions

The testchip has four distinct power supplies. Table 5.1 lists each supply and the components of the chip that each power region contains. The four power supplies share the same ground that is common to the entire chip. In terms of gate load, the peripheral supply serves the largest number of gates. This load consists primarily of the JTAG shift registers. On the other hand, the core logic power supply serves essentially the same amount of area as the peripheral supply. The other two supplies provide power to much smaller regions of the chip.

The multiple power supplies serve a number of purposes. First and foremost, they permit an easy partitioning of chip power for measurement purposes. The peripheral circuits, primarily the JTAG shift registers, add power overhead to the chip during

Table 5.1: Power Supply Regions

Power Supply Name	Components in Power Region
Flip-Flop Power	48 registers (4 in each CLB)
Core Logic Power	LUTs Adders Muxes, Buffers, and Inverters in CLBs
Sleep Power	Sleep Signal Buffers, Muxes, and Inverters
Peripheral Power	Configuration Registers JTAG Shift Registers Input Generation Block Output Capture Block Reset Generation Block Clock Gating Block Other Miscellaneous Peripheral Circuits

both active and standby modes that would not exist in a finalized FPGA design. A completed FPGA architecture would most likely replace the shift registers with some sort of low power SRAM distributed across the chip. Placing the peripheral circuits on a separate power supply permits this overhead power to be excluded from more interesting power measurements. On the other hand, the power consumed from this supply certainly is measurable if desired.

The other three power supplies (core logic, sleep, and flip-flop) represent the more interesting set for observation. The active power of the core during filtering or other arbitrary function consists of the power drawn from these three blocks. The power drawn from the sleep supply during active operation should consist only of leakage power through high V_T devices. The entire buffering system for the sleep signal consists of high V_T devices. Most of these devices are sized for minimum leakage (as discussed in Chapter 2). This means that the sleep power region should draw a relatively small amount of power. Futhermore, the sleep buffers clearly need to remain powered during standby mode to drive the core actively into sleep. The distinct sleep supply also facilitates measuring the power dissipated in the sleep network while entering and exiting sleep mode.

The core logic power region consumes the most power during active mode because

it contains many devices that are switching. Furthermore, many of the devices in this region are low V_T devices. For this reason, the subthreshold leakage current is relatively high for this region. The core logic region offers the most easily measurable look at the reduction in subthreshold leakage current achieved during sleep mode. Additionally, the region does not contain any sequential logic. As discussed previously in the voltage scaling chapter, all of this combinational logic can be grounded at both rails during sleep since it has no state to save.

The Leakage Feedback Flip-Flops (LFBFFs) in the CLBs need the option of saving state even while the core is sleeping. The separate power supply for the flip-flops allows them to maintain state even if the core logic supply is dropped to ground. Furthermore, the separate supply for the flip-flops provides more precise measurements of the actual power a LFBFF draws in silicon in this process. The power savings offered during sleep mode by the MTCMOS implementation are also readily visible with the separate power supply.

In summary, the separate power supplies are a tool for facilitating measurement of power consumption for different sections of the chip both in and out of sleep mode. Additionally, they enable experimentation with voltage scaling as discussed in Chapter 3.

5.3 Sleep Mode Specifics

5.3.1 Multiple Regions for Sleep in a CLB

The CLB in the new architecture examined in this testchip has three primary components. They are the 16-bit LUTs, the 4-bit adder, and the 4-bit register. Some of the configurations available for this CLB do not use all of these components. For this reason, the CLB is divided into four sleep regions. All four regions respond in unison to the global sleep signal, but three of them also respond separately to local sleep signals. Figure 5-1 shows the four sleep regions within a CLB. Basically, each of the three primary blocks receives its own sleep signal. All of the multiplexors and

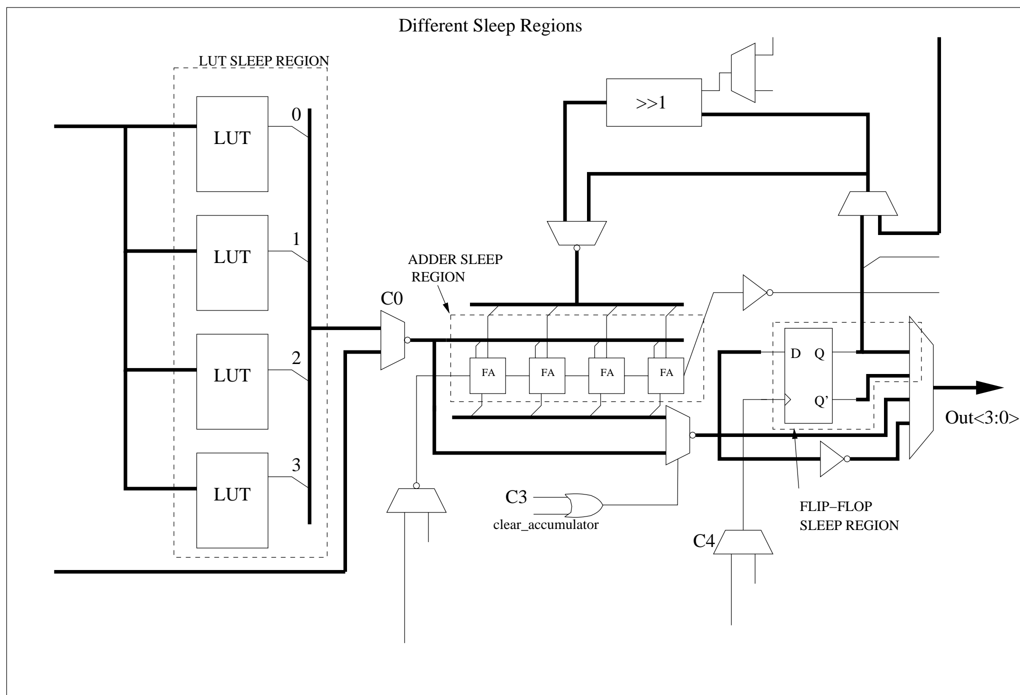


Figure 5-1: Sleep Regions within the CLB

other gates in the rest of the CLB operate only on the global sleep signal. The sleep signals to the primary blocks are selected based on the configuration bits.

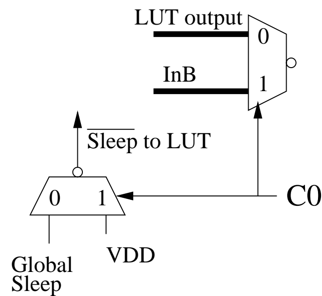


Figure 5-2: Sleep Signal Selection for LUT

Figure 5-1 shows that configuration bit **C0** selects between the output of the LUT and a second input to the CLB. Clearly, if **C0** selects the other input, then the LUTs cannot be used inside of the CLB for this configuration. Thus, **C0** also selects between two sleep signals for the LUTs as shown in Figure 5-2. When the LUT block is in use,

C0 directs the global sleep signal to the LUT. If the second input to the CLB is used instead, then the LUT block automatically enters sleep mode. Notice that the signal actually routed to the LUT block is $\overline{sleep}$. The inverted sleep signal is appropriate because the LUT has only NMOS high V_T sleep devices. In a similar fashion, **C3** selects the sleep signal for the 4-bit adder, and **C4** selects the sleep signal for the 4-bit register. This approach ensures that any major component that cannot be used in a given configuration consumes the lowest possible power due to subthreshold leakage.

One concern about this approach is that the block in cutoff might have floating outputs that drive CMOS gates. If this occurred, the CMOS gate could theoretically open a short circuit path from power to ground. Careful design of the CLB prevents such a situation from occurring. Take the LUT block as an example. The output of the 16-bit LUTs drives a multiplexor. The muxes in the CLB use transmission gates to select which input signal will drive the output inverter. This design means that the LUTs in sleep mode drive one side of a transmission gate in the "off" state. Therefore, any floating nodes inside the sleep region remain isolated from the gates of other devices. The transmission gate is a low V_T gate, so there can be a large subthreshold leakage current through the gate. However, this current is no larger than the current that would be there if the LUT was not in sleep mode.

The sum output of the adder interface with a transmission gate mux just like the LUT output, so no short circuit currents can arise there. The carry out signal, however, directly drives the input to an MTCMOS gate that will remain active when the adder is in sleep. This could present a problem if the carry out signal floated to an intermediate value, producing short circuit current through the inverter. No explicit measures are taken in the design to prevent this because the carry out node transitions to V_{DD} reasonably quickly upon entering sleep mode. This transition occurs because the leakage currents from the power rail are at least an order of magnitude higher than the current through the off sleep device. This mismatch of current quickly charges the node. In a future design, adding an explicit means of preventing the floating node would be advisable. One solution would be to replace the carry out inverter inside the adder with a leakage feedback gate. As previously discussed, this gate would

continue to drive its output actively in sleep mode.

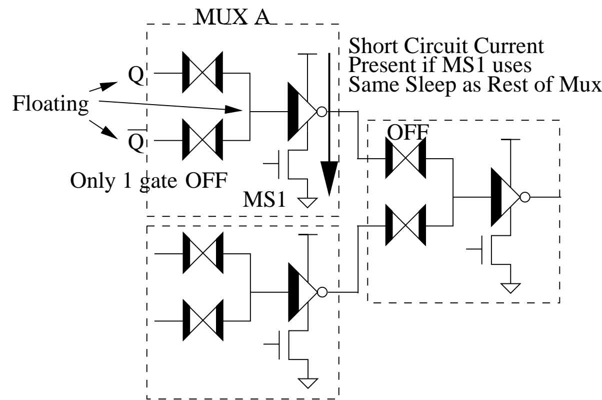


Figure 5-3: Short Circuit Current due to Poor Sleep Region Interface

The outputs of the flip-flops drive transmission gates, but the situation is different than for the LUT and adder. In the previous cases, the transmission gate driven by the floating inputs always was off. Figure 5-3 shows that one of the flip-flop outputs, either Q or $\overline{Q}$, will always drive the input to an inverter. Furthermore, the Q node does not necessarily charge up to V_{DD} like the carry out node does. Instead, this node tends to float at an intermediate value. This creates large short circuit currents in the output inverter of *mux A* in the figure. Q or $\overline{Q}$ might float because the combination of having a 0 stored in the flip-flop and having a large sleep device means that the leakage current from the power rail cannot dominate the leakage through the sleep device enough to charge up those nodes.

Including the first mux from the 16-bit mux in the flip-flop sleep region eliminates the problem of short circuit currents. Now *mux A* is in sleep mode, and its output drives a transmission gate that is off.

The proper interfacing of the sleep regions depends on correct values for the configuration bits. Table 5.2 shows how other configuration bits must be set in a certain way when one of the sleep regions is in sleep mode. The non-X bits in the table prevent large short circuit currents from arising by maintaining the sleep region interfaces described above.

Table 5.2: Configuration Bit Constraints

Sleep Region in Sleep Mode	Determining Bit	Dependent Bits (MSB-LSB)
LUT region	C0	XXXXXXXX1
Adder region	C3	XXXXX1XXX
Flip-Flop region	C4	X1X11XXXX

The properly interfaced sleep regions provide automatic power savings for any CLB that does not use one or more of the major elements in the CLB. Table 5.3 shows the reductions in steady-state current drawn by the CLB when each sleep region is sleeping. These numbers come from a simulation of a single CLB with the clock gated to isolate steady-state power. Once the steady-state current for the CLB settles, the configuration bit that controls each respective sleep region changes. This sequence represents a reprogramming of the CLB. This simulation assumes that all other configuration bits remain constant and adhere to any applicable constraints from Table 5.2. When the LUT region or the adder region enters sleep, the flip-flop power supply current does not change. Similarly, the core logic power supply current does not change when the flip-flop region enters sleep. The second column in the table shows the factor reduction in current for the power supply that should change for

Table 5.3: Steady-State Current Reduction Using Sleep Regions

Sleep Region in Sleep Mode	Core/FF Supply Current Reduction	Total Reduction
LUT region	2.7X	2.3X
Adder region	3.5X	2.8X
Flip-Flop region	5.8X	1.2X

the given sleep region. The third column shows the factor reduction in steady-state current for the entire CLB including the core logic power supply current, the flip-flop power supply current, and the sleep power supply current. The flip-flop sleep region has the least impact on the total power because the flip-flops draw a relatively small amount of current compared to the rest of the logic in the CLB.

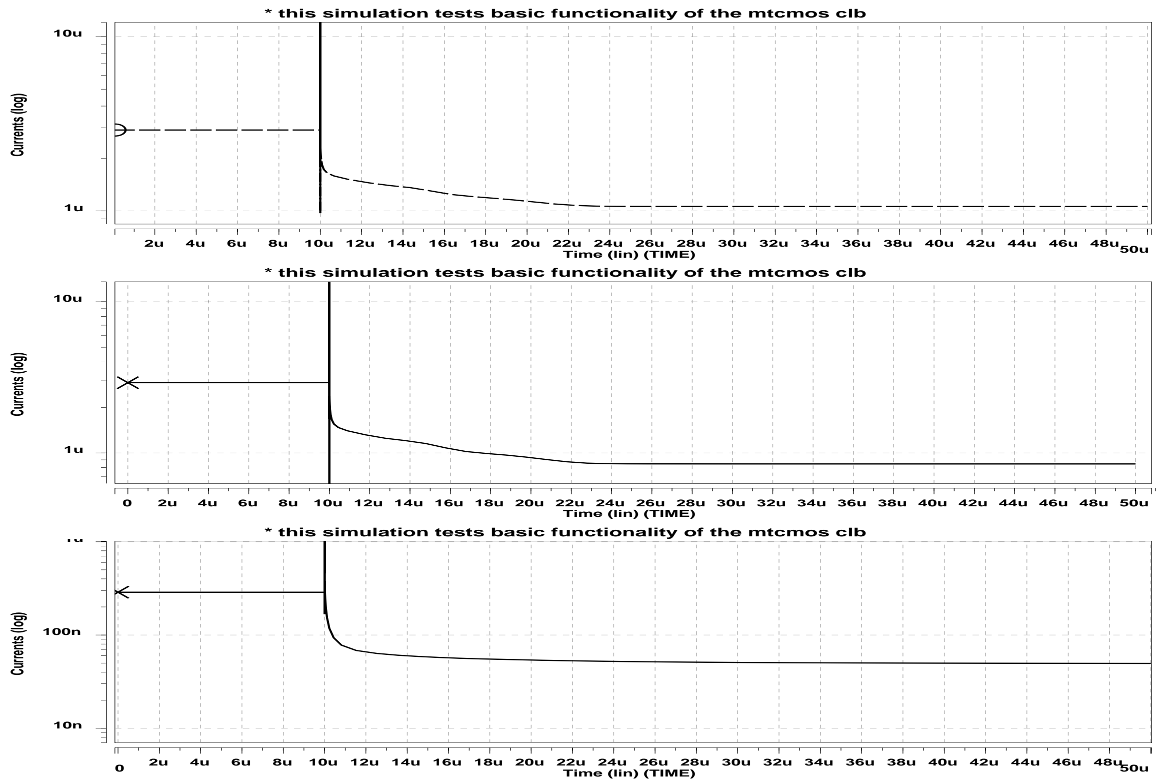


Figure 5-4: Steady-State Current Change Entering Sleep

Figure 5-4 provides plots of the steady-state current settling to the new value. The top plot shows the core power supply current settling when the LUT region enters sleep. The center plot shows the same power supply current settle when the adder enters sleep. The final plot shows the flip-flop power supply current settle when the flip-flop region enters sleep. All power supplies not shown on a given plot maintain their steady-state value. The current spike at the moment the configuration bit changes indicates that there is some power overhead entering these modes, but reprogramming an FPGA occurs rarely. Furthermore, the current settles quickly to its steady-state value. The numbers in the center column of Table 5.3 come from these plots.

These results show that the multiple sleep regions can reduce the amount of steady-state current drawn by a CLB by several times for some configurations. This element of the design reduces the power consumed by the FPGA in active mode.

5.3.2 Fine Granularity

This testchip design uses MTCMOS circuits to reduce subthreshold leakage in the sleep mode. Traditionally, high V_T sleep devices are shared across many gates. Sometimes, entire sections of a chip are gated using a single sleep device. Sharing sleep devices provides several benefits. First, the shared approach funnels the leakage currents of every shared device through a single high V_T FET. This funneling immediately reduces the combined leakage of all of the low V_T gates to that of the sleep device. Secondly, the shared approach reduces the amount of area dedicated to sleep devices. The widths of sleep devices for two gates that can switch simultaneously must be added when those sleep devices are combined. Combining the sleep devices of gates that never switch simultaneously, however, does not require that the widths be added. In theory, the larger of the two widths serves to gate both devices without increasing delay.

The practice of combining sleep devices also has a few significant disadvantages. First, sizing a shared sleep device becomes quite difficult when the number of gates sharing that device is large. Determining the amount of discharge current the sleep device must support is difficult except in rare instances. Additionally, sharing one large sleep device among many gates does not facilitate synthesis based layout.

The testchip design explores the use of sleep devices at a fine granularity. Essentially, each gate has its own sleep devices. Gates only share sleep devices when they can be logically combined into larger cells (*i.e.* - two inverters make a buffer). This approach made the sizing of sleep devices easier. It also allowed a semi-custom standard cell approach to layout that reduced the amount of time for implementation.

5.3.3 Sizing Sleep Devices

The problem of sizing sleep devices optimally has received a significant amount of attention. This design effort chose a relatively simple approach. The CLB was first implemented and simulated using all low V_T devices. This low V_T version of the CLB was designed to have a relatively fast critical path through the adder.

The MTCMOS version of the CLB was compared to the low V_T version for sizing. All of the low V_T devices in the MTCMOS design kept the same sizes as their counterparts in the low V_T design. As the above section describes, the sleep devices in the design are distributed at a fine granularity. This enabled an easy comparison between the delays of a given gate (mux, buffer, FF, etc) in the low V_T and the MTCMOS designs. The addition of sleep devices to the low V_T design will inevitably increase delay. The devices in the testchip were sized to ensure that the delay of each gate or component in the MTCMOS design was no more than 10% slower than the low V_T design. This number sets a standard of performance. The testchip will show what subthreshold leakage reduction is possible without increasing delay by more than 10%. The estimated area overhead of the sleep devices is less than 5%.

5.3.4 Sleep, FFs, and Clock

One particular section of the CLB deserves special attention regarding the sleep mode configuration. The flip-flops in the CLB maintain their state in sleep by design. The flip-flop remains unaffected by any changes at the data input to the flip-flop. However, the design does require that the *clock* and $\overline{clock}$ signals remain correctly asserted during sleep to prevent short circuit currents inside the flip-flop. The clock distribution elements of the CLB (including muxes and buffers) are MTCMOS gates. Sleep mode causes these clock drivers to disconnect from the rails. This means that either or both of the *clock* and $\overline{clock}$ signals can float during sleep mode.

We corrected this problem by using a high V_T keeper device for the $\overline{clock}$ signal. Figure 5-5 shows the implementation of the clock distribution to the flip-flops. The local *clock* inverter already holds the *clock* signal high prior to entering sleep because

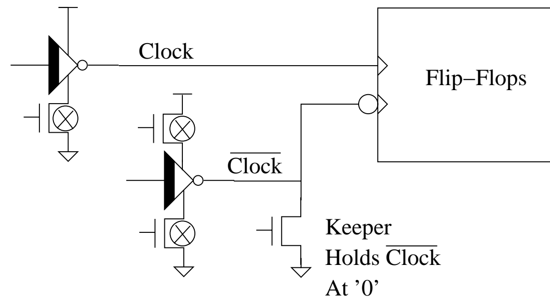


Figure 5-5: Clock Buffers to Flip-Flops During Sleep Mode

the clock is gated. Upon entering sleep, the *clock* signal remains high without a keeper device. The $\overline{clock}$ signal, on the other hand, requires sleep devices of both polarities to isolate that node during sleep. A high V_T keeper device turns on in sleep mode and holds the $\overline{clock}$ node low. This setup ensures that the clock signals remain at the correct values during sleep mode and prevents short circuit currents inside the flip-flops.

5.3.5 Energy Overhead for Sleep Mode

The MTCMOS techniques applied to the CLB promise to reduce the standby power consumption of the FPGA architecture dramatically. This section examines the energy overhead associated with entering and exiting sleep mode. The control circuitry should not place the CLBs into sleep if the overhead energy expended for that purpose exceeds the amount of energy saved while the CLB remains in sleep.

The energy overhead associated with transitioning to and from sleep mode comes from two places. First, the network of drivers that switches the sleep signals consumes active power. Whenever the global sleep signal changes, the drivers switch the *sleep* and $\overline{sleep}$ nodes inside the CLB. The capacitance on these nodes (particularly the $\overline{sleep}$ node) is relatively large since there are many sleep devices in the CLB. The transition out of sleep mode actually consumes more energy in the sleep signal network because the large capacitance at the $\overline{sleep}$ node must be charged to V_{DD}. The floating gates inside the CLB comprise the second significant source of overhead

energy consumption. When the sleep devices disconnect the MTCMOS gates from ground, large leakage currents through the PMOS devices charge most of the floating nodes in the CLB to V_{DD}. The overhead energy contributed by these sources must be considered before the CLB enters sleep mode.

Figure 5-6 shows the total current drawn and the energy consumed by the CLB while entering and exiting sleep mode. The top plot shows that the total current

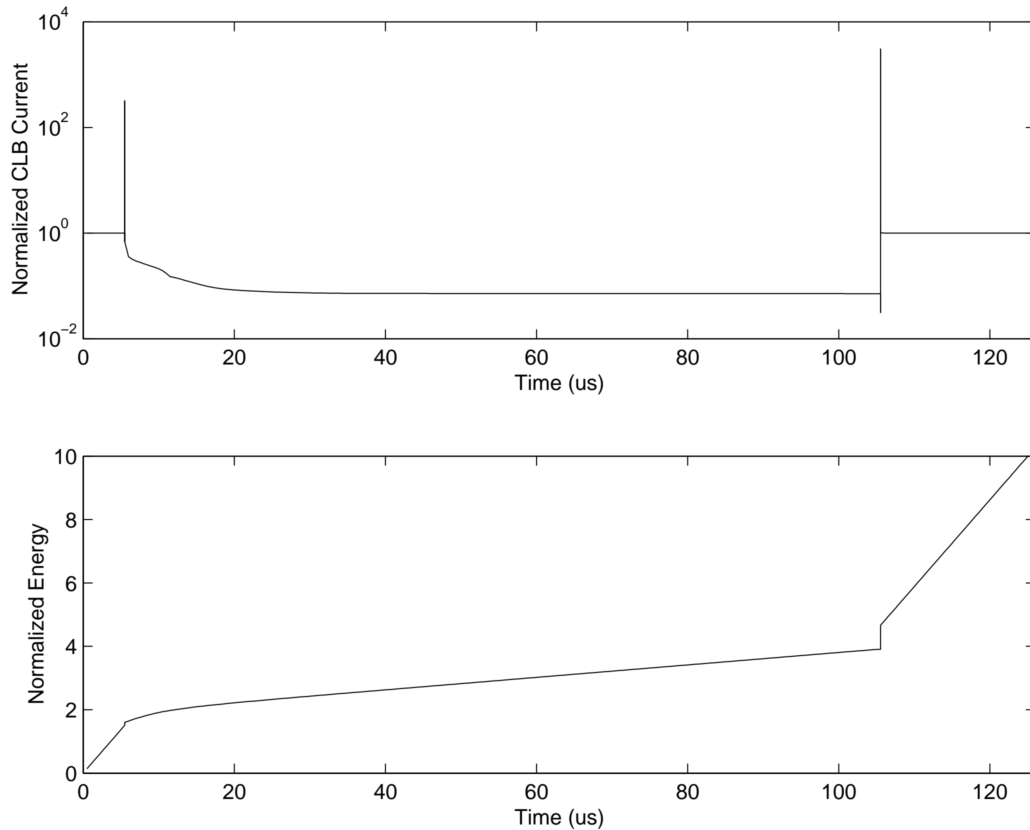


Figure 5-6: CLB Current and Energy Transitioning to/from Sleep Mode

drops by over an order of magnitude when the CLB enters sleep mode. The initial spike of current at the assertion of the sleep signal corresponds to the switching of the sleep signal network. The current drawn by the CLB takes 10s of μ-seconds to settle to the steady state value during sleep mode. The larger current spike occurs when the CLB exits sleep mode. Again, the increased current draw during the transition out of sleep primarily is due to charging up the large capacitance at the $\overline{sleep}$ node.

The bottom plot in Figure 5-6 shows the energy consumed by the CLB during the transitions to sleep. The reduced slope of the curve during sleep mode corresponds to the lower rate of power consumption. The current spikes at the transitions to and from sleep mode appear as small step-ups in the energy graph.

In order for sleep mode to be worthwhile, the energy saved must exceed the overhead energy that appears as steps in the energy plot. Figure 5-7 shows the energy consumed by the CLB at the break-even point. In other words, the energy consumed by the CLB entering and then exiting sleep mode approximately equals the energy

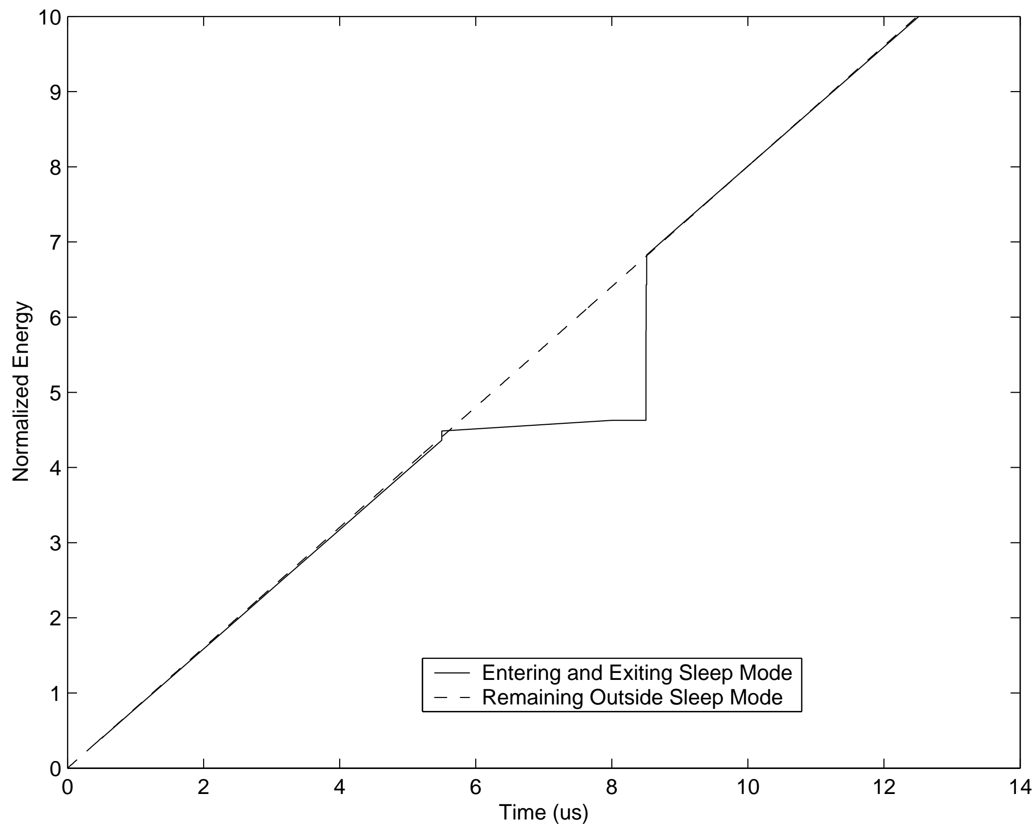


Figure 5-7: Break-Even Time for Entering Sleep Mode

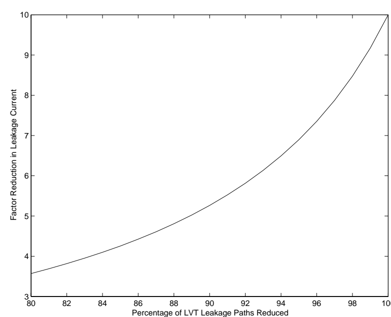
the CLB consumes during the same time spent outside of sleep mode. The dashed line in the plot shows the energy consumed by a CLB that remains outside of sleep. The solid line shows the energy consumed by a CLB entering sleep mode. This energy initially jumps above the other case due to the switching energy consumed while

entering sleep. The lowering currents in sleep mode quickly cause the total energy to fall below the non-sleep case. However, the CLB must remain in sleep long enough to account for the energy consumed while exiting sleep. The plot shows that the break-even time is less than 5μs. Notice that the break-even time occurs well before the current even settles to its steady-state value in sleep mode. Whenever the CLBs can remain in sleep mode for longer than $\sim 5\mu$s, sleep mode will reduce the energy consumed by the CLBs.

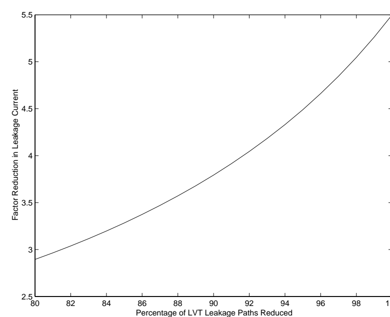
5.4 Stopping Sneak Leakage Paths

An MTCMOS design intends to block every path between power and ground with a high V_T FET that is turned off. The term *sneak leakage path* refers to paths between power and ground that remain during sleep mode without being cut off in this way. Even a relatively few sneak leakage paths can rapidly degrade the sleep mode power savings that MTCMOS could potentially provide.

A simple example can demonstrate how sneak leakage paths can degrade savings. Suppose there are 100 distinct current paths from power to ground in an MTCMOS circuit. Before entering sleep mode, each path has a normalized subthreshold leakage current of 10. If a high V_T FET correctly gates a path in sleep mode, the leakage current reduces to 1. This single order of magnitude reduction roughly coincides with the process used for the testchip. Figure 5-8(a) shows how the current reduction factor (subthreshold leakage current before sleep / subthreshold leakage current after sleep) varies with the percentage of current paths that are gated. This figure shows that a relatively small percentage of sneak leakage paths (~10%) can reduce the savings by one half. Clearly, the potential savings are reduced if there are other overhead current paths in the circuit. Figure 5-8(b) adjusts the previous plot in the case where there are 100 "overhead" high V_T (current=1) current paths during both active and sleep mode.



(a) Current Reduction Factor vs % Low V_T Paths Eliminated



(b) Same as 5-8(a) with Overhead

Figure 5-8: Effect of Sneak Leakage Paths on Current Reduction

5.4.1 Causes of Sneak Leakage Paths

Previous work regarding sneak leakage paths suggests that there are three primary sources for these paths [19]:

- The output of a CMOS gate connects directly to the output of an MTCMOS gate.
- The output of a CMOS gate connects to the output of an MTCMOS gate through low V_T transmission gates.
- Reverse conduction paths occur using the virtual ground node of a shared high V_T sleep device.

Sneak leakage paths can arise from the occurance of one or more of these three situations. This work highlights a few additional sources of sneak leakage paths.

- The output of a CMOS gate connects to a second CMOS output through low V_T transmission gates.
- The output of a CMOS gate or MTCMOS gate connects to a power rail through low V_T transmission gates.

The next section describes some situations in the testchip where sneak leakage paths could have occurred. It also details the measures taken to eliminate these paths.

5.4.2 Sneak Leakage Paths Eliminated from the Testchip

High V_T Latches in LUT

This first sneak leakage path exists because two high V_T latches in the LUT have outputs that are connected through low V_T passgates. The high V_T latch shown in Figure 5-9 holds one bit worth of the LUT contents. The LUT architecture selects

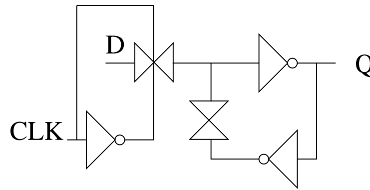


Figure 5-9: High V_T Latch Used in LUTs

the contents of one of these latches as the LUT output using a funnel of transmission gates. This implementation is efficient in terms of speed and power for a 16-bit LUT. The transmission gates themselves consist of low V_T devices since the LUT is on the critical path in many FPGA configurations. The signals that select the output by turning on the correct passgates are themselves driven by low V_T devices. Figure 5-10 shows how sneak leakage paths can arise in this architecture. The outputs of any two

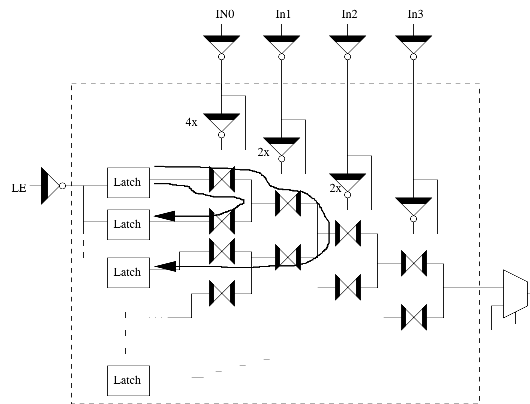


Figure 5-10: Sneak Leakage Paths in 16-bit LUT

high V_T latches in the design are separated only by one or more low V_T passgates.

If any two latches store opposite values, then this situation effectively places one or more low V_T passgates directly between power and ground. Clearly, multiple sneak leakage paths can arise from this design.

The situation can actually get much worse in sleep mode. The low V_T drivers that turn on and off the passgates are themselves MTCMOS gates. In sleep mode, these inverters are placed into a low leakage state using an NMOS high V_T device. As a result, the inverter's output eventually rises to V_{DD}. This might occur simply because it is driven high (the input to the inverter is zero). More subtly, the large leakage current through the low V_T inverter charges up the output node. This occurs because the relatively small leakage current through the sleep device cannot balance the larger current from the power rail. Once both the *select* and $\overline{select}$ signals for each transmission gate are both high, the NFET in the transmission gate turns on. This essentially creates a short circuit current between any of the latches with opposite values.

Figure 5-11 shows the simple solution to this problem. Buffering the output of the high V_T latches immediately with an MTCMOS inverter isolates the latches from one another in sleep. Now, the high V_T sleep device that the inverters all share cuts off all paths from power to ground in the transmission gate network. This fairly simple solution eliminates the fairly simple sneak leakage path.

Sneak Leakage Paths and Multiplexors

A similar sneak leakage path can appear anywhere that low V_T multiplexors select between two actively driven signals. This occurs in several places inside the CLB. For example, consider the bit that shifts in for the shifted feedback path. A multiplexor selects between either the input port or a zero. Figure 5-12 shows this arrangement with a hardwired zero. The figure includes the MTCMOS driver that drives the shift-in signal from an adjacent CLB. Both the shift-in driver and the output inverter of the mux are gated from the ground rail in sleep mode. Nevertheless, a sneak leakage path exists through the driving inverter, through both transmission gates in the mux, and straight to ground. As mentioned in the previous section, the shift-in signal

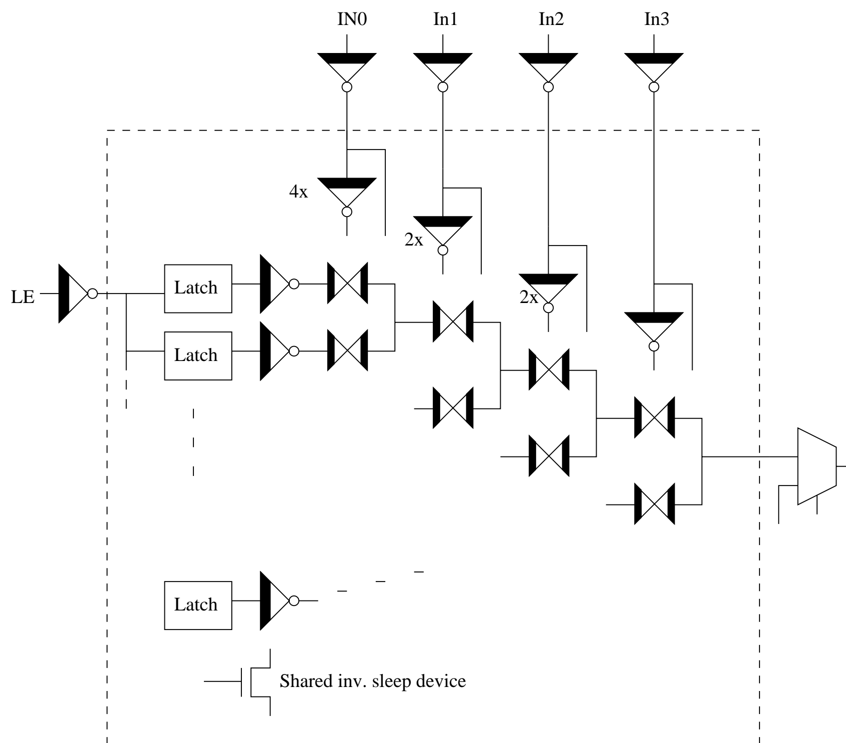


Figure 5-11: Final Design for 16-bit LUT

will eventually charge up to V_{DD} because of leakage current mismatch, even if it was originally grounded. Thus, a sneak leakage current exists through the mux that is an order of magnitude higher than necessary.

The direct connection of one mux input to a power rail can always cause this problem. One solution to the problem would be to replace the passgate that connects to the rail with a high V_T passgate. This solution will not work if an MTCMOS gate drives the select signal to the mux. In that case, the select signal (or $\overline{select}$) could float high during sleep. This would turn on the high V_T transmission gate and reintroduce the sneak leakage path. In the case of the CLB, this solution might work well because a high V_T configuration latch drives the *select* signal of the mux. However, the inverter that generates $\overline{select}$ also would have to become high V_T to prevent its floating during sleep. Also, this approach only works if the select signals choose the low V_T transmission gate's input during sleep. Since that side remains low V_T for speed (it is on the critical path), the sneak leakage path reappears if the new

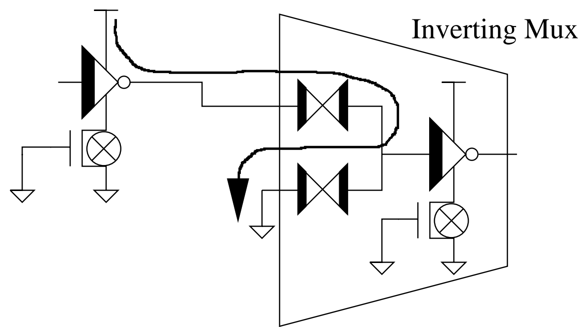


Figure 5-12: Sneak Leakage Path through Multiplexor

high V_T transmission gate is "on" during sleep. The solution chosen to eliminate this sneak leakage path in the testchip does not have these complications.

A second solution that eliminates this sneak path simply replaces the direct ground connection with an MTCMOS inverter. As Figure 5-13 shows, this FET eliminates the sneak leakage path at the cost of some small area and power overhead. The

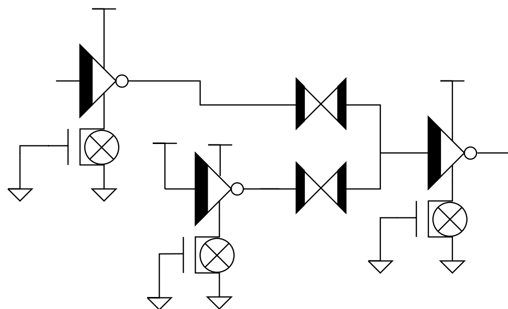


Figure 5-13: Sneak Leakage Path Eliminated through Multiplexor

device has essentially no dynamic power because it never switches. Also, its sleep mode leakage power is at worst that of a minimum sized high V_T NFET.

Subtle Sneak Leakage Paths

The examples thus far have both involved sneak paths through low V_T passgates. The leakage currents can take much more complicated routes through a circuit as well. The next example shows how seemingly positive circuit decisions can create

extremely subtle sneak leakage paths. In fact, the separate design elements in this example from the CLB are each correctly designed to eliminate sneak leakage paths. The combination of these elements, however, produces a path from power to ground that, in some cases, is not severed by a high V_T device.

Figure 5-14 shows the sneak leakage path that arises when a Leakage Feedback Flip-Flop (LFBFF) is driven by another MTCMOS inverter. The high V_T inverter

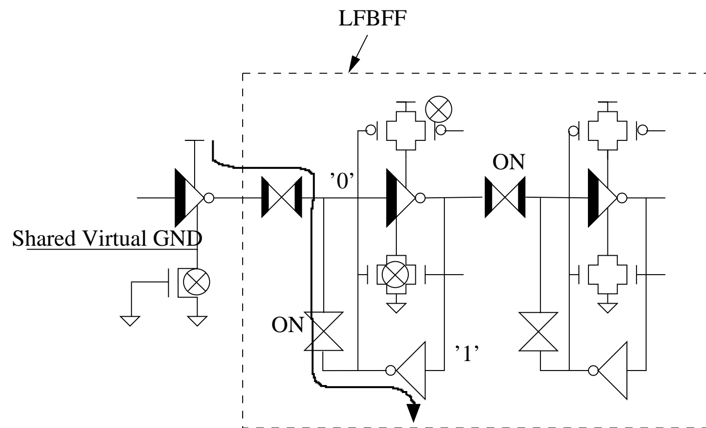


Figure 5-14: Sneak Leakage Path Due to Missing Sleep Device

that feeds back inside the first stage of the LFBFF can provide a short circuit to ground when that latch stores a logical '1'. During sleep mode, this latch is in a hold state, so its high V_T passgate is open. This leaves only a low V_T passgate to isolate the short circuit to ground from the outside world. The previous sections have shown that any isolation of a power or ground rail using a low V_T passgate deserves special attention. In this case, the use of a single sleep device does not suffice to prevent sneak leakage paths. Placing a PMOS high V_T sleep device on the driving inverter eliminates the sneak leakage path from the figure.

Another interesting thing to notice is that the virtual ground in the figure is shared with other devices. The designer of the LFBFF indicates that sharing the virtual rails of the input inverter with other MTCMOS gates outside of the flip-flop is safe [19](p127). In fact, the next example shows that this is not always the case.

Figure 5-15 shows the sneak leakage path that courses through several elements

from the top level CLB hierarchy. The path basically begins at the drivers of the flip-

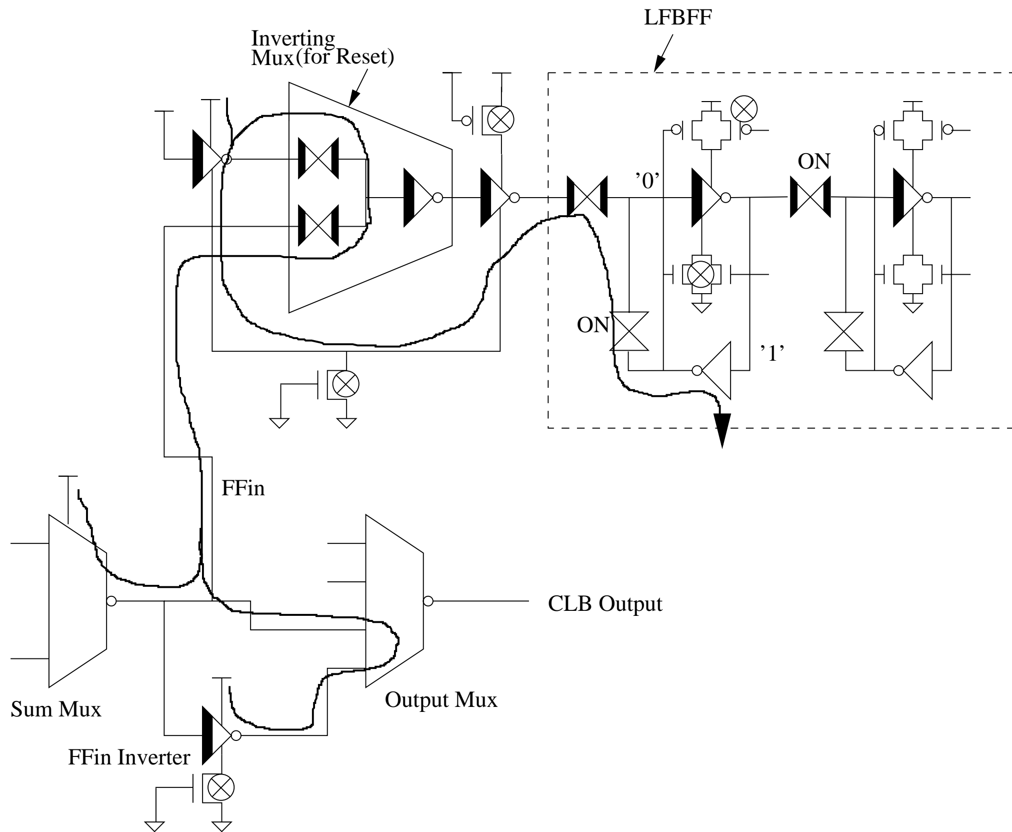


Figure 5-15: Subtle Sneak Leakage Path Due to Shared Sleep FETs

flop input signal and terminates inside the flip-flop itself. The figure shows one bitslice of the 4-bit path within each CLB. The “Sum Mux” selects between the output of the adder and the adder’s “A” input. This signal, *FFin* in the figure, branches to several locations. First, it connects directly to the 4:1 output multiplexor, “Output Mux” in the figure. It also drives an inverter that then connects to an adjacent input on the Output Mux. *FFin* finally drives the flip-flop block itself. The flip-flops in the CLB have a synchronous reset that is implemented using a mux. Some of the internal elements of this mux are shown in the figure. The select signal inverter and the sleep devices for the mux are excluded from the figure. The second input to this mux is properly buffered from the rail using an MTCMOS inverter as the previous section discussed. The output of the mux is buffered with one last inverter prior to

driving the input of the flip-flop.

The figure shows that the inverters at the input of the mux and at its output share the same sleep device. The figure also makes it apparent that the sneak path uses this virtual ground to bypass the mux entirely! This connection certainly is not obvious during design. In fact, sharing the sleep device makes a lot of sense. The inverter at the mux input does not ever switch, so its sleep device will hardly be loaded in active mode. Sharing this sleep device with the output inverter serves to reduce the area without slowing the output inverter during active mode. It also reduces subthreshold leakage in sleep mode by funneling the leakage of two low V_T inverters through a single sleep FET. If the output inverter drove an MTCMOS logic gate, then the shared sleep device would be a good decision. The problem arises because of the feedback inside the LFBFF.

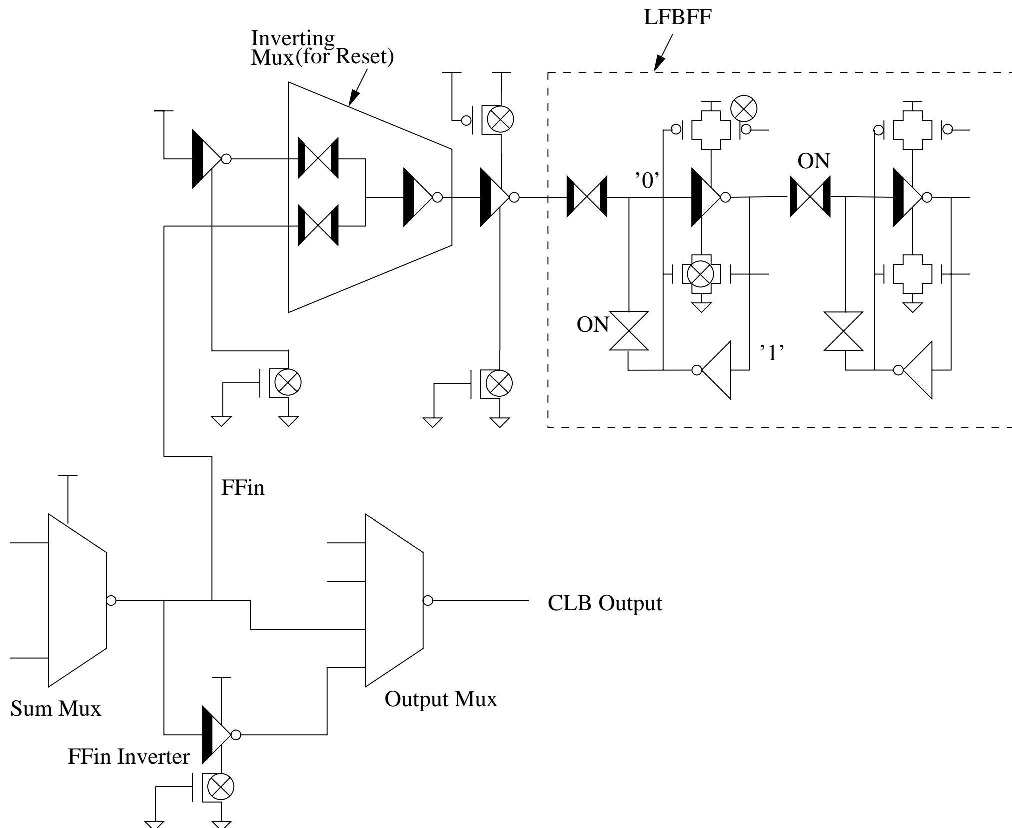


Figure 5-16: Isolating LFBFF Input Removes Sneak Path

The sneak path generated by the missing PMOS sleep device served to show that the LFBFF needs its input to be isolated carefully from other circuits. In fact, even sharing a virtual ground of the input inverter can produce sneak leakage paths. Sharing a virtual ground node is not safe unless all of the gates that share that node are also cut off explicitly from power using PMOS sleep FETs. This includes gates that might access either virtual node through low V_T passgates.

The simplest solution to this problem is the one used on the testchip. Figure 5-16 shows that the use of separate sleep devices eliminates the subtle sneak leakage path. In fact, as long as the inverter driving the LFBFF has its own sleep FETs, this type of sneak path disappears. The MTCMOS gates in the combinational logic section of the circuit are then free to share sleep devices.

5.4.3 Guidelines for Preventing Sneak Leakage Paths

Sneak leakage paths occur when a current flows from power to ground without being gated by a high V_T FET. Using both polarities of sleep device for each MTCMOS gate without sharing any sleep devices eliminates all sneak leakage paths. This conservative approach adds unnecessary area overhead. An awareness of the sources of sneak leakage paths can allow designers to prevent them with the minimum necessary area overhead. Note that for this discussion the idea of "shared outputs" includes outputs that are connected through low V_T transmission gates.

The following list provides guidelines for designing MTCMOS circuits that prevent sneak leakage paths. While these guidelines do not cover every case, they prevent the most common occurrences of sneak leakage paths. Again, the notion of shared outputs includes outputs that connect through one or more low V_T transmission gates.

- Any MTCMOS gate that shares an output with a CMOS gate or power rail needs to use both polarity sleep devices.

- An MTCMOS gate that shares outputs with other MTCMOS gates must use the same polarity sleep device (or devices) as the other gates.

- Do not share sleep devices if the shared line creates a connection between outputs of multiple CMOS gates.
- Any MTCMOS gate that shares a sleep device with a gate that uses both polarity sleep devices must also have (or share) both polarity sleep devices.

5.5 Layout

5.5.1 General Strategy

The 0.13μm process used is new enough for synthesis from standard cells to be unavailable to us. Even if this was not the case, no public standard cell libraries are available with dual V_T or MTCMOS cells. For this reason, we adopted a semi-custom layout approach for this testchip. The CLB and surrounding logic was broken down into cells that might be found in a typical standard cell library. These cells included MTCMOS cells, low V_T cells, and high V_T cells. The cells were sized to minimize leakage without incurring significant delay penalties (over 10% relative to an all low V_T design) as specified in the section describing sleep device sizing. The completed, semi-standard cells provided the blocks for custom placement and routing to implement the testchip. This layout approach even affected earlier schematic-level design. Cells at design time were often sized to the nearest of a reasonably small set of sizes to minimize the number of standard cells needed for layout. For example, most low V_T inverters on the testchip used one of four sizes. The PMOS widths were selected to equalize rise and fall times. Restricting the sizes of different gates at design time reduced the complexity and workload during implementation of the testchip.

We also developed a general routing strategy for the testchip. All of the important signals on the chip were handrouted for efficiency. The next two sections elaborate on clock and power routing. Most of the logic signals were simply routed in a direct fashion. An autorouting tool produced by ICCraftsman automatically routed the remaining non-critical signals. For example, the autorouter connected the outputs of the JTAG shift registers to the correct holding registers in the circuit. This type of

signal included the 768 bits that routed to the LUTs in the design.

5.5.2 Clock Signals

The testchip design has two clocks for distributing: the core clock and the JTAG clock (TCK). The core clock ultimately drives 48 flip-flops (4 in each of 12 CLBs), the 8-bit LFSR, and the 113-bit circular shift register in the input generation block. The primary clock drivers, which buffer the internal clock signal generated by the clock gating circuitry, consist of two large inverters. These inverters drive additional buffers at the CLB level. Each CLB buffers the clock out of necessity, because a configuration bit can gate the clock locally at the CLB level. As discussed in the sleep device section of this chapter, the inverted clock signal required at the CLB's flip-flops is generated locally using an inverter with two sleep devices. As mentioned above, both sleep devices are necessary because of the keeper FET that holds the $\overline{clock}$ line stable during sleep mode. The driver for the non-inverted clock signal requires only one sleep device because it drives only the gates of FETs in the flip-flop.

The single large clock buffer for driving all the way to the local level eliminates the problem of varying skew until the local clock buffers. The clock signal was routed in high level metals with plenty of width.

The JTAG clock drives a considerably higher load than the core clock, although its frequency can be arbitrarily low since the configuration process occurs separately from core activity. Similar to the core clock, the JTAG clock is driven by one large buffer all the way to the local level. Most of the shift registers at the local level then buffer the clock for their own use.

Since the size of the shift registers is large (up to 768 bits), some care was taken during clock routing. The shift registers' design includes buffers at the output and muxes at the input that create a good amount of hold time in the system. However, the size of the register conceivably could cause problems if the clock is distributed unwisely. For example, consider that the individual bits of the shift register are placed in a folded pattern to use area efficiently. Suppose the clock is routed linearly to this array of shift register bits. The clock might then reach bit 32 at the bottom of one

column of registers well before reaching bit 33 at the top of the next column. If the large capacitance on the clock node increases the relative arrival times beyond the hold time, then bit 33 will latch the new and incorrect value. A scenario of this type seems unlikely in the testchip design due to the inherent hold time of the shift registers. However, in the name of precaution, the JTAG clock was routed linearly from the MSB to the LSB of each shift register. This ensures that the more significant bit always begins to latch its new value before the less significant bit even begins to change. This routing strategy couples with the inherent hold time of the registers to ensure proper operation of the shift registers.

5.5.3 Power Signals

As discussed above, there are four power regions inside the testchip. The core logic region and the peripheral region by far constitute the majority of the chip area. These two power signals and the ground signal for the entire chip were routed using an irregular grid. At the cell level, power and ground are routed in horizontal strips primarily in metal 1. Vertical metal 3 wires connect these local lines at the next level of hierarchy, and then higher level metals continue to build up the grid with increasingly wide wires. A large ring of metal for the core power, peripheral power, and ground signals surrounds the entire area of the design. The edges of the power grids connect to this ring at every convenient location. This approach ensures that current can access the local regions of the chip through the power grid from every side. The remaining two power signals were routed directly to the relatively small areas they needed to access.

5.5.4 Critical Path Implementation

The critical path for most configurations of an FPGA with the testchip architecture will include the carry path through the adders. This is because any function that performs an addition of greater than 4 bits will ripple the carry signal through multiple CLBs. The placement decisions for the CLB and slice components account for this

consideration. Figure 5-17 shows the basic layout for a slice (with 4 CLBs). The

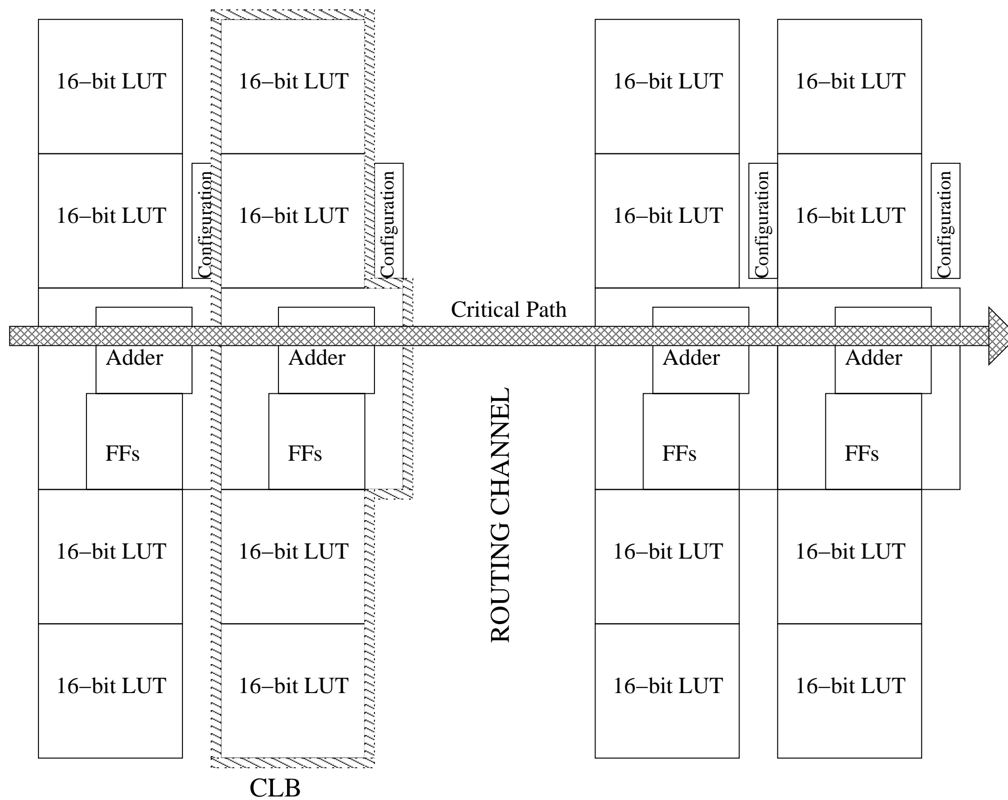


Figure 5-17: Layout Placement for a Slice (with 4 CLBs)

figure shows that the large LUTs are placed to minimize their effect on the critical path. The distance from the LUTs to the multiplexors in the core remains small even though the LUTs do not increase the distance between adders in different CLBs. The center of each CLB contains the adder, the 4-bit flip-flops, and the various muxes and other elements in the CLB architecture. The carry path between adders is as short as possible.

Notice that the slice contains a routing channel in its center. This channel has enough width to accomodate a switching block. This centralized routing channel provides substantial flexibility for higher level routing on a finalized FPGA by providing access to at least one edge of every CLB in a slice. In this testchip, the three slices stack vertically with all of the inter-slice routing down the center channel.

Adder

The adder deserves special mention as a key component on the critical path. As the architecture discussion in the previous chapter showed, all of the inputs to the adder were inverted. The inverting nature of a transmission gate multiplexor (two transmission gates feeding the input of an inverter) made this decision natural. Another naturally inverting mux produces the final correct sum, and an explicit inverter buffers the carry out signal to produce the correct carry. A mirror adder is used for the full adder block. The signals $\overline{C_O}$ and $\overline{S}$ are inverted in each full adder before continuing to the next stage. The inverter in the carry out path reduced the capacitance at the $\overline{C_O}$ node, thereby reducing delay. The sum logic is minimum sized in the first three bits and somewhat larger in the MSB of each 4-bit adder. This sizing accounts for the position of the MSB sum on the critical path in the final bit of an addition.

5.5.5 Layout Screenshots

This section provides some screenshots of the testchip layout at various levels of the hierarchy. The screenshots only include enough layers to give the general idea of the

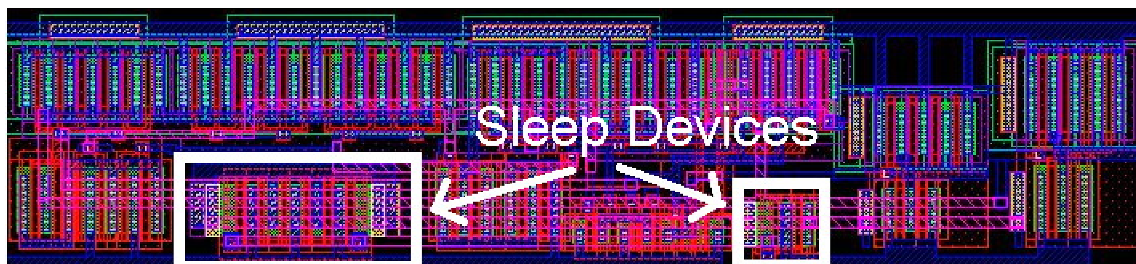


Figure 5-18: Layout of Full Adder

relative sizes of regions in the layout. For example, they omit the higher level metals. Figure 5-18 shows the layout for the full adder. The highlighted regions are the NMOS sleep devices. The smaller sleep device gates the inverters that produce the $\overline{C_O}$ and $\overline{S}$ signals. These inverters attach to internal nodes in the mirror adder design. The larger sleep device connects to the rest of the carry and sum logic. The picture of

the layout shows that the sleep devices comprise a relatively small percentage of the adder area. This area overhead is similar in other components on the testchip.

Figure 5-19 shows the layout of the CLB. The lookup tables (LUTs) comprise the majority of the area in the CLB. The addition of the 4-bit adder contributes a relatively small amount of area overhead. The non-highlighted sections of the CLB include all of the buffers and multiplexors that tie together the major components.

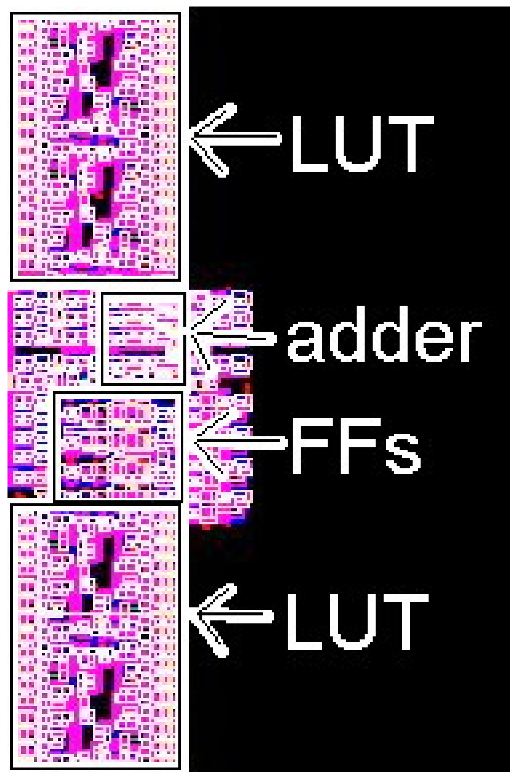


Figure 5-19: Layout of CLB

Figure 5-20 shows the top level layout of the testchip. The highlighted regions contain the three slices. Logic above the slices generates the inputs for the filtering operation. Logic below the slices captures the outputs of the filter and generates control signals. The large blocks to the left of the slices primarily contain the shift registers used for configuring the slices. As previously mentioned, a final version of the FPGA would replace these shift registers with small, localized SRAM blocks. The overall size of the chip is roughly 0.7μm by 1μm. The testchip has roughly 100 thousand devices.

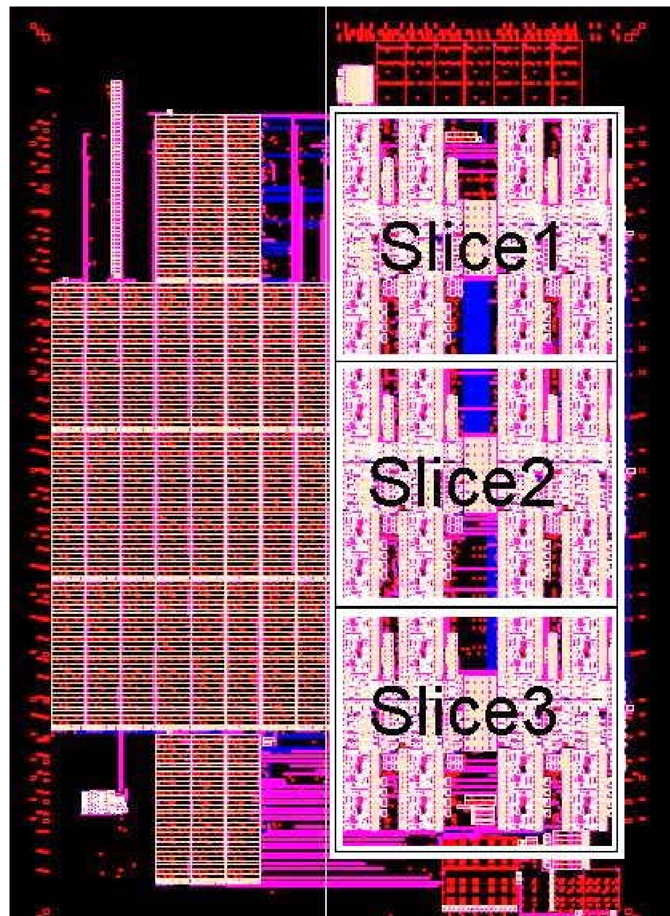


Figure 5-20: Layout of Chip

Chapter 6

Conclusion

6.1 Introduction

This chapter summarizes the project and provides some additional results. It lists the contributions of the thesis and suggests future directions for additional research.

6.2 Summary and Results

6.2.1 Flip-Flop Voltage Scaling

Many applications require a circuit to maintain state even during standby mode. The Leakage Feedback Flip-Flop (LFBFF) can maintain its state with the sleep signal asserted even when its inputs change. This ability implies that the combinational logic can be disconnected entirely from its power supply. Chapter 3 introduces the idea of reducing the power supply to the flip-flops during standby mode to provide additional power savings.

Analysis of the LFBFF with a reduced power supply shows the mechanisms of failure for the flip-flop. This analysis indicates how to size a LFBFF to function at lower supply voltages. The LFBFF sized to perform on the critical path of the testchip shows the ability to hold both states at 80mV supply. An analysis of the potential power savings using this technique suggests that the CLBs on the testchip

could see power reduction of several hundred times.

The sizing analysis of the LFBFF provides insight into sizing a flip-flop to fail at a specific supply voltage. Two failure flip-flops are shown to fail at 130mV. A new architecture using these failure flip-flops adjusts the supply voltage adaptively to provide maximum power savings without sacrificing robustness.

6.2.2 Low Power FPGA Testchip

The testchip designed for this thesis contains circuits for testing and demonstrating MTCMOS subthreshold leakage reduction techniques. It applies these techniques in a state-of-the-art 0.13μm technology. The circuit under test is a simplified version of an FPGA. The FPGA arcitecture groups 12 Configurable Logic Blocks (CLBs) into 3 slices. The CLBs inside these slices can connect in a number of ways based on bits used to configure the interconnect circuitry. This small FPGA architecture can provide a number of different functions, but the function of primary interest in this project is an 8-bit filter with 16 bits of precision. The testchip includes test circuitry that generates inputs for the filtering operation and that captures the output.

The CLB architecture itself uses distributed arithmetic for filtering operations. To facilite this form of filtering, each CLB contains four 16-bit lookup tables, a 4-bit adder, and a 4-bit register (of LFBFFs). The design partitions the CLB into four sleep regions based on these three primary elements and the remaining circuits. These distinct regions enter sleep during the active mode if a given configuration excludes their use. A careful analysis of the interface between sleep regions prevents short circuit currents due to floating nodes. These sleep regions can provide up to 2.3X total CLB steady-state power reduction during the active mode at 1.0V for some configurations.

The CLB uses three power supplies for the logic, flip-flops, and sleep signal respectively. The multiple supplies facilitate measurement and permit flip-flop voltage scaling. Table 6.1 shows the breakdown of total CLB power based on these supplies. The larger contribution of the flip-flop and sleep supplies during sleep mode occurs

Table 6.1: Breakdown of CLB Power

Operation Mode	Logic Supply	Flip-flop Supply	Sleep Supply
Active, Clock Gated	91.3%	8.6%	0.1%
Sleep	88.5%	9.7%	1.8%

because the total power decreases. The power savings in the logic power supply are higher than for the flip-flop supply, and the sleep supply power does not decrease in sleep mode.

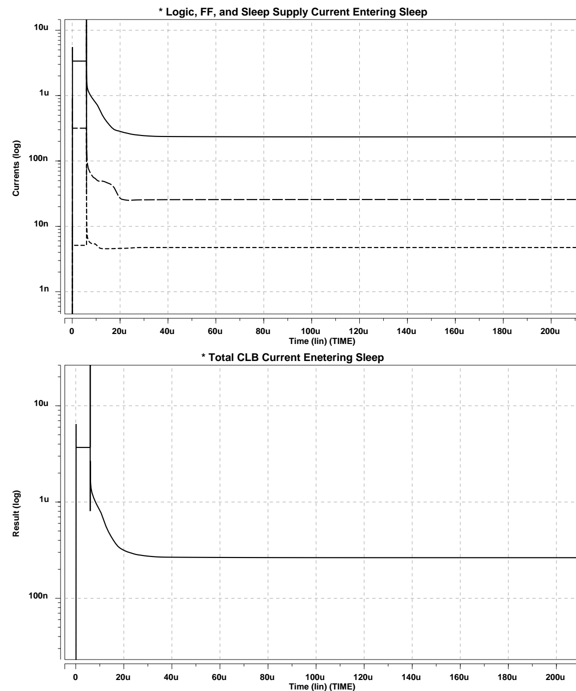


Figure 6-1: Leakage Current Reduction Entering Sleep

Figure 6-1 shows the normalized reductions in current for the CLB entering sleep mode. The top panel shows the three separate power supplies in the order (from the top): logic supply, flip-flop supply, and sleep supply. The bottom panel shows the total leakage current reduction entering sleep. The MTCMOS circuits provide a

14X reduction in leakage current (and power) during sleep mode. These savings are relative to the current in the active mode with the clock gated.

6.2.3 CLB Power Savings Using Voltage Scaling

Previous analysis of power savings shows that voltage scaling can reduce the power consumed by the flip-flops on the testchip by 25X during the standby mode. This reduction in power occurs in addition to the savings gained by placing the flip-flops into sleep mode from active mode ($\sim$ 12X). Thus, the total reduction in power consumed by the flip-flops from active mode (clock gated) to sleep mode at 80mV is about $12 \cdot 25 = 300$X. The flip-flops maintain their state correctly while achieving these savings.

The even more significant savings arise from the ability to turn off the combinational logic completely. Table 6.1 shows that the flip-flop power supply constitutes less than 10% of the CLB power for either mode of operation. This means that $\sim$ 90% of the CLB power can be immediately eliminated by reducing the combinational logic power supply to 0 during standby mode. The remaining $\sim$ 10% attributed to the flip-flops can be reduced by about 300X. If the sleep power supply remains at 1.0V, the total standby power of the CLB essentially comes from the sleep power supply. The sleep power supply constitutes 1.8% of the total CLB power in sleep mode at 1.0V, so keeping the sleep power supply at 1.0V reduces total CLB power by $\frac{100}{1.8} \cong 55$X.

Scaling the sleep power supply along with V_{DD} causes the flip-flop to fail a little earlier, but it still can function properly at 90mV with a power reduction factor of 16X. This means the total flip-flop power supply is reduced by $\sim$ 200X from active mode to standby mode. Furthermore, simulation shows that the sleep power supply sees a power reduction of 25X. The total power for the CLB in this case reduces to $88.5\% \cdot 0 + \frac{9.7\%}{200} + \frac{1.8\%}{25} \cong 0.12\%$. The total power reduction from active mode to deep standby mode is therefore $\frac{100}{0.12} \cong 800$X.

The application of voltage scaling along with other sleep mode techniques can dramatically reduce the power consumed in standby mode. For a battery powered device, this reduction shifts the attention to the active mode as the largest source

of energy consumption. These techniques could extend the standby lifetime of the experimental FPGA in this project by several hundred times.

6.3 Contributions

The following list points out the contributions of this thesis. Please recall that all numerical results come from simulation. The tapeout schedule prevents the testchip from being fabricated in time to include measured results in this document.

- Introduce supply voltage scaling as a method to reduce power consumption during standby mode while retaining state information.
- Analyze the failure mechanisms for Leakage Feedback Flip-Flop (LFBFF) holding state with scaling supply voltage.
- Suggest an architecture for adaptively scaling the supply voltage for flip-flops during standby mode.
- Provide method for sizing the LFBFF to fail at higher supply voltage.
- Show a LFBFF holding both states at 80mV.
- Demonstrate Leakage Feedback Flip-Flop (LFBFF) in silicon.
- Apply MTCMOS leakage reduction techniques to a new FPGA architecture.
- Demonstrate 14X power reduction in sleep mode with $< 10\%$ active mode delay penalty.
- Introduce configurable sleep regions in CLB architecture.
- Demonstrate active mode power reduction up to 2.3X using some configurations.
- Analyze interface of sleep regions with active regions.
- Analyze sneak leakage paths in CLB and provide additional characterization.

6.4 Future Work

6.4.1 Voltage Scaling

The voltage scaling technique offers significant power savings, but a lot of work is necessary to make it practical. For example, selecting or designing an appropriate adjustable power supply is necessary for the adaptive voltage scaling method. The details of detecting a failure in the failure flip-flops and of controlling the power supply remain to be determined. Also, more detailed characterization of the LFBFF could possibly produce improvements on the design.

6.4.2 Low Power FPGA Testchip

The testchip will provide several interesting sets of results once fabricated. First, standby leakage reduction will be measured both in and out of sleep mode. The maximum speed of the circuit will be compared to simulation to verify the delay penalties incurred. Additionally, the circuit will contain the hooks necessary for independently changing the supply voltage to various parts of the circuit. This will permit functional testing in silicon of voltage scaling with leakage feedback flip-flops.

Bibliography

[1] James Kao and Anantha Chandrakasan. MTCMOS Sequential Circuits. In *Proc. ESSCIRC 2001*, 2001.

[2] Shekhar Borkar. Leakage Reduction in Digital CMOS Circuits. In *ISSCC'02, Special Topic Evening Session SE2*, Feb. 2002.

[3] M. Horiguchi, T. Sakata,K. Itoh. Switched-Source-Impedence CMOS Circuit for Low Standby Subthreshold Current Giga-Scale LSI's. In *IEEE JSSC*, Vol 28, No 11, pages 1131-1135, November 1993.

[4] Jonathan P. Halter and Farid N. Najm. A Gate-Level Leakage Power Reduction Method for Ultra-Low-Power CMOS Circuits. In *Proc. IEEE 1997 Custom Integrated Circuits Conference*, pages 475-478, 1997.

[5] Srinath R. Naidu and E. T. A. F. Jacobs. Minimizing stand-by leakage power in static CMOS circuits. In *Proc. Design, Automation and Test in Europe, 2001*, pages 370-376, March 2001.

[6] Narendra, S., S. Borkar, Vivek De, Dimitri Antoniadis, and Anantha Chandrakasan. Scaling of Stack Effect and its Application for Leakage Reduction. In *International Symposium on Low Power Electronics and Design, 2001*, pages 195-200, August 2001.

[7] Sundararajan, Vijay, and Keshab K. Parhi. Low Power Synthesis of Dual Threshold Voltage CMOS VLSI Circuits. In *International Symposium on Low Power Electronics and Design*, 1999.

[8] James T. Kao and Anantha Chandrakasan. Dual-Threshold Voltage Techniques for Low-Power Digital Circuits. In *IEEE Journal of Solid-State Circuits*, Vol 35, No 7, July 2000.

[9] Mutoh, Shin'ichiro, Takakuni Douseki, Yasuyuki Matsuya, Takahiro Aoki, Satoshi Shigematsu, and Junzo Yamada. 1-V Power Supply High-Speed Digital Circuit Technology with Multithreshold-Voltage CMOS. In *IEEE JSSC*, Vol 30, No 8, August 1995.

[10] Michael Powell, Se-Hyun Yang, Babak Falsafi, Kaushik Roy, and T. N. Vijaykumar. Gated-V_{dd}: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories. In *Proc. of the 2000 International Symposium on Low Power Electronics and Design*, pages 90-95, July 2000.

[11] Michael Powell, Se-Hyun Yang, Babak Falsafi, Kaushik Roy, and T. N. Vijaykumar. Reducing Leakage in a High-Performance Deep-Submicron Instruction Cache. In *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol 9, No 1, February 2001.

[12] Sakata, Takeshi, Kiyoo Itoh, Masashi Horiguchi, and Masakazu Aoki. Subthreshold-Current Reduction Circuits for Multi-Gigabit DRAM's. In *IEEE JSSC*, Vol 29, No 7, July 1994.

[13] T. Inukai, M. Takamiya, K. Nose, H. Kawaguchi, T. Hiramoto, and T. Sakurai. Boosted Gate MOS (BGMOS): Device/Circuit Cooperation Scheme to Achieve Leakage-Free Giga-Scale Integration. In *Custom Integrated Circuits Conference, 2000. CICC. Proceedings of the IEEE 2000*, pages 409-412, May 2000.

[14] Hiroshi Kawaguchi, Ko-ichi Nose, and Takayasu Sakurai. A CMOS Scheme for 0.5V Supply Voltage with Pico-Ampere Standby Current. In *IEEE International Solid-State Circuits Conference*, pages 192-193, 1998.

[15] Shigematsu, Satoshi, Shin'ichiro Mutoh, Yasuyuki Matsuya, and Junzo Yamada. A 1-V High-Speed MTCMOS Circuit Scheme for Power-Down Applications In *Symposium on VLSI Circuits Digest of Technical Papers*, 1995.

[16] Mutoh, Shin'ichiro, Satoshi Shigematsu, Yasuyuki Matsuya, Hideki Fukuda, and Junzo Yamada. A 1V Multi-Threshold Voltage CMOS DSP with an Efficient Power Management Technique for Mobile Phone Application In *IEEE ISSCC*, 1996.

[17] Shigematsu, Satoshi, Shin'ichiro Mutoh, Yasuyuki Matsuya, Yasuyuki Tanabe, and Junzo Yamada. A 1-V High-Speed MTCMOS Circuit Scheme for Power-Down Application Circuits In *IEEE JSSC*, Vol. 32, No 6., June 1997.

[18] P. R. van der Meer, A. van Staveren, A. H. M. van Roermund. Ultra-low Standby Currents for deep sub-micron VLSI CMOS Circuits: Smart Series Switch. In *IEEE International Symposium on Circuits and Systems*, May 2000.

[19] James T. Kao. Subthreshold Leakage Control Techniques for Low Power Digital Circuits. Doctor of Philosophy in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology. May 2001.

[20] J. Sheu, et. al. BSIM: Berkeley short-channel IGFET model for MOS transistors. In *IEEE JSSC*, SC-22, 1987.

[21] Miura-Mattausch, M.,M. Suetake,H. J. Mattausch,S. Kumashiro,N. Shigyo, S. Odanaka,N. Nakayama. Physical modeling of the reverse-short-channel effect for circuit simulation In *Electron Devices, IEEE Transactions on*, Vol. 48, Iss. 10, October 2001. Page(s): 2449 -2452

[22] Hane, M., T. Ikezawa, M. Hiroi, and H. Matsumoto. Dopant diffusion model refinement and its impact on the calculation of reverse short channel effect In *International Electron Devices Meeting (IEDM)*, 1996.

[23] Ono, Atsuki, Ryuichi Ueno, and Isami Sakai. TED Control Technology for Suppression of Reverse Narrow Channel Effect in 0.1μm MOS Devices. In *IEDM*, 1997.