Emerging Power-Gating Techniques for Low Power Digital Circuits

Michael B. Henry

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Engineering

Leyla Nazhandali, Chair
Wu-chun Feng
Mary Jane Irwin
Cameron D. Patterson
Joseph G. Tront

November 3rd, 2011
Blacksburg, Virginia

Keywords: Digital Electronics, Low Power, Power-Gating,
NEMS, Sense-amplifier Pass Transistor Logic

Emerging Power-Gating Techniques for Low Power Digital Circuits

Michael B. Henry

ABSTRACT

As transistor sizes scale down and levels of integration increase, leakage power has become a critical problem in modern low-power microprocessors. This is especially true for ultra-low-voltage (ULV) circuits, where high levels of leakage force designers to chose relatively high threshold voltages, which limits performance. In this thesis, an industry-standard technique known as power-gating is explored, whereby transistors are used to disconnect the power from idle portions of a chip. Present power-gating implementations suffer from limitations including non-zero off-state leakage, which can aggregate to a large amount of wasted energy during long idle periods, and high energy overhead, which limits its use to long-term system-wide sleep modes. As this thesis will show however, by vastly increasing the effectiveness of power-gating through the use of emerging technologies, and by implementing aggressive hardware-oriented power-gating policies, leakage in microprocessors can be eliminated to a large extent. This allows the threshold voltage to be lowered, leading to ULV microprocessors with both low switching energy and high performance.

The first emerging technology investigated is the Nanoelectromechnical-Systems (NEMS) switch, which is a CMOS-compatible mechanical relay with near-infinite off-resistance and low on-resistance. When used for power-gating, this switch completely eliminates off-state leakage, yet is compact enough to be contained on die. This has tremendous benefits for applications with long sleep times. For example, a NEMS-power-gated architecture performing an FFT per hour consumes 30 times less power than a transistor-power-gated architecture. Additionally, the low on-resistance can lower power-gating area overhead by 36-83%.

The second technology targets the high energy overhead associated with powering a circuit on and off. This thesis demonstrates that a new logic style specifically designed for ULV operation, Sense Amplifier Pass Transistor Logic (SAPTL), requires power-gates that are 8-10 times smaller, and consumes up to 15 times less boot-up energy, compared to static-CMOS. These abilities enable effective power-gating of an SAPTL circuit, even for very short idle periods. Microprocessor simulations demonstrate that a fine-grained power-gating policy, along with this drastically lower overhead, can result in up to a 44% drop in energy.

Encompassing these investigations is an energy estimation framework built around a cycle-accurate microprocessor simulator, which allows a wide range of circuit and power-gating parameters to be optimized. This framework implements two hardware-based power-gating schedulers that are completely invisible to the OS, and have extremely low hardware overhead, allowing for a large number of power-gated regions. All together, this thesis represents the most complete and forward-looking study on power-gating in the ULV region. The results demonstrate that aggressive power-gating allows designers to leverage the very low switching energy of ULV operation, while achieving performance levels that can greatly expand the capabilities of energy-constrained systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the biggest challenges related to low-power design is mitigating and controlling the leakage power of a circuit. What was once considered a minor nuisance a decade ago has now become one of the top design priorities in modern microprocessors. The assault of leakage power has seemingly come from all sides. Technology scaling has allowed for more functionality per area, higher speeds, and lower switching energy, but has also increased leakage power exponentially. Likewise, supply voltage scaling has yielded a dramatic reduction in switching energy, but has typically been accompanied by a reduction in threshold voltage in order to maintain performance, which also increases leakage power. Finally, as the demand for more capability and performance continues to grow, the number of functional blocks and the amount of memory has increased dramatically, and more transistors means more leakage power.

High levels of leakage power have now impacted all levels of digital design, and has stopped supply voltage scaling in its tracks, which used to be the most popular method for keeping power consumption under control. The last few decades has seen a steady drop in the power consumption of integrated circuits, due to the reduction of supply voltage, which at one time used to be as high as 5 V. To prevent any lost performance, this was accompanied by a drop in threshold voltage, which also increases the severity of leakage. With the recent

dominance of leakage power, however, it became untenable to continue to reduce the threshold voltage, which meant the supply voltage for performance-constrained applications has remained around 1 V for the last decade.

This has not stopped researchers and designers from continuing to reduce the supply voltage for applications that are more energy-constrained than performance-constrained. Recent advances in circuit design have allowed researchers to operate chips with supply voltages well below the range normally used by integrated circuits. In the so-called *Ultra-Low-Voltage* (ULV) region, switching energy is dramatically reduced, which is advantageous for emerging applications with very tight energy budgets. Without the accompanying reduction in threshold voltage, however, performance is seriously impacted, so current ULV prototypes are only suited for non-performance-constrained applications. This clearly demonstrates how leakage power is intertwined with switching energy and performance: by mitigating leakage power, designers can push supply voltages into the ULV region, while at the same time lowering the threshold voltage. This would lead to a large drop in switching energy, while keeping performance levels constant, thus greatly expanding the capabilities of energy-constrained integrated circuits.

One of the most obvious places to mitigate leakage power is during standby periods, when no processing is occurring. To accommodate the growing popularity of mobile, wireless, and sensor processing, all of which have substantial idle times, the microprocessor industry has widely adopted power-gating and software-controlled sleep modes. With this technique, transistors are used to disconnect the power from unused portions of a microprocessor, which substantially reduces leakage power. This approach is attractive because it mitigates leakage without requiring any modification to the logic or operation of the power-gated circuit. That being said, there are still many design challenges related to power-gating, as well as inefficiencies that limit its effectiveness.

In this thesis, emerging technologies are examined that target the inefficiencies of power-gating. Special attention is paid to the ULV region, where large reductions in leakage can

also lead to a large drop in switching energy. The first inefficiency stems from the unavoidable aspects of using transistors as power-gates. When a circuit is turned off, the transistor power-gates themselves leak, meaning leakage is reduced, but not eliminated. Also, a voltage drop is created across the power-gate, which lowers the effective supply voltage of the power-gated circuit, thus slowing it down. Making a transistor power-gate wider decreases the voltage drop, but increases the off-state leakage, meaning a trade-off exists. This trade-off can make it exceedingly difficult for designers to size the power-gates, since different operating conditions may have different optimal sizings. In this thesis, an emerging CMOS-Compatible technology, the Nanoelectromechanical-Systems (NEMS) Switch, is investigated as an alternate power-gating structure. Due to an air-gap, NEMS switches have a nearly infinite off-resistance, which essentially eliminates off-state leakage. They also have a very low on-resistance, which greatly mitigates the performance drop of power-gating. NEMS-based power-gating is examined in a wide variety of scenarios, including long-term power-gating, where the switch-rate is low, and fine-grained power-gating of the functional units of a microprocessor, where a unit may be frequently power-gated for short periods of time.

The second inefficiency of power-gating results from the energy overhead associated with powering a functional block on and off. When power-gating is implemented, there are varying levels of granularity at which an unused functional unit can be powered off. In a coarse-grained approach, software-controlled sleep modes are implemented which shut down the entire processor during the long idle periods. More fine-grained approaches shut down idle functional units of a microprocessor, even while active processing is occurring, generally using either software or hardware control. The limit to the granularity of power-gating is the energy overhead: if it takes a lot of energy to turn a unit off and on, then it may not be possible to target short idle periods, since the leakage energy saved would be outweighed by the power-gating energy overhead.

The energy overhead of static-CMOS is generally high due to the high power-network capacitance and large current spikes. In this thesis, an alternative form of logic is investigated: Sense-Amplifier-Pass-Transistor-Logic (SAPTL). This style of logic, originally designed for

ultra-low-voltage operation in highly variant environments, also has significantly lower power-gating overhead. This opens the door to a new paradigm: *Ultra-Fine-Grained* power-gating. With this approach, functional units are shut down almost immediately after use, which, when paired with a power-gating system with very low overhead, leads to large drops in energy consumption.

As part of these investigations, a detailed analytical energy estimation framework is presented that can quickly optimize the various parameters of a power-gated system, including supply voltage, threshold voltage, transistor power-gate width, NEMS switch selection, etc. Integrated with this energy framework is a cycle-accurate simulator with the capability of power-gating individual functional units. The cycle-accurate simulator implements two power-gating policies that can be implemented completely in hardware with very little overhead. Thus, not only does this thesis present the most complete and methodical investigation into power-gating in the ULV region, covering nearly every aspect of its implementation, but it also looks forward into how emerging technologies and logic styles can greatly improve its efficiency.

1.1 Organization and Key Findings

After a brief background on ULV operation and power-gating in Chapter 2, Chapter 3 analyzes long term power-gating using NEMS switches as power-gating structures. This is done for an FFT application using a comprehensive simulation framework and NEMS energy model. The key findings of this chapter are as follows:

- The infinite off-resistance of NEMS switches leads to large reductions in average power consumption for architectures with long sleep periods. As an example, a NEMS-power-gated architecture performing an FFT every hour consumes 30 times less power than a transistor-power-gated architecture.

- The area overhead of a power-gating implementation using state-of-the-art NEMS prototypes is estimated to be 36-83% lower, compared to transistor power-gates. Further improvements to the contact resistance of the switches would yield a much larger reduction.

- By eliminating off-state leakage, architecture selection becomes more straightforward. With the zero off-state leakage of NEMS switches, no matter how long the idle periods are, the optimum architecture is generally the one that is the most efficient in terms of switching energy.

Chapter 4 analyzes fine-grained functional unit power-gating, where individual functional units of a microprocessor, such as the integer ALU, floating point unit, etc., are shut off when not in use through the use of a hardware-based power-gating policy. Both an ideal oracle policy and a simple hardware-based policy are simulated, and both NEMS and transistors are compared as power-gating structures. This chapter also presents the power-gating energy estimation framework and the microprocessor simulation environment, which are also used in Chapter 5. The key findings of this chapter are as follows:

- Functional unit power-gating pushes the optimum supply voltage and threshold voltage lower by 100-250 mV, which positively benefits all benchmarks, regardless of their functional unit usage behavior. Even a highly active floating point FFT benchmark that rarely shut any units off sees a 20.7% drop in total functional unit energy.

- With an ideal oracle-based scheduler, NEMS-based functional unit power-gating yields a 29.5% average reduction in total functional unit energy with respect to no functional unit power-gating, while an ideal transistor-based system yields a 23.5% reduction. While the difference is not overwhelming, it must be taken into account that NEMS switches are truly advantageous for long idle periods. These results, however, demonstrate that the high activation energy and delay of NEMS switches do not hinder their use when targeting shorter idle periods.

- With the more realistic hardware-based Flag Policy, NEMS power-gating yields a 28.9% drop in energy, while transistor power-gating yields a 23.0% drop in energy. As is evident, the Flag Policy achieves results that are very close to the ideal policy.

In Chapter 5, Ultra-Fine-Grained power-gating is examined, which targets very short idle periods of microprocessor functional units. A recently developed style of logic, SAPTL, is examined and the power-gating energy overhead is compared to CMOS through the simulation of hand-crafted small circuits and synthesis of larger circuits. The effect of the lower power-gating energy overhead of SAPTL on the effectiveness of functional unit power-gating of a microprocessor is then determined using the simulation framework from Chapter 4. The key findings of this chapter are as follows:

- Compared to CMOS, SAPTL requires power-gating structures that are 8-10 times smaller than CMOS. When booting up from a sleep mode, SAPTL circuits consume up to 40 times less energy.
- When integrated into the microprocessor simulations, this lower overhead, along with an extremely simple scheduler that simply shuts a functional unit off immediately after its use, leads to as high as a 44% drop in total functional unit energy.

The 44% drop in total functional unit energy that was achieved in Chapter 5 is remarkable considering the fact that this was achieved without any delay penalty, without any change to the logical structure, and in such a way that is completely invisible to the operating system or program that is running. This was accomplished solely by eliminating leakage and reducing V_{DD} and V_{TH}.

Chapter 6 offers conclusions as well as recommendations for future work on ULV power-gating. Appendix A presents a novel synthesis approach for SAPTL circuits, which is used in Chapter 5.

Chapter 2

Background

2.1 Ultra-Low-Voltage Operation

Supply voltage scaling has long been a popular method for energy and power reduction due to the fact that reducing the supply voltage decreases switching power consumption quadratically, while only increasing circuit delay linearly. Traditionally, voltage scaling has been limited to around 700-800 mV, mainly due to issues with SRAM reliability and performance loss [9]. In recent years, the application space has opened up for devices such as wireless and ubiquitous sensors, biomedical implants, etc. with relaxed performance constraints and extremely tight energy budgets. This has motivated the scaling down of energy per operation and has led researchers to examine the limits to how low supply voltage scaling can go.

Traditional CMOS operation has been described as having an on-network of transistors operating in saturation and an off-network in weak-inversion (or cut-off). In reality, the line between on and off can be blurred: as long as the gain of the transistors is greater than one, and the on-network has a stronger drive-strength than the off-network, then complementary logic is possible [34]. This opens up the possibility of CMOS devices operating with a supply voltage that is at or less than the threshold voltage of the transistors, in the so-called *Ultra-*

Low Voltage (ULV) region. The basic principles of ULV operation are the same as nominal voltage operation (nominal voltage being the typical supply voltage the foundry recommends; currently around 1.0 V). The only difference is that rather than the on-network in saturation and the off-network in weak-inversion, both networks are in weak-inversion. Owing to the still-existing gate bias, however, the on-network has a higher drive strength, which makes complementary logic possible.

Theoretical operation of CMOS operation at ULV voltages has been known for decades [33], but in recent years, the fast-paced growth in energy constrained applications and devices has driven researchers to overcome major obstacles and build functional prototypes. These accomplishments include accurate circuit models, low voltage SRAM that can operate down to 180 mV [34], design standards for robust cells that can overcome the high potential for functional failure [19], and methodologies that can mitigate the severe process variation that ULV circuits experience [14]. Recent prototypes exemplify this surge in research and demonstrate the huge energy savings that are possible. The Phoenix processor [13] is a 1 mm^2 ultra-low-voltage processor that consumes 226 nW when active and 35 pW in a carefully designed sleep mode. In [19], a 300 mV ULV version of the popular Texas Instruments MSP-430 is presented that consumes 12.2 uW when active and 1 uW in a sleep mode. In was demonstrated in [17] that ULV operation can be used for energy efficient high-performance applications through parallelization by fabricating an MPEG processor that achieves 411 billion operations per watt(GOPS/W) at 300 mV, compared to only 43 GOPS/W at nominal voltage [17].

A major challenge of ULV operation is selecting an appropriate supply voltage and threshold voltage that balances performance, switching energy, and leakage power. Even if a set clock frequency is decided upon, the supply and threshold voltage can be jointly raised and lowered, which maintains a constant performance, but optimizes for either leakage energy or switching energy. This optimum point is dependent on the activity factor of the circuit. Given a constant delay, if the activity factor of a circuit is high, then a low supply (V_{DD}) and threshold voltage (V_{TH}) is optimal, whereas if the activity factor is low, a high V_{DD} and

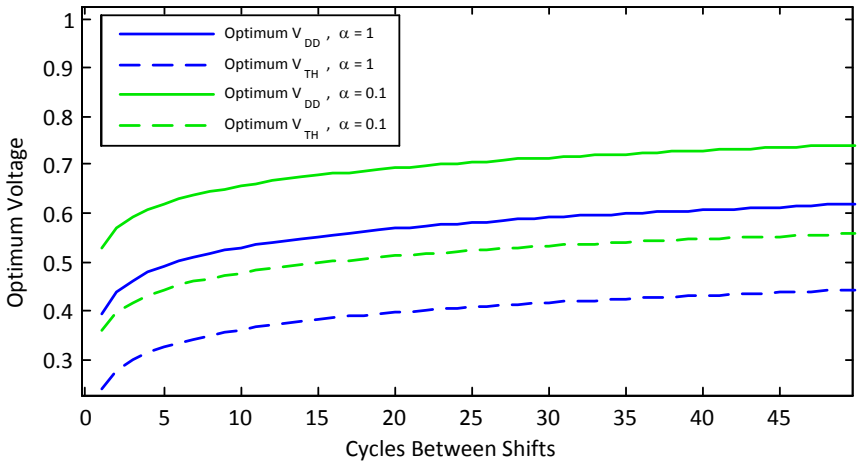

Figure 2.1: Optimum Supply and Threshold Voltage of a 32-bit barrell shifter with a clock frequency of 100 MHz.

V_{TH} is optimal. Figure 2.1 demonstrates this, which plots the optimum V_{DD} and V_{TH} for a 32-bit barrel shifter (130 nm technology, 100 MHz clock frequency), with respect to the number of idle clock cycles between shift operations. Two activity factors are shown, which correspond to the percentage of gates that switch during each shift operation. This graph clearly shows what was previously stated: high activity factors favor a low V_{DD} and V_{TH}, while low activity factors favor a high V_{DD} and V_{TH}.

It is conceivable to construct a circuit that can dynamically tune V_{DD} and V_{TH} to track the optimum voltages as the activity factor changes; circuits have been demonstrated that switch the power network between two V_{DD}s [26], and body biasing can be used to dynamically alter V_{TH} [18]. Actual implementation of a system such as this is exceedingly challenging, however. This is because 1) multiple power rails and body bias regions introduces significant additional area; 2) body biasing has greatly lost effectiveness in newer technologies due to higher doping concentrations [27]; and 3) tracking the activity factor in an energy efficient manner is a non-trivial task. A simpler approach, one that this thesis investigates thoroughly, is to use power-gating to shut off a circuit when not in use. By eliminating leakage during idle cycles, the activity factor is effectively increased, pushing the optimum V_{DD} and V_{TH} lower. The limiting factor to this approach is the effectiveness of power-gating. The present industry standard techniques used for power-gating are too energy inefficient to allow for

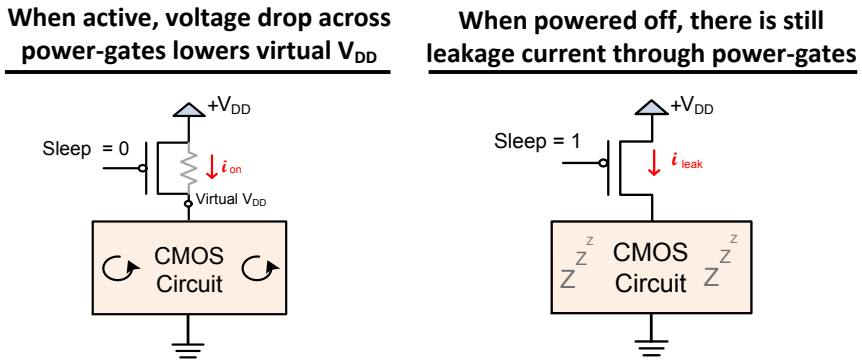

Figure 2.2: The limitations of power-gating: performance drop and non-zero off-state leakage.

rapid power-gating, and the off-state leakage is non-zero. This motivates the focus of this thesis, which details how emerging technologies can be used to vastly improve the efficiency of power-gating. This would allow for a single optimum V_{DD} and V_{TH} across a wide range of operating conditions, which would greatly increase the energy efficiency of ULV circuits.

2.2 Power-Gating

Transistor-based power-gating is implemented by placing sleep transistors in-line between the circuit and the power network (headers) or the ground network (footers). Footers are generally more area-efficient as the high n-type mobility means less of them are needed. That being said, most commerical designs implement headers due to easier design and analysis, especially when multiple power domains or an external switchable voltage regulator is used [35]. There are two major limitations of headers (and footers), which are outlined in Figure 2.2. First, the headers themselves leak which means leakage is reduced, but not eliminated. Second, when the headers are on, the current flowing though them creates a voltage drop on the power network that lowers the performance of the power-gated circuit [32]. This performance drop is especially severe with ULV processors, where delay is exponentially dependent on supply voltage; in this region, a small variation in voltage leads to a very large drop in performance.

The controlling factor of the trade-off between delay increase and off-state leakage reduction

is the width of the headers. A common industry practice is to size the headers such that the worst-case supply voltage drop is less than 10%, but this results in a large area overhead, and generally the sleep-mode leakage reduction is only around 90% [27]. During long idle periods, this remaining leakage can lead to significant amounts of wasted energy. The ultra-low power Phoenix processor took the opposite route and used a small amount of footers with non-minimal channel lengths [13]. This resulted in picowatt-range off-state power, but the large supply voltage drop limited the clock frequency to 106 kHz. This trade-off, and the desire to achieve a large reduction in off-state leakage without impacting performance motivates one of the examinations in this thesis, where an emerging technology, the CMOS-Compatible NEMS switch, is used as a power-gating structure.

Granularity of Power-Gating:

An important aspect of power-gating is the *granularity* at which it is performed. *Temporal granularity* determines how short the targeted idle periods can be, whereas *regional granularity* determines how many separable power-gating regions there are. It is usually the case that these two are intertwined. The most coarse-grained approach is to use an external switchable regulator. Switching off the voltage regulator would effectively eliminate on-chip leakage (though the regulator itself would still leak), but it is very coarse grained in both regards. First, the capacitance of the switched power network would be extremely large and would take a significant amount of time and energy to recharge upon boot-up [35], meaning only extremely long idle periods could be targeted. Second, each power-gated region would require separate power supply pins. In a pin-constrained chip, this places limits on the number of potential regions.

Modern low-power microprocessors tend to take a more fine-grained approach. It is now nearly universal that power-gating structures are moved on chip to avoid having to recharge the large external power network and decoupling capacitance present on-chip. There is also a great deal of active research into having separate power-gated regions of the chips that are individual targeted. One example is functional unit power-gating, where portions of a

microprocessor, such as the integer and floating point ALU, memory, I/O, are individually shut off when not in use. Many studies have been performed on functional unit power-gating, and they tend to focus on selecting appropriate sizes for headers, modeling the break-even cycle count of a functional unit [15], or crafting policies that increase the time a unit spends powered off [20, 29].

The goal of this thesis is to effectively eliminate leakage in such a way that a single low V_{DD} and V_{TH} becomes optimal. As such, granularity is a very important aspect that will be extensively analyzed. NEMS-based power-gating, which is presented in the next two chapters, allow for leakage elimination that is equivalent or better than what is achievable with an external switchable regulator. It is also CMOS-compatible and can be implemented on-chip, which means that the high external capacitances can be avoided, and much smaller power-gated regions can be targeted. Chapter 5 also focuses on granularity and introduces a new paradigm: *Ultra-Fine-Grained* power-gating. By greatly reducing the overhead of turning functional units on and off, very brief idle periods can be targeted and the level of leakage elimination can be greatly increased. Finally, a major focus of Chapter 4 and 5 is simple, hardware-only power-gating schedulers. By keeping the schedulers simple and in hardware, more regions can be power-gated, and their on/off state can be completely invisible to the OS.

Chapter 3

Effective Long-Term Power-Gating with NEMS Switches

3.1 Introduction

A major limitation of CMOS power-gates is the fact that they do not completely eliminate off-state leakage when a circuit is powered off. For the rapidly emerging class of devices with long idle periods, such as environmental sensor networks, biomedical implants, wireless security systems, etc., the residual off-state leakage can accumulate to a large amount of wasted energy. The transistor power-gates can be made narrower or longer in order to reduce the residual leakage, however this increases the effective resistance of the power-gates, which slows down the circuit. This leads to a counterproductive trade-off between the amount of off-state leakage and the performance penalty. The only remaining method to combat the off-state leakage, while maintaining performance, is to raise both the threshold and supply voltage of the chip, which moves it away from the ULV region and deprives it of the low-energy benefits.

The over-arching parameter that determines the trade-off between leakage reduction and

delay-penalty is the on/off ratio of the transistor; a high on-current minimizes the voltage drop across the power-gating structure, while a low off-current minimizes residual off-state leakage. It is possible to use non-minimal lengths and adjust the sizing of a transistor to maximize the on/off ratio, but there is an upper limit and it is still too low for effective power-gating of energy constrained devices with long idle periods. Additionally, technology scaling has caused a large drop in the on/off ratio meaning power-gating is less effective at newer technology nodes. This motivates the search for a new power-gating structure with a significantly higher on/off ratio that can be seamlessly integrated onto a CMOS die, and has a reasonable area and energy overhead.

One leading contender is the CMOS-Compatible Nanoelectromechanical Systems (NEMS) switch, which is a nano-scale mechanical relay that has a near-infinite off resistance, low on-resistance, and a nanosecond-range mechanical switching delay [11]. Researchers of these devices envision entire circuits comprised of NEMS-based complementary logic, which would be entirely free of leakage power, but issues with switching speed and device lifetime have hampered these efforts [25]. However, a more feasible intermediate step would be a hybrid system with on-die NEMS switches acting as power-gates for CMOS circuitry. With the periodic systems described previously, the actuation rate for power-gates would be low enough that the billion-cycle lifetime of the NEMS devices is acceptable. Additionally, the sub-100 ns mechanical delay of a NEMS switch is tolerable for power-gating, since the wake-up times of most systems are in the microsecond range [27]. Thus, the zero off-leakage, low on-resistance, and on-die integration make NEMS switches very attractive for power-gating. This chapter presents an investigation that models and simulates all important aspects of CMOS-NEMS integration for power-gating and provides results that should motivate both industry and academia to pursue further research and eventual fabrication and commercialization of CMOS-NEMS systems.

Included in this chapter is a comprehensive study of NEMS-based power-gating in the context of coarse-grained power-gating of an FFT architecture. First, a background is provided on both the operation and integration of NEMS switches. Next, a simple energy model is

presented that provides insight into the behavior of a NEMS-power-gated circuit. The rest of this chapter presents the simulation framework that is used to compare the effectiveness of NEMS power-gates to transistor power-gates across a variety of metrics, including area overhead, activation energy, performance loss, and sleep-mode leakage. We do this for a range of FFT architectures and technologies and show that the many advantages of NEMS switches warrant further investigation into their use as power-gates. More specifically, the contributions of this chapter are as follows:

- We provide a system level simulation framework for NEMS-based power-gating that captures the major sources of energy, performance, and area overhead.
- Using an FFT processor benchmark, we compare the effectiveness of NEMS-based power-gating to transistor-based power-gating in the presence of aggressive voltage scaling and across a range of operating conditions and system-level parameters, such as core complexity and technology node.
- We parameterize the NEMS switch geometry and contact resistance and determine their effect on the energy and area overhead of power-gating.

Our findings are as follows:

- The infinite off-resistance of NEMS switches leads to large reductions in average power consumption for architectures with long sleep periods. As an example, a NEMS-power-gated architecture performing an FFT every hour consumes 30 times less power than a transistor-power-gated architecture.
- With transistor power-gates, the optimum FFT architecture, in terms of complexity, depends heavily on how often the FFT is performed. For example, architectures that favor low leakage power over switching energy efficiency are optimal at very low throughputs. With NEMS power-gates, due to the zero off-state leakage, the optimal architecture is always the one with the lowest switching energy consumption, which results in an architecture that is flexible for changing throughputs.

- The activation energy of the NEMS power-gate switch array in a 130 nm technology is in the nanojoule range, the activation time is under 40 μs, and the on-power is effectively zero. The source of this energy and delay overhead is primarily from the charge-pump boot-up. These overhead values fall dramatically with lower technologies.
- The area overhead of a power-gating implementation using current NEMS prototypes is 36-83% lower, compared to transistor power-gates. Further improvements to the contact resistance of the switches would yield a much larger reduction.

3.2 CMOS-Compatible NEMS Switches

Nano-Electro-Mechanical System (NEMS) switches are nano-scale relays with a suspended cantilever beam (representing the source) that bends under electrostatic force from the gate to form an ohmic contact with the drain. The off-resistance is nearly infinite due to the air gap between the source and drain, while the on-resistance is low due to the ohmic contact. The overriding motivation of NEMS researchers has been to develop entire circuits comprised solely of NEMS switches, and in fact complementary and complex logic built solely with these switches has been demonstrated [7]. Two major drawbacks have stifled progress on this front: first, the switching speed ranges from 10-100 ns, which is too slow for logic [6]; second, the reported lifetime has ranged from 10E8 to 10E12 on-off cycles [21, 25] which does not guarantee a long enough lifetime for logic implementation. Since the switching is much less frequent with power-gating, the implications of these drawbacks are not as severe.

Principles of Operation:

Figure 3.1 gives a compact layout of a laterally actuated electrostatic NEMS switch and is based on previously fabricated devices [6, 7, 25]. The device operates by electrostatic force between the gate and a cantilever beam suspended across a trench in the substrate. When the voltage of the gate reaches the activation voltage of the switch, the electrostatic force overcomes the elastic force of the beam, which then collapses towards the drain. This forms

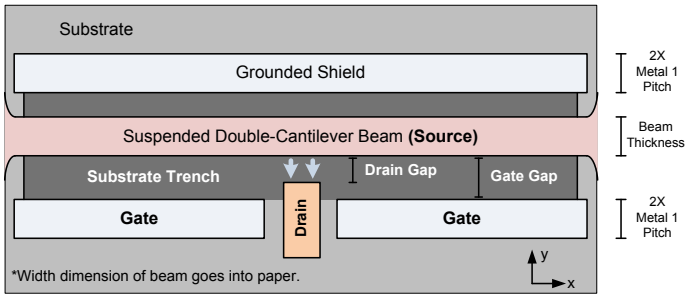

Figure 3.1: Layout of a compact double-cantilever NEMS switch supsended over a trench.

an ohmic contact that permits current to flow from the source to the drain. A wide range of contact resistances have been reported, ranging from 83 Ω with gold contacts [11], to several thousand Ω with contact materials such as platinum [25] or aluminum oxide [16]. The primary reason for this wide range is that most prototypes are designed for NEMS-based digital logic, where the high mechanical delay completely dominates the electrical delay (and a high resistance is beneficial by limiting the current draw).

Fabrication:

NEMS switches are considered CMOS compatible for three reasons. First, the structural and contact materials can be built using CMOS materials such as polysilicon, platinum, and tungsten. Second, the switches can be patterned using photolithography and released (i.e. suspended over an air gap) using techniques that already exist in CMOS flows. Finally, the entire switch fabrication flow does not use temperatures or process steps that will harm the CMOS circuitry. Fabrication of NEMS switches broadly involves three steps. First, a sacrificial layer, represented by the dark gray rectangle in Figure 3.1, is patterned. Next, the NEMS structure, consisting of the beam, source, and gates, are patterned using polysilicon or metal, with the beam being patterned on top of the sacrificial layer. At this point, a low-resistivity metal coating can be applied to reduce the contact resistance. Finally, the sacrificial layer is removed leaving an air-gap above which the beam is suspended. Detailed fabrication steps of NEMS switches using standalone MEMS/NEMS processes can be found in [6, 7, 23, 25]. Additionally, Figure 3.2 presents a CMOS-NEMS hybrid manufacturing flow, originally proposed in [8], that requires the addition of only three process steps to the

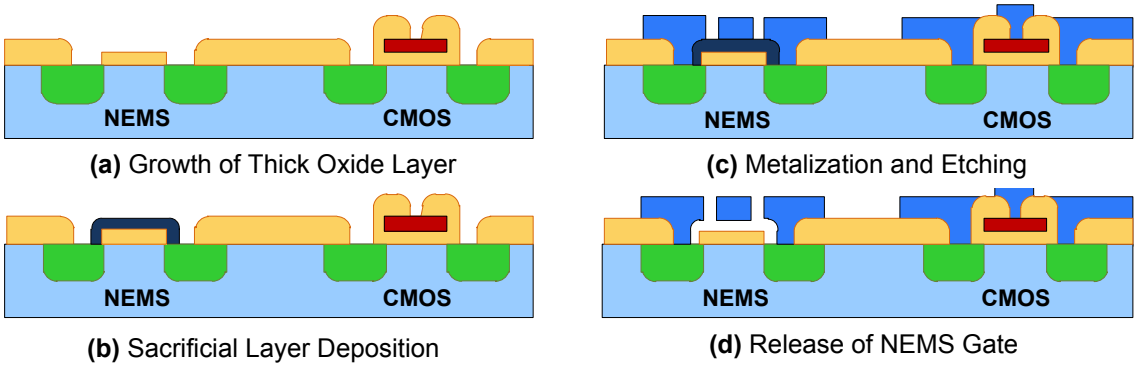

Figure 3.2: Hybrid CMOS-NEMS production flow. Only three additional masks would need to be added to the CMOS process.

standard CMOS process flow.

It is also conceivable to manufacture the NEMS switches on a separate die and use 3DIC integration techniques to bond it to the logic die. This would be advantageous in two situations: first, if designers wish to exploit a state-of-the-art process which may not yet have NEMS integrated into the flow, they can manufacture the NEMS switches on an older, more mature process; second, a 2D array of switches can be mass produced on a state-of-the-art technology with the through-silicon-vias in predetermined spots. Designers could then connect to this array through 3D integration and use as many switches as they need. This could greatly drive down the costs because a single NEMS array design could be reused for thousands of different applications. This would be similar to the gate-array or sea-of-gates approaches.

Tunable Parameters:

Adjusting the beam length, thickness, and width (width dimension goes into the substrate trench), as well as the drain gap and the gate gap allows for both the switching speed and activation voltage to be tuned [6]. A longer, narrower beam generally results in a slower switch with a lower activation voltage,which translates to lower energy per actuation, while a shorter, wider beam produces a faster switch with a higher activation voltage. A narrower gate gap also produces a lower activation voltage, but the gap is limited by the lithography

of the technology. This implies that, much like transistors, the scaling down of feature sizes can have a direct positive impact on the activation voltage and energy per actuation.

3.3 Theoretical Analysis of Transistor- and NEMS-based Power-Gating

In order to provide insight into the behavior of transistor- and NEMS- power-gated systems, we derive some basic equations for the energy and power in the presence of voltage scaling. We first pay attention to the fact that voltage scaling and power-gating are not two independent knobs for power reduction. For example, if a design at nominal voltage is four times faster than the required throughput, it should spend 75% of the time power-gated. If we use voltage scaling to reduce its speed by half, it should spend 50% of the time power-gated. Therefore, depending on the efficiency of power-gating and the gain from voltage scaling, any of these solutions might be superior. The purpose of this section, therefore, is not to derive an analytical solution for optimizing power (which would be prohibitively difficult), but rather to describe the relationship between switch characteristics, power, supply voltage and target throughput. It should be noted that this is a simple model suitable for long idle periods. Chapter 5 presents a much more detailed model suitable for more fine-grained power-gating that accounts for power-gating overhead.

Let us define target throughput, R_T, as the desired number of tasks performed per second (for example, FFTs per second). Also, let us define energy, E, as energy per task. Average power consumption, P, can be determined by $R_T \cdot E$. For any given design, switching energy, E_S, leakage power, P_L, and maximum throughput, R_M, are all dependent on supply voltage and can be considered functions of V. Using these definitions, the average power consumption P of a non-switched, i.e. not power-gated, circuit with respect to R_T and supply voltage V, is defined as:

$$P(V, R_T) = R_T \cdot E_A(V) + P_L(V) \tag{3.1}$$

It should be noted that with non-switched circuits, as R_T approaches 0, the average power consumption approaches P_L, meaning the least complex design would consume the least amount of power.

3.3.1 Transistor-Based Power-Gating

When transistor power-gates (headers) are used, they affect both the critical delay and leakage power of a circuit. To examine this, we use the terminology from [32], which identifies two effects that headers have on a circuit: K_D is the delay penalty, and is the factor by which a circuit slows down due to the voltage drop across the header. K_L is the leakage reduction, and is the factor by which the leakage is reduced when the power switch is off. K_D and K_L are dependent on the relative header width, W_H, which is defined as the width of the header divided by the sum of the widths of the NFETs of the circuit. The duty cycle, D, of the circuit is the the ratio between the time the switch has to be on and the total cycle time. Equation (3.2) provides the equation for duty cycle with respect to V, R_T, and W_H.

$$D(V, R_T, W_H) = \frac{R_T}{R_M(V)/K_D(W_H)} \tag{3.2}$$

By modifying (3.1), we can get (3.3) that provides the average power equation for transistor switched circuit:

$$P(V, R_T, W_H) = R_T \cdot E_S(V) + \frac{(1-D)P_L(V)}{K_L(W_H)} + D \cdot P_L(V) \tag{3.3}$$

As the target throughput, i.e. R_T and D, approaches zero, the power consumption does not go to zero, but instead consumes $P_L/K_L(W_H)$ watts.

In [13], it was shown that it is possible to target very low-throughput architectures by using a very small header. However, if the header is too small, a large voltage drop will result and the effective supply voltage of the circuit may be below the minimum operational voltage of the circuit, rendering the circuit inoperable. Once this header size limit is reached, $K_L(W_H)$ reaches its maximum value and no matter how low the throughput goes, the power consumption will always be at $P_L(V)/K_L(W_H)$.

3.3.2 NEMS-Based Power-Gating

NEMS power-gates affect the power and energy consumption in three ways. First, the sleep-mode power becomes zero due to the effectively infinite off-resistance of the NEMS switch. Second, a certain amount of activation energy, E_{ACT} is needed to actuate the NEMS switches. Finally, the NEMS switches consume power when on, represented as P_{SW}[1]. With these changes, Equation (3.1) can be rewritten for NEMS-gated circuits as follows, where $K_D(\Omega)$ is the delay penalty due to the finite contact resistance of the switch array:

$$P(V, R_T) = R_T(E_A(V) + E_{ACT}) + \frac{R_T}{R_M(V)/K_D(\Omega)}[P_L(V) + P_{SW}] \tag{3.4}$$

Rearranging (3.4) gives (3.5):

$$P(V, R_T) = R_T \cdot [E_A(V) + E_{ACT} + \frac{P_L(V) + P_{SW}}{R_M(V)/K_D(\Omega)}] \tag{3.5}$$

Let E_{MIN} be the minimum energy of a circuit, across all supply voltages, as shown in (3.6) and V_{opt} be the voltage where E_{MIN} is achieved:

$$E_{MIN} = \min_{\forall V}[E_A(V) + E_{ACT} + \frac{P_L(V) + P_{SW}}{R_M(V)/K_D(\Omega)}] \tag{3.6}$$

As long as $R_M(V_{opt})$ is equal or more than R_T, which holds for lower-throughput applications, (3.5) becomes:

$$P(R_T) = R_T \cdot E_{MIN} \qquad : R_T < R_M(V_{opt}) \tag{3.7}$$

[1]It will later be shown that this on-power is negligible for electrostatic switches. Other classes of switches, such as thermal switches, do have significant on-power though, so it is included for completeness.

Comparing (3.7) and (3.3) shows several important distinctions between the behavior of a transistor switched circuit versus a NEMS switched one:

- In NEMS-gated circuits, the average power consumption approaches 0 as R_T approaches 0.
- The supply voltage that minimizes energy of NEMS-gated circuits is no longer dependent on the target throughput, and can be determined separately.
- The most suitable architecture, regardless of the target throughput, is the one that has the least E_{MIN}. Since the architecture is running at its maximum throughput, this tends to be the architecture with the lowest switching energy. This contrasts to transistor power-gated applications, where the most suitable architecture highly depends on the target throughput.

3.4 Implementation

Since NEMS switches have an actuation voltage that is much higher than the typical voltage of a CMOS circuit, a system is needed that is able to generate the high voltages and distribute them to the switches. This section details that system, as well as the various FFT architectures that are implemented for the architectural exploration.

3.4.1 NEMS Power-Gating System Overview

Figure 3.3.A gives a system-level overview of the NEMS-based power-gating scheme. For clarity, each of the major components are labeled with numbers. The charge pump (1) takes a voltage of 1 V (which is generally available on ultra-low-voltage chips for I/O) and produces the activation voltage necessary to actuate the switches. The charge pump simulated in this study is a bootstrap charge pump [24], which can be compactly implemented on-die with

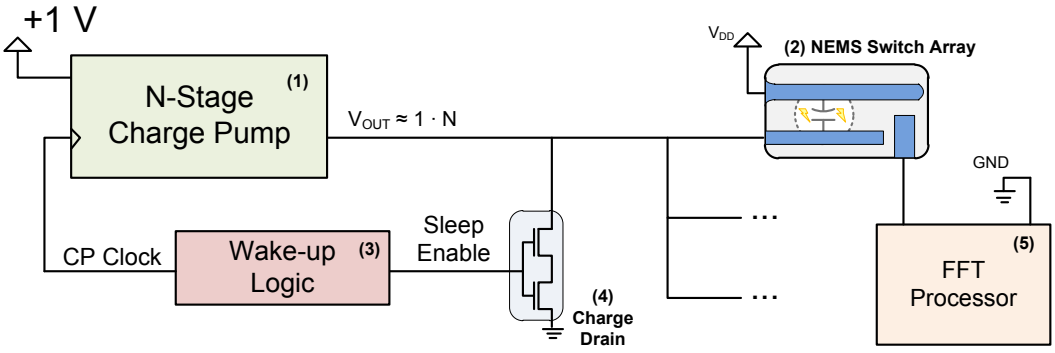

Figure 3.3: System diagram of a NEMS power-gating scheme.

MOSFET capacitors [35] and (ideally) produces an output voltage $V_{OUT} = V_{DD} * N$, where N is the number of stages in the pump. The load of the charge pump is the combined capacitance of the NEMS switch array (2). When the wake-up logic (3) decides to switch on the FFT processor (5), it activates the charge pump clock. The charge pump charges the gate of the NEMS switches to the activation voltage of the switches, which actuates them and connects the FFT processor to V_{DD}. Once the wake-up logic determines the FFT processor is no longer needed, it opens up a charge drain (4), which drains the charge from the NEMS switches, breaking the FFT processor's connection to V_{DD}.

Figure 3.4.A presents a schematic representation of the simulated boot-strap charge pump. Two MOSFETs per stage are used as diodes to prevent the charge from flowing back to the V_{DD} source, while a clock signal ϕ effectively acts as the pumping mechanism. The clock is connected to 250 fF capacitors, which were simulated using MOSFET gates. Figure 3.4.B presents SPICE simulation results of the output voltage of a 16-stage charge pump with respect to time ($V_{DD} = 1$). The non-idealities and charge pump losses are clear from this graph since the 16 stage pump only charges to around 15 V. Also shown for comparison is the turn-on voltage of a NEMS switch with a 90 nm drain gap (see Table 3.1).

The leakage current of NEMS switches has been shown to be in the femto-amp range [6]. Likewise, the speed of the charge drain is not critical, so it can be stacked and sized such that leakage through it is also in this range. This implies the NEMS switches can hold their charge for a very long time, so once the charge pump charges the NEMS switches, it can

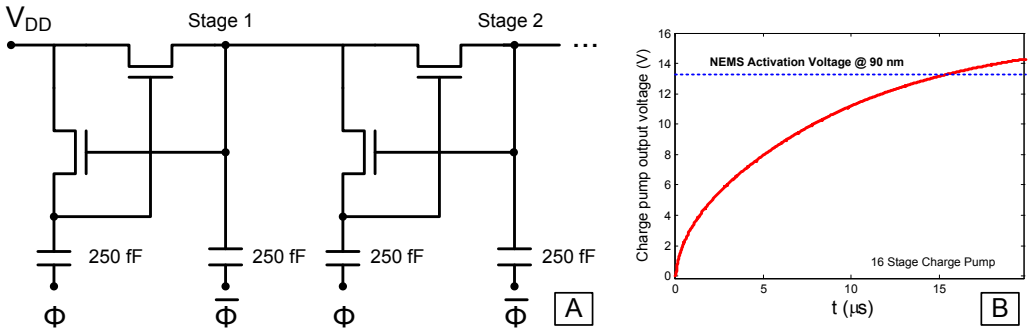

Figure 3.4: A. Schematic of the simulated bootstrap charge pump. B. Simulated output voltage with respect to time for a 16 stage charge pump; $\phi = 20$MHz.

be shut-off for the duration of the processing. Therefore, effectively, the NEMS switches require no power consumption to keep them on; only a fixed amount of energy is needed to charge them to the actuation voltage. The experiments in this paper measured the activation energy and activation time through charge pump simulations and the impact on the overall energy consumed by processing the FFT was determined.

An important consideration with any power-gating system is what has to be left on while the unit is asleep. As far as wake-up periphery, this is highly application-dependent and can be as simple as a picowatt-range timer [13]. A requirement for any NEMS-based power-gating system, however, is that the charge pump is able to function while the NEMS switches are off. This necessitates putting the charge pump on a power network that is not switched with NEMS switches and the leakage will be non-zero leakage when idle. Due to a relative lack of leakage paths, the charge pump exhibits low leakage when idle, and additionally, it can be power-gated using conventional transistors and activated using the wake-up periphery. As part of the experimental methodology, the leakage of a power-gated charge pump will be measured.

Table 3.1: Physical characteristics of the switches used in this study.

Target CMOS Tech.	**Beam Length (μm)**	**Beam Width (nm)**	**Drain Gap (nm)**	**Activation Voltage**	**Switching Speed (ns)**
130 nm	11	100	130	19.5	98
90 nm	10	100	90	13.1	81
65 nm	10	100	70	9.0	81
45 nm	10	100	50	5.9	81
32 nm	10	100	30	3.2	81

3.4.2 NEMS Switch Physical Parameters

In [6], an analysis is provided of the effect of switch dimensions on activation voltage and switching times. This is done through both simulation and fabrication of switches with a wide variety of dimensions. The mechanical delay of the NEMS switches fall in the nanosecond range, so the boot-up time of the FFT processor is dominated by the charge pump. A lower activation voltage means less work for the charge pump, so the switches with the lowest activation voltages were chosen from the set in [6], which also happen to be the slowest switches. Architectures with transistor channel lengths ranging from 130 nm to 32 nm were tested in this study. In a CMOS process, this channel length is usually the smallest dimension that is achievable by the photolithography, which means there can be no structures on the NEMS switch that are smaller than the channel length. The drain gap is the critical dimension of the NEMS switches (i.e. it is desirable to minimize this dimension), so it will be matched to the channel length of their corresponding CMOS technology. An overview of the physical characteristics of the chosen NEMS switches is given in Table 3.1, listed by their corresponding CMOS technology. It should be noted that there was no data for the 130 nm and 90 nm switch, so the parameters for those switches were extrapolated using analytical equations from [6].

The contact resistance can be considered a tunable parameter of the NEMS switch. Lower contact resistance means less NEMS switches are needed in the power-gating array, which

has a direct impact on the area overhead. To quantify this, a range of contact resistances was examined and the resulting area overhead was determined and compared to the area overhead of transistor-based power-gating.

3.4.3 FFT Architectures

One of the topics studied is the effect of transistor-based and NEMS-based power-gating on architecture choice. To that end, four FFT architectures that vary in complexity and speed were simulated. They each perform a 1024-point radix-2 FFT with 32-bit complex numbers. The low- and medium-complexity implementations use a single real valued 16-bit ALU with a multiplier and an adder, a register bank, and a very simple microprogram to implement the FFT. The low-complexity design uses a 16-bit add/shift multiplier that takes 16 clock cycles, while the medium-complexity design uses a high speed Booth multiplier that takes a single clock cycle. The high-complexity implementation takes advantage of the repetitive nature of the signal flow of an FFT, and contains 9 adders and 3 Booth multipliers. Finally, a parallel architecture with 16 high-complexity cores is investigated. All of the designs are optimally pipelined with respect to energy per FFT at maximum frequency [28]. The designs require a memory bank to hold the 1024 input points, and the same memory bank also holds intermediate values and the final result. A ROM bank is also required to store the constant coefficients (sometimes called twiddle factors) used in the FFT. Data retention is an important concept with power-gating, because a powered down memory bank loses its data. As is the case with a lot of signal processing applications, the raw data does not need to be kept after it is processed. Therefore, data retention is not necessary. More general purpose architectures, on the other hand, may require a data retention plan.

Figure 3.5 presents the total energy per FFT of the four different designs with respect to supply voltage (the designs are running at their respective maximum clock rate) for the 130 nm technology node, using the methodology in Section 3.5. As the complexity of the designs increase, the energy per FFT decreases, mainly due to the fact that the smaller designs

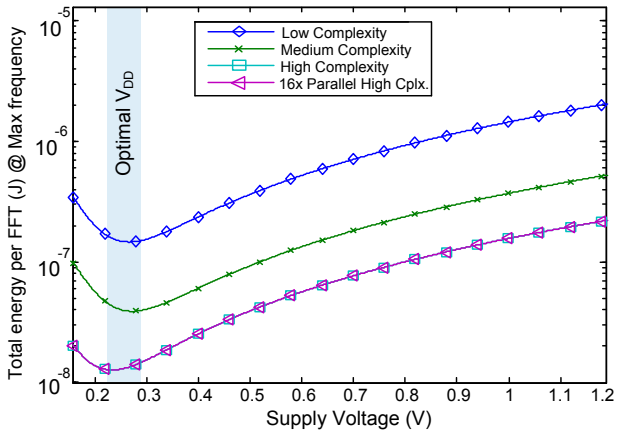

Figure 3.5: Total energy of the 4 FFT architectures (at their respective max. frequencies) with respect to V_{DD}.

must save intermediate operands in registers and have a lower activity factor. However, the parallel design is not more energy efficient than its scalar counterpart, because parallelization in general does not decrease the energy per task [28].

3.5 Methodology

This chapter presents results of experiments that compare NEMS- and transistor-based power-gating across a wide variety of metrics. An area analysis was carried out which compared area overhead required for power-gating. The energy and time required to actuate the NEMS switches were also determined. An architectural exploration was carried out, where FFT architectures of various complexities were examined with respect to switching energy efficiency and sleep-mode power consumption. Finally, through a technology scaling exploration, it was determined how effective the proposed power-gating structures are as feature sizes decrease. The specific methodologies for each exploration follow:

3.5.1 NEMS Switch Simulation

The simulation of NEMS switches, and their older micron-scale MEMS counterpart, has received a lot of attention in the literature and is important for understanding how various parameters, including material and geometry selection, affect the switching speed, turn-on voltage, and reliability. Considering NEMS switches are actually mechanical devices, doing a mechanical co-simulation with an electronic simulator such as SPICE would be difficult and time-consuming. For these reason, we chose instead to treat NEMS switches as a simple switch in our simulator. The simulated switch has an infinite off-resistance, a specific turn-on gate voltage, a turn-on delay once that gate voltage is reached, and a finite contact resistance once on. Additionally, plate capacitors are used to model the gate-beam capacitance as well as the various parasitic capacitances that are evident in Figure 3.1. This simple switch definition allows us to abstract the many switch parameters into what is important for digital logic designers, and is also flexible enough to handle a wide range of switch types.

As for switch material selection, the switch from [6] was simulated because it contained the most comprehensive data on the effect of switch geometry versus speed and turn-on voltage. Both the structural and contact material of this switch was strained Ruthenium. Other structural materials have been reported, including polysilicon switches that demonstrate high reliability [25], but comprehensive data in the literature is lacking. The contact resistance is an important parameter for power-gating, but is not well researched due to its relative unimportance with NEMS-based digital logic. Additionally, the structural portion of any switch can be coated with metal to improve the contact resistance, e.g. the polysilicon switch from [25] uses a platinum coating. Since contact resistance is such a highly controllable parameter, we abstracted the material and contact design into a simple range of potential contact resistances based on previously reported values.

3.5.2 NEMS Area Analysis

With NEMS-based power-gating, the two major determinants of area overhead are the contact resistance of the NEMS switch, and the magnitude of the performance drop that is acceptable. For the area overhead experiments, contact resistances ranging from 50 to 6000 Ω were considered. Two performance drops were considered as well: 10% (corresponding to $K_D = 1.11$, as shown in Equation 3.3) and 90% ($K_D = 10$). SPICE simulations were carried out to determine, for a given contact resistance, how many NEMS switches are needed to limit the delay increase to these two values. Then, using the switch dimensions in Table 3.1 and the layout in Figure 3.1.B, the total area overhead was determined. This was done for the 130 nm technology and the High Complexity FFT architecture with a supply voltage of 0.5 V.

As a point of comparison, the area overhead of transistor-based power-gating was also determined. The number of transistor headers (Regular V_{TH}, channel width = 1.5 μm, channel length = 130 nm) necessary to limit the delay increase to $K_D = 10$ and $K_D = 1.11$ was measured in SPICE, and this was multiplied by the area of the header (determined by a standard cell layout in Virtuoso).

3.5.3 NEMS Energy Analysis

The primary sources of NEMS actuation energy are the charging of the gate capacitance of the NEMS switches, and energy loss resulting from the non-ideal nature of the charge pump. To quantify this energy, as well as the additional actuation delay caused by the charge pump charging up the output load, SPICE simulations of the system in Figure 3.3 were carried out for the switches listed in Table 3.1. The gate capacitance was calculated using a parallel plate model and, as an example, was found to be 1.60 fF for the 130 nm switch. For each activation voltage, the number of charge pump stages needed, the time required for the output to reach the activation voltage, and the energy consumption of the

charge pump and NEMS switches were measured. The output capacitive load of the charge pump, which is the combined gate capacitance of the NEMS switches in the power-gating switch array, impacts the boot-up time and activation energy. To quantify the effect of the number of switches in the array on activation time and energy, two loads were tested in the simulations. The first load is the combined gate capacitance of 17 switches, which is the number of 800 Ω NEMS switches required to achieve $K_D = 10$ (90% drop in performance) with the 130 nm High Complexity FFT architecture at 0.5 V. The second load is the gate capacitance of 170 switches, which results in a $K_D = 1.11$ (10% performance drop).

3.5.4 FFT Architectural Exploration

An industry-level 130 nm Regular V_{TH} standard cell library was used to synthesize the four FFT architectures. For memory, custom SRAM bit-cells that can operate down to 450 mV [5] were integrated into CACTI [22]. For the transistor-gated circuits, an exhaustive search was used to find the optimal supply voltage and header size in order to minimize power consumption for each target throughput, which ranged from one FFT per μs to one FFT per day. Supply voltages ranging from 1.2 V to 150 mV were tested. To determine the optimal header size, the methodology from [32] was used, whereby K_L and K_D are determined for various relative widths, ranging from one down to the minimum functional size. This process was repeated for each of the four FFT architectures. For the NEMS-gated circuits, the same procedure was applied, with the exceptions that the exhaustive search was only done for V_{DD}, the activation energy was added in, off-state leakage was assumed to be zero, and a 10% performance drop due to the contact resistance was assumed.

3.5.5 Technology Scaling Exploration

The technology scaling exploration examined technology nodes with channel lengths of 90 nm, 65 nm, and 32 nm. For brevity, only the high complexity architecture was examined for

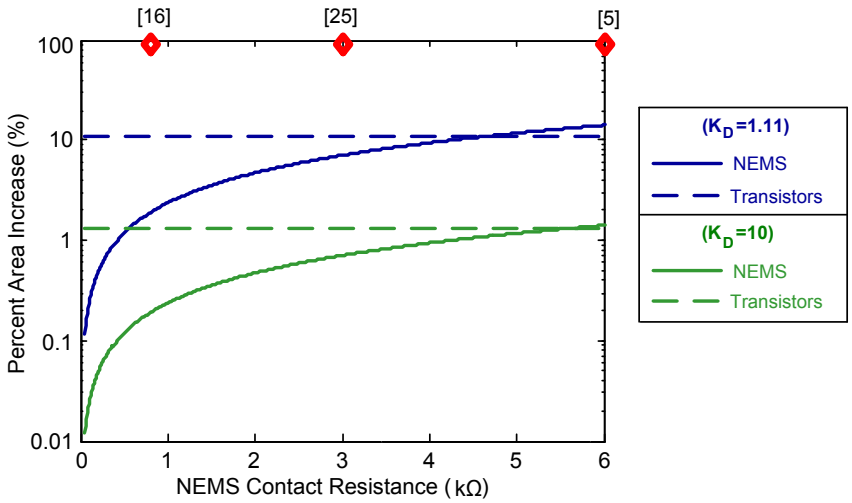

Figure 3.6: Percent area increase with respect to the NEMS contact resistance for two target performance drops. Shown for comparison is the area increase with a transistor-based power-gating implementation.

this exploration. The power and timing were determined using scaling based on predictive technology model [37] SPICE simulations. The power and timing of the memory banks were gathered from [22]. The use of custom low-voltage bitcells and the transistor-gating and NEMS-gating methodology is identical to the architectural exploration.

3.6 Results

3.6.1 NEMS Area Analysis

Figure 3.6 gives the power-gating area overhead with respect to the contact resistance of the NEMS switches for the 130 nm High Complexity FFT architecture with a supply voltage of 0.5 V. The baseline area, of which percent increase is given, is 40,100 μm^2. Both a 10% ($K_D = 1.11$) and 90% ($K_D = 10$) target drop in performance is considered, and the area overhead of a transistor power-gating implementation is shown for comparison. At the top of the graph, the contact resistance of three NEMS prototypes is marked on the x-axis.

The contact resistance of the NEMS switch directly determines how many switches are needed to achieve the target performance drop. It can be seen that as the contact resistance falls, the area overhead reduces dramatically to the point of being negligible. It would take a contact resistance of above 5 $k\Omega$ for the area overhead to equal or exceed the overhead of transistor power-gates, and early NEMS prototypes [16, 25] have already achieved lower resistances. In summary, transistor power-gates led to an area overhead of up to 10%, depending on the target performance drop, while current NEMS prototypes would only led to an area increase of a few percentage points, which is expected to improve as NEMS technology improves.

3.6.2 NEMS Energy Analysis

As was noted earlier, the sources of NEMS actuation energy are the charging of the gate capacitance of the NEMS switches, and energy loss resulting from the non-ideal nature of the charge pump. Table 4.2 gives the actuation time (T_{BOOT}) and actuation energy (E_{BOOT}) for the various NEMS switches (130 nm High Complexity FFT processor with 0.5 V supply and 800 Ω contact resistance). Considering the 130 nm FFT architecture consumes 3.92 μJ, it can be seen that even with a 19.5 V activation voltage, the energy overhead of the charge pump is negligible. Additionally, a performance drop of 10% requires ten times as many switches as 90%, but leads to only a small increase in activation time and energy, suggesting most of the delay and energy consumption comes from the non-ideal efficiency of the charge pump. It is clear that for applications with a similar processing time and energy consumption of this FFT application, the primary design focus of the NEMS switch should be reducing switch area and contact resistance, with the activation voltage and switching time being the secondary focus. Considering the charge pump cannot be power-gated with NEMS switches, its own leakage power is an important consideration. The off-state leakage power of the largest charge pump was found to be 50.2 pW when power-gated off with transistors. The source of leakage was mostly in the peripheral control structures and is

relatively independent of the number of stages.

Table 3.2: Time and energy required to active NEMS switch bank.

CMOS Technology	NEMS Actuation Voltage (V)	Target Performance Drop 90% T_{BOOT}	90% E_{BOOT}	10% T_{BOOT}	10% E_{BOOT}
130 nm	19.5	38.1 μs	2.97 nJ	40.6 μs	3.16 nJ
90 nm	13.1	15.6 μs	0.945 nJ	17.0 μs	1.03 nJ
65 nm	9.0	5.17 μs	225 pJ	5.97 μs	260 pJ
45 nm	5.9	0.966 μs	28.0 pJ	1.22 μs	35.5 pJ
32 nm	3.2	92.3 ns	2.46 pJ	147 ns	3.93 pJ

3.6.3 Architectural Exploration Results

Transistor-Based Power-Gating

Figure 3.7 presents the results of the architectural exploration for transistor-based power-gating . For a given target throughput (horizontal axis), the best combination of supply voltage (chart C) and relative header width (chart B) that yields the minimum average power consumption (chart A) is shown. As the target throughput reduces, due to the relaxed performance requirements, the header width can be reduced, which reduces the sleep-mode power but also increases the performance drop. As was mentioned in Section 3, if the header is too small, then the voltage drop across the header would render the circuit inoperable. Figure 3.7.B shows the fundamental limit to the relative size of the header. For the 130 nm technology, the optimal relative header width approaches $5 * 10^{-2}$ with an optimal supply voltage of around 200 mV.

It can be seen that with target throughputs higher than an FFT every second, the parallel architecture is the most power efficient. The reason is that the parallel architecture is much faster than the others, so the supply voltage can be set lower, thereby reducing power [28].

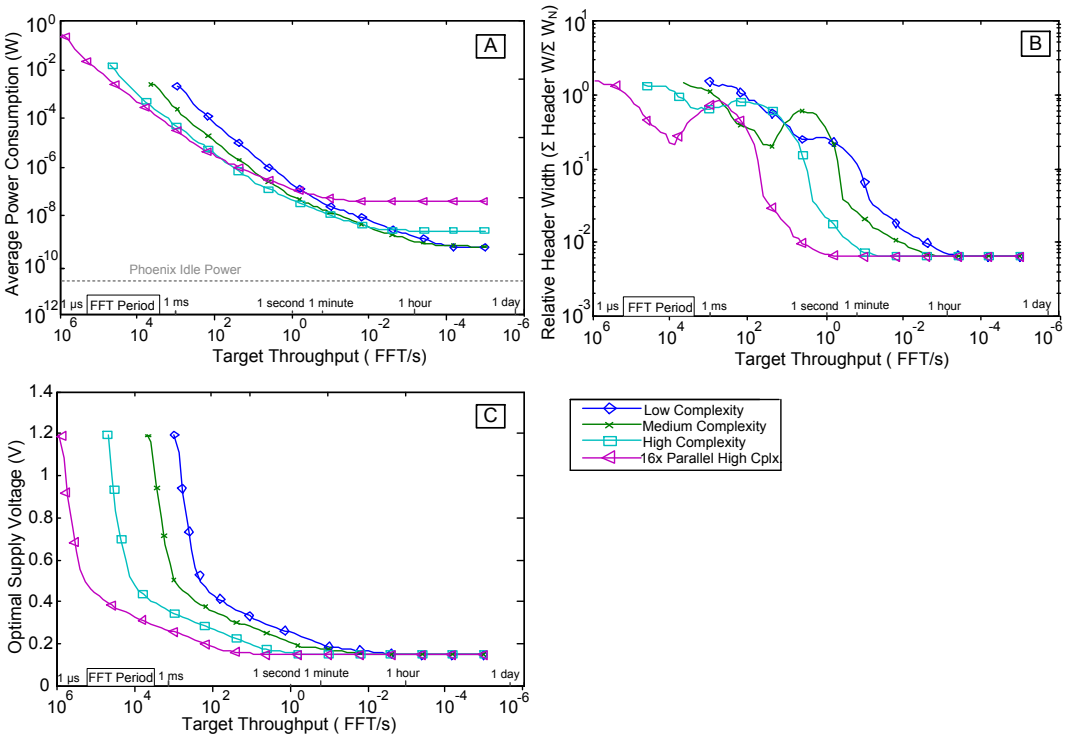

Figure 3.7: A: Average power consumption of transistor-power-gated FFT architectures w.r.t. target throughput. B: Optimal relative header width w.r.t. target throughput. C: Optimal supply voltage w.r.t. target throughput.

At lower throughputs, where the header size starts to reach the above mentioned limit, the average power consumption approaches P_L/K_L. In this region the least complex architectures become the most power efficient. *This implies that even in the presence of transistor switches, the target throughput is an important consideration when selecting an architecture.*

It is important to note that, since the header width varies, each data point on the graph represents a different physical circuit. Once the chip is manufactured, the header size is unchangeable, which leads to a non-flexible architecture. If a chip must operate for a period of time at 1 FFT per minute, then operate at 1 FFT per millisecond, the power consumption would be sub-optimal.

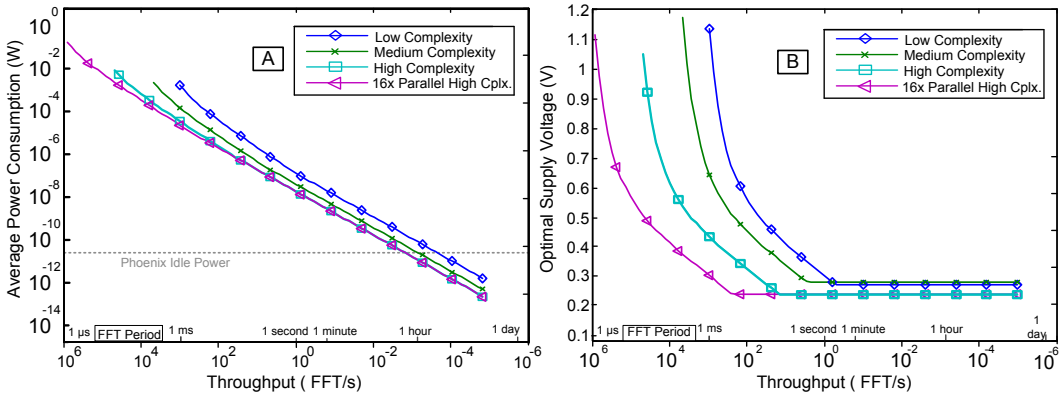

Figure 3.8: A: Average power consumption of NEMS-power-gated FFT architectures, w.r.t. target throughput. B: Optimal supply voltage, w.r.t. target throughput.

NEMS-Based Power-Gating

Figure 3.8.A presents the average power consumption of the four FFT architectures with NEMS-based power-gating. With the exception of very high throughputs, where the supply voltage must be raised to meet the throughput requirement, the power consumption forms a linear relationship with target throughput, as predicted by the mathematical analysis, namely Equation (3.7). It can be seen that, with a very small exception in the high throughput region, the high complexity and parallel architecture are the clear winners in terms of power consumption. This means one can sacrifice area and gain a significant drop in both energy per FFT and processing time, no matter what the target throughput is. This is a significant improvement over the transistor-gated architectures, where the high complexity architectures are only the most energy efficient at high target throughputs.

The power consumption shown in these figures is for the FFT processor (and power-gating system) only. When the FFT processor is in a sleep-mode, the wake-up logic would still contribute to the overall power consumption of the system. Shown in Figure 3.8.A is a dashed line representing the 35 pW sleep-mode power consumption of the Phoenix processor [13]. While in a sleep-mode, the Phoenix still has half of the data memory and a wakeup-timer powered on. If the FFT processing system were to have a similar sleep-mode system, the

power consumption of all four architectures would eventually approach and converge on that line rather than cross it. The dashed line is also shown in the transistor power-gating graph, and due to the higher sleep-mode power, the architectures converge on values that are at least 30 times larger than the Phoenix sleep-mode power.

3.6.4 Technology Scaling Exploration Results

Figure 3.9 presents energy per FFT with respect to supply voltage and technology node for the NEMS-gated High Complexity FFT processor and two transistor-gated High Complexity FFT processors. The first transistor-gated FFT processor has header widths that optimize energy for a target throughput of 1 FFT per second, while the second is optimized for 1 FFT per 30 seconds. Like the previous transistor-gated graphs, each data point for both supply voltage and technology node has optimum header widths for that specific point, so the header widths are not the same across each curve.

It is clear from Figure 3.9.A that for NEMS-gated architectures, as the channel length decreases, energy per FFT decreases substantially. The same cannot be said for transistor-gated architectures which suffer from the increasing leakage of lower technology nodes. With 1 FFT per second, the throughput is high enough that the newer technologies deliver lower energy, but the energy reduction is not nearly as high as the NEMS-gated FFT processor. With 1 FFT per 30 seconds, shown in Figure 3.9.C, going from 90 nm to 65 nm leads to a drop in energy, while going from 65 nm to 32 nm leads to no drop in energy. It is clear from these graphs that, in terms of energy efficiency, NEMS-gated devices can reap much larger benefits from technology scaling.

Figure 3.10 breaks down the total energy of the optimal point of each energy curve in Figure 3.9 (indicated with an arrow) into its individual components, namely switching, leakage and off-state leakage energy. There is no off-state leakage energy with the NEMS systems due to the infinite off resistance of the switches. Additionally, the leakage energy of the NEMS-gated processor is negligible compared to the switching energy, which is typical of

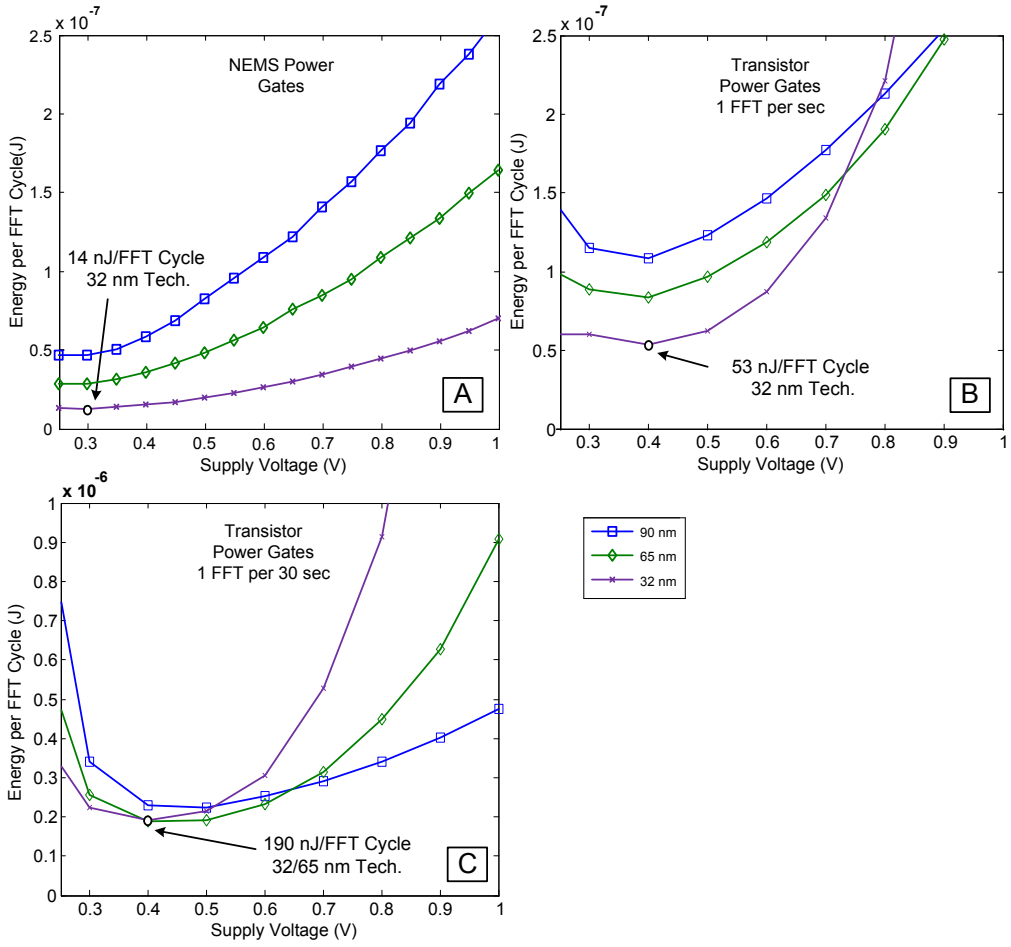

Figure 3.9: Energy per cycle with respect to supply voltage and technology node for A: NEMS-based power-gating with 1 FFT per cycle, B: Transistor-based power-gating for 1 FFT per 1 second cycle, and C: Transistor-based power-gating for 1 FFT per 30 second cycle. It should be noted that, due to the lack of off-state leakage, the NEMS energy would be the same regardless of cycle time (see Equation 3.7).

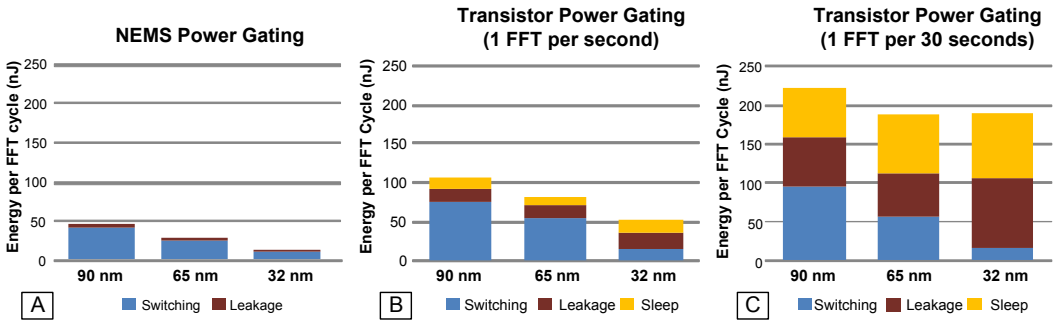

Figure 3.10: Breakdown of the various energy components (optimal supply voltage from Figure 3.9) with respect to technology node for A: NEMS-based power-gating, B: Transistor-based power-gating with 1 FFT per 1 second cycle, and C: Transistor-based power-gating with 1 FFT per 30 second cycle.

a circuit with a high activity factor. The leakage energy of the transistor-gated processors, on the other hand, makes up a significant portion of the total energy. This is because the very small headers are what optimizes total energy, and these headers greatly slow down the circuit. This lowers the activity factor of the circuit to the point that leakage energy becomes a major component of total energy. Since leakage currents increase with lower technology nodes, going from 65 nm to 32 nm leads to no decrease in the total energy consumption for a cycle time of 30 seconds.

In summary, lower technology nodes have traditionally been avoided for ultra-low power design due to the transistor leakage currents, which increase exponentially with a decreasing channel length and oxide thickness. Newer technologies consume less switching energy, however, because of transistor capacitances. With NEMS-based power-gating, this benefit can be leveraged for an ultra-low power application without concern for the higher leakage. It was shown that going from 90 nm to 32 nm led to around a 70% drop in energy for a NEMS-gated FFT processor. For a transistor-gated design, on the other hand, with a cycle time of an FFT every 30 seconds, going from 90 nm to 65 nm only led to a 15% drop in energy, while going from 65 nm to 32 nm led to almost no drop in energy.

3.7 Conclusion

In this chapter, an emerging technology – the CMOS-Compatible NEMS switch – was proposed as an alternate power-gating structure to transistors. It was shown that the infinite off-resistance of a NEMS switch completely eliminates off-state leakage, which can lead to a large drop in power consumption for devices that are rarely active. We used an FFT processor as a benchmark to explore the effectiveness of transistor and NEMS power-gating with respect to target throughput, architecture selection, technology, and supply voltage. We demonstrated that the sizing of transistor power-gates must be carefully tuned to the target throughput of the application. This leads to an inflexible design that is sub-optimal at other throughputs. In addition, headers that are too small can cause a voltage drop large enough to render the circuit non-operational, which means transistor-based power-gating can be ineffective at extremely low throughputs. With NEMS-based power-gating, a single architecture was shown to be optimal for all target throughputs. This means the processor is much more flexible to changing throughput requirements. The infinite off-resistance of NEMS switches leads to large reductions in average power consumption for architectures with long sleep periods. As an example, a NEMS-power-gated architecture performing an FFT every hour consumes 30 times less power than a transistor-power-gated architecture.

The downside of NEMS switches is their high activation voltage, which requires a charge pump that contributes to energy and delay overhead. The activation energy of the NEMS power-gate switch array in a 130 nm technology is in the nanojoule range, the activation time is under 40 μs, and the on-power is effectively zero. The primary source of activation delay and energy is from booting up the charge pump and this overhead falls dramatically with lower technologies. As for area overhead, a power-gating implementation that uses current NEMS prototypes would lead to a 36-83% reduction in area compared to transistor power-gates. Further improvements to the contact resistance of the NEMS switches would yield a much larger reduction.

The targeted applications of this chapter were ones with long sleep periods, such as periodic sensor processors. As such, the granularity of the power-gating was coarse-grained in nature, which meant the mechanical delay and relatively large energy-overhead of NEMS switches were masked by the extremely large reduction in leakage energy. With modern microprocessors, especially those operating in the ULV region, leakage can be severe even when a microprocessor is actively processing. In cases like these, a more fine-grained approach is needed, where individual functional units of the processor are shut-off when not in use and quickly turned on when requested. In a system such as this, the mechanical delay and energy overhead of NEMS switches become more apparent, which motivates the investigation in the next chapter: comparison of transistors versus NEMS as power-gating structures for a microprocessor with fine-grained functional unit power-gating.

Chapter 4

Fine-Grained Power-Gating: Transistors vs. NEMS

4.1 Introduction

Considering that eliminating leakage leads to low optimum voltages and an overall reduction in energy consumption, it is desirable to move towards fine-grained power-gating, which is more aggressive in targeting idle circuitry. As Chapter 2 detailed, granularity can be broken up into two categories: temporal granularity determines how short the targeted idle periods can be, while regional granularity determines how many separately controllable power-gating regions there are. The studies in the previous chapter only looked at coarse-grained power-gating in the sense that only long periods were targeted, and only the entire FFT processor was capable of being shut down. This level of granularity may be appropriate for hardware as specific as an FFT processor, but for general purpose processing, it is insufficient.

Consider a processor with an integer and floating point unit that is calculating the hamming distance of a series of integers. During this time, only the integer adder and shifter are being used, while the other functional units simply consume leakage energy. If the cumulative

activity factor of the entire processor were to be determined, it would be very low, favoring a high supply voltage (V_{DD}) and high threshold voltage (V_{TH}). On the other hand, if a floating point benchmark was being processed and all functional units were being used, the collective activity factor would be high, favoring a low V_{DD} and V_{TH}. Since only one V_{DD} and V_{TH} can be selected, it would have to be somewhere in the middle, meaning the energy efficiency of both benchmarks suffers.

By shutting down the unused functional units during the Hamming benchmark, however, the same high collective activity factor could be attained for both cases, and the low V_{DD} and V_{TH} would be favored across the board. This reduces energy in multiple ways: shutting down the idle units reduces leakage energy, while lowering V_{DD} reduces switching energy. The implication is that, in order to lower the V_{DD} and V_{TH}, while reducing both leakage and switching energy, a high degree of regional granularity is needed that allows individual functional units to be shut down. Additionally, considering that some benchmarks use certain functional units only part of the time, and that an OS may switch between concurrent benchmarks, a relatively high degree of temporal granularity is also needed, in the form of a scheduler that can quickly turn functional units on and off.

To quantify these statements, this chapter explores a microprocessor with fine-grained functional unit power-gating and determines what effect the higher level of granularity has on overall energy efficiency. Considering that the previous chapter advocated NEMS as power-gating structures, NEMS are explored alongside transistors, and it is determined if the relatively high activation energy, mechanical delay, and limited life-time limit their effectiveness when used for fine-grained applications. To study the effect of granularity and run-time leakage-control on the optimal V_{DD} and V_{TH}, a detailed energy model is presented, which is integrated with a cycle-accurate CPU simulator. Finally, a very important aspect when considering regional granularity is the complexity of the power-gating scheduler: if the scheduler is overly complicated, then the energy and hardware overhead of adding additional power-gating regions would be prohibitively large. To this end, a simple power-gating policy is presented that simply needs a global counter, and a flip-flop for each power-gating region.

This policy is compared to an ideal "oracle" power-gating policy that has perfect knowledge of the future. Our key findings are as follows:

- Functional unit power-gating pushes the optimum supply voltage lower by 100-250 mV, which positively benefits all benchmarks. Even a highly active floating point FFT benchmark that rarely shut any units off sees a 20.7% drop in total functional unit energy.
- With an ideal oracle-based scheduler, NEMS-based functional unit power-gating yields a 29.5% average reduction in total functional unit energy with respect to no functional unit power-gating, while an ideal transistor-based system yields a 23.5% reduction.
- With the more realistic hardware-based Flag Policy, NEMS power-gating yields a 28.9% drop in energy, while transistor power-gating yields a 23.0% drop in energy. These results suggest that the hardware-based policy achieves results that are very close to the ideal policy, despite being very simple in nature.
- When scaling back the aggressiveness and limiting the NEMS Flag Policy to an on/off rate that allows for multiple years of constant operation before NEMS failure, the total functional unit energy reduction only falls from 29.5% to 25.1%.
- A NEMS/transistor hybrid policy is proposed, which adds transistors to the system for short-term power-gating. This brings the average energy savings up to 31.7%. For the basicmath benchmark, the energy savings jumps from 5.0% to 25.8%.

The rest of this chapter is organized as follows: Section 4.2 presents an energy model for functional units that are power-gated using either transistors or NEMS. It also presents the Flag Policy, which is a simple hardware-only power-gating policy for functional units. Section 4.4 presents the NEMS simulation environment, which is an expanded version of the NEMS charge-pump system presented in the previous chapter. Section 4.5 presents the methodology of the experiments in this paper, while Section 4.6 presents the results of the experiments.

4.2 Functional Unit Energy Model in the Presence of Power-Gating

A major difference between traditional circuit design and low-power/low-voltage design is that the nominal V_{DD} and V_{TH} supplied with a process is typically not well suited for effective low-power design. This shifts the burden to the designer to decide on a V_{DD} and V_{TH} that balances performance, switching energy, and leakage energy. In this section, we provide an optimization framework for measuring the energy and optimizing the V_{DD} and V_{TH} of the functional units of a CPU in the presence of both NEMS- and transistor-based functional-unit level power-gating.

For all of the modeling herein, the Enz-Krummenacher-Vittoz (EKV) model is used, which allows for seamless modeling of power and performance for V_{DD}s both above and below the V_{TH} of a transistor [10, 27]. With the EKV model, a *process* is characterized by the Drain-Induced Barrier Lowering coefficient, λ_D, which models how V_{DD} affects V_{TH}, the subthreshold slope, n, and the thermal voltage, V_T. A *circuit* is characterized by the total specific current of the circuit, I_S, the average activity factor, α, switching capacitance, C_S, and critical delay coefficient, $\frac{KC_{CP}}{I_{S,cp}}$, which is the capacitance of the critical path times a fitting parameter, divided by the average I_S of the critical path. The specific current of a single transistor is modeled as follows:

$$I_S = 2nC_{ox}\mu\frac{W}{L}{\phi_t}^2 \tag{4.1}$$

According to the EKV model, given a target delay, t_d, and V_{DD}, the V_{TH} needed to meet the critical delay is [1]

$$\begin{aligned} &V_{th,opt}(V_{DD}, W_H) = \\ &(1+\lambda_D)V_{DD} - 2nV_T log(e^{\sqrt{\frac{KC_{CP}V_{DD}K_D(W_H)}{I_{S,cp}t_d}}} - 1) \end{aligned} \tag{4.2}$$

[1] For al of the equations, parameters on the left side in parentheses are those that are subject to optimization.

Included in (4.2) is a delay coefficient, $K_D(W_H)$. This accounts for the performance reduction due to the resistance across the power-gating structure. For transistors, it is dependent on the relative width of the headers, W_H, which is the cumulative width of the headers divided by the cumulative width of the entire circuit. A smaller W_H yields a larger K_D, which means a higher V_{DD} and V_{TH} is needed to meet the critical delay. For NEMS switches, it is dependent on the contact resistance.

The total leakage energy and switching energy is given by

$$E_L(V_{DD}) = t_d V_{DD} I_S e^{\frac{\lambda_D V_{DD} - V_{th,opt}(V_{DD})}{nV_T}} \tag{4.3}$$

$$E_S(V_{DD}) = \frac{1}{2}\alpha C_S {V_{DD}}^2 \tag{4.4}$$

Equations (4.2-4.4) allow for optimization of the total energy of a circuit given a critical delay. The addition of power-gating requires further modeling, due to the energy overhead needed to turn the power networks on and off.

Power-Gating Overhead:

There is an energy overhead with powering a functional unit on and off, so to account for this, the power-gating energy model from [15] will be used. When a unit is first turned off, the power network does not instantly go to zero volts, but instead slowly drains away at a rate that is dependent on the leakage current. This means that the amount of energy needed to recharge the supply networks depends on how many cycles the unit has been off. According to [15], the number of cycles before a unit is discharged is

$$n_{dis} = \frac{C_B n V_T V_{DD}}{E_L \lambda_D} \tag{4.5}$$

where C_B, or boot capacitance, consists of the gate, diffusion, and metal capacitances of a circuit that must be recharged when a unit is turned on. Given V_{DD} and M, which is the number of cycles a unit has been off, the boot-up energy is

$$E_B(M, V_{DD}) = \begin{cases} NE_L(1 - \frac{\lambda_D M E_L}{2nV_T C_B V_{DD}}) & N < n_{dis} \\ \frac{1}{2} C_B {V_{DD}}^2 & N \geq n_{dis} \end{cases} \tag{4.6}$$

Equation (4.7) gives the overhead of activating the header network, given W_H:

$$E_{OH,T}(V_{DD}, W_H) = W_H C_S {V_{DD}}^2 \tag{4.7}$$

Using (4.6) and (4.7), the break-even cycle count, i.e. the number of cycles that a unit must be off for the leakage savings to justify the power-gating overhead, for transistor-power-gated units is

$$n_{BE,T}(V_{DD}, W_H) = \frac{\sqrt{2nV_T W_H C_B {V_{DD}}^3 C_S}}{\sqrt{\lambda_D} E_L} \tag{4.8}$$

Equations (4.2-4.8) give the equations necessary to construct an energy model of a functional unit that is power-gated using transistors. For power-gating with NEMS switches, these equations need to be slightly modified to account for the different switching characteristics. While NEMS switches do have a measurable capacitance, the primary source of energy overhead comes from the charge pump that generates the high activation voltages. This is difficult to quantify analytically, so an activation energy value measured from simulations, E_A, will be used.

$$E_{OH,N}(NS) = E_A(NS) \tag{4.9}$$

This is dependent on the particular NEMS switch in use, NS. Likewise, the break even cycle count needs to be slightly modified. NEMS switches have an activation delay that spans multiple clock cycles, and the break even cycle count cannot be lower than the delay of the switch, otherwise the switch would not be able to turn on fast enough. This is reflected in (4.10), which also replaces (4.7) with (4.9):

$$\begin{aligned} &n_{BE,N}(V_{DD}, NS) = \\ &max(\frac{\sqrt{2nV_T C_B V_{DD} E_A(NS)}}{\sqrt{\lambda_D} E_L}, Delay(NS)) \end{aligned} \tag{4.10}$$

This section provided an energy model for an individual functional unit that is power-gated using either transistors or NEMS switches. To get the total functional unit energy for a given benchmark, a cycle-accurate trace of the functional unit usage is needed, along with a model of the power-gating policy in place. In the next section, five such policies are presented.

4.3 Functional Unit Power-Gating Policies

In this section, various power-gating policies are explored, which determine when to shut a functional unit off. First, to serve as a baseline, an ideal oracle policy, which has perfect knowledge of the future, is presented for both NEMS and transistors . Next, the Flag Policy is presented that requires minimal hardware overhead and no intervention or interface from the operating system. Finally, a hybrid Flag Policy system is presented that uses both transistors and NEMS for functional unit power-gating.

Ideal Transistor Policy:

With perfect knowledge of the future, the ideal policy only shuts a unit off if the future idle period is longer than n_{BE}. Given an instruction trace, let $S_{FU,j}$ be the number of occurrences where functional unit FU is idle for j clock cycles. For a given V_{DD}, the number of sleep cycles is

$$n_{sleep}(FU, V_{DD}) = \sum_{j=n_{BE,T}(FU,V_{DD},W_H)}^{\infty} jS_{FU,j} \tag{4.11}$$

The total boot energy, including header overhead is

$$E_{B,tot}(FU, V_{DD}, W_H) = \sum_{j=n_{BE}(FU,V_{DD})}^{\infty} [E_B(j, V_{DD}) + E_{OH,T}(V_{DD}, W_H)]S_{FU,j} \tag{4.12}$$

The total energy of a functional unit, E_T, consumed by a benchmark for a given V_{DD} and W_H is

$$E_T(V_{DD}, W_H) = E_{B,tot} + n_{active}\alpha C_S {V_{DD}}^2 + n_{sleep}E_L K_L(W_H) + n_{awake}E_L \tag{4.13}$$

where K_L is the factor by which the headers reduce leakage, α is the average activity factor, and n_{active} is the number of cycles where the functional unit is used. Using (4.13) and a brute force optimization search, the optimum V_{DD} and W_H can be determined for a given benchmark.

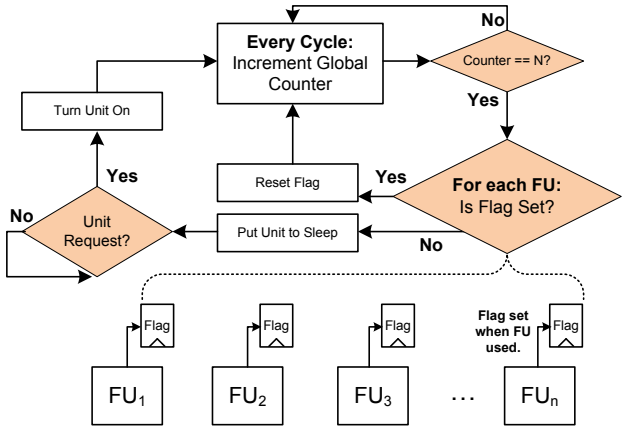

Figure 4.1: Hardware flowchart for the Flag Policy.

```
flagSleep ( N, FU, V_DD )
  sleepFlag = 0, FD = ∅ , curNumSleep = 0
  foreach ( block of N cycles in CPU trace )
    if ( FU not used within block )
      if ( sleepFlag == 0 )
        sleepFlag = 1
      else
        curNumSleep += N
    else
      if ( sleepFlag ==  1 )
        curNumSleep += ( # of cycles in block before FU used )
        FD = FD ∪ { curNumSleep }
        sleepFlag = 0
  return { FD }
```

Figure 4.2: Pseudocode used to analyze a simulator trace for the Flag Policy energy model.

Flag Policy:

The Flag Policy, which is outlined in Figure 4.1, is different from the ideal policy in that it is both simple and realizable in hardware. It operates by assigning a flip-flop-based flag to each functional unit. Whenever a functional unit is used, the flag is set high. A global counter counts up to a certain number of clock cycles, N, and then checks whether each functional unit flag is set. If the flag is set, it is then reset and checked again N cycles later; otherwise the unit is put to sleep. This policy is very compact and has very low hardware overhead, however, since the counter is global, the value of N must be the same for all functional units.

To estimate total functional unit energy given a Flag Policy-based system that uses *transistor power-gates*, let FD be the set of events whereby a functional unit is put to sleep, and each value in FD represents the length of the sleep event in clock cycles. The pseudo-code in Figure 4.2 can be used to calculate FD, given a cycle-accurate simulator trace. The total

time a functional unit is asleep is simply

$$n_{sleep}(FU) = \sum FD \tag{4.14}$$

and the total boot energy is

$$E_{BT}(V_{DD}, W_H) = \sum_{\forall n \in FD} [E_B(FD_n, V_{DD}) + E_{OH,T}(V_{DD}, W_H)] \tag{4.15}$$

These equations can then integrated into (4.13), replacing their ideal policy counterparts, to get total functional unit energy. It should be noted that with transistor power-gates, it is assumed that waking up functional units is the responsibility of the CPU pipeline and that its done early enough to prevent stalling, as is described in [29, 31].

The Flag Policy model for NEMS switches becomes slightly more complicated, due to the fact that the turn on time is much longer than transistors. Since the Flag Policy has no way of predicting the future, if a sleeping unit is needed, the CPU must stall until that unit is turned on. To calculate the stall leakage energy, first it must be known whether or not a given functional unit is asleep when the stall occurs. Let $FS_{n,f}$ be a set of the values 0 or 1. During sleep event FD_n, $FS_{n,f} = 0$ if functional unit f is on, otherwise it is 1 if f is asleep. The stall energy can then be modeled as

$$\begin{aligned} &E_{stall}(V_{DD}, NS) = \\ &\sum_{\forall n \in FD} \Big[\sum_{\forall f \in FU | FS_{n,f}=0} Delay(NS) \cdot E_L(V_{DD}, f) \Big] \end{aligned} \tag{4.16}$$

which can then can be added into (4.13). Note that for NEMS-only policies, $K_L = 0$.

Hybrid NEMS/Transistor Flag Policy:

This study also presents a hybrid power-gating system, where transistor power-gates are in series with a NEMS power-gate, with the transistor switches used for short-term sleep events and NEMS switches used for long-term sleep events. The purpose of this system is

to compensate for the fact that NEMS switches have a limited device lifetime. By allowing transistors to handle the short idle periods, the switch-rate of the NEMS switches can be greatly decreased, thus prolonging the lifetime of the switches. This system is implemented by adding a second counter for NEMS switches in Figure 4.1. When the first counter finds an unset flag, it puts the unit to sleep by turning off the transistor power-gates. When the second, longer counter finds an unset flag it puts the unit to sleep by turning off the NEMS switch, thus reducing the leakage to zero. With this system, the stall energy equation must be modified to account for the the fact that a functional unit can exist in three states: awake, asleep via transistors, and asleep via NEMS. Let FD, T be the delay set for transistors and FD, N be the delay set for NEMS. Also, let FS, N be the functional unit state table for the NEMS counter, which can take three values: 0 if the unit is on, 1 if the unit is off via transistor power-gates, and 2 if the unit is off via NEMS power-gates. The stall energy can then be given by

$$E_{stall}(V_{DD}, NS, W_H) =$$

$$\sum_{\forall n \in FD,N} \Big[\sum_{\forall f \in FU|FS,N_{n,f}=0} Delay(NS) \cdot E_L(V_{DD}, f) + \tag{4.17}$$

$$\sum_{\forall f \in FU|FS,N_{n,f}=1} Delay(NS) \cdot K_L(W_H) \cdot E_L(V_{DD}, f) \Big]$$

Likewise, n_{sleep}, which is used in (4.13) must be modified to account for the fact that when a unit is asleep via NEMS, no energy is contributed to E_T:

$$n_{sleep} = \sum FD, T - \sum FD, N \tag{4.18}$$

Optimization Search:

This section outlined energy models for the five power-gating policies that are examined in this study: ideal NEMS, ideal transistor, Flag NEMS, Flag transistor, and hybrid transistor-NEMS Flag. Each of these policies has a selection of the following tunable parameters: V_{DD},

V_{TH}, relative width of the transistor power-gates (W_H), NEMS switch selection (NS), and flag counter limit (N). The policy with the most tunable parameters is the hybrid system, where V_{DD}, V_{TH}, W_H, NS, and two flag counter limits must be selected. Since the range of these values was relatively small, we were able to use a brute force search to find the optimum settings. This optimization search was done individually for each benchmark yielding a local optimum, which is the set of parameters that minimizes energy. The energy values for all other possible parameters were divided by this number, yielding a normalized energy number which is one at the local optimum, and greater than one elsewhere. Next, for all benchmarks, these normalized energy numbers were added together and the minimum of this sum was found, which is a *global optimum*. Essentially this process treats all benchmarks evenly for the optimization process, despite the fact that some benchmarks consume much more energy than others.

To determine all of the model parameters outlined in the previous section, highly accurate simulations were used to implement the previously detailed energy model, including a SPICE based NEMS simulation environment, a cycle accurate CPU simulator, and SPICE simulations of the various functional units. The various simulations are detailed in the following two sections.

4.4 NEMS Simulation Environment

Unlike the previous chapter, where the NEMS switches were only turned on once when the entire unit was booted up, the switches in this Chapter turn on and off many times during a given benchmark run. The charge pump system presented in the previous chapter would be ill-suited for this purpose because the entire pump must boot up in order to turn the switches on. This section presents a modified system that uses a charge store in order to accumulate a large amount of high-voltage charge. This prevents the charge-pump from having to recharge from 0 V multiple times and saves a considerable amount of energy.

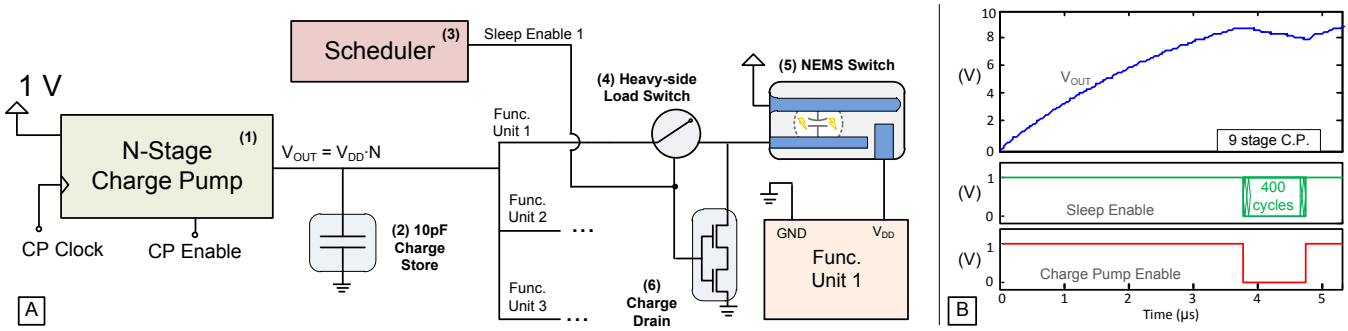

Figure 4.3: A. System level diagram of NEMS-based functional unit power-gating. B. SPICE simulations of the system in (A) with a 9-stage charge pump and 400 NEMS switch actuations.

4.4.1 System Overview

Figure 4.3.A gives a system-level overview of the NEMS-based functional-unit power-gating system. The major modification from the previous chapter is that, at the output of the pump, is a charge store capacitor (2). At boot-up, the charge pump charges the charge-store capacitor to a voltage slightly above the activation voltage of the switch and shuts off to conserve energy. The charge store is sized such that it can supply charge at or above V_{OUT} for hundreds of NEMS switchings before the charge pump must turn on and recharge it.

When the scheduler (3) decides to to wake a unit up, it de-asserts Sleep Enable, turning on an active-low heavy-side load switch (5),which is a device that allows a low-voltage signal to control a high voltage pass transistor. This charges the NEMS switch (5) to V_{OUT}, causing it to turn on and power up the functional unit. When the scheduler decides to put a unit to sleep, the Sleep Enable signal is asserted. This shuts off the load switch and opens the charge drain (6), which quickly drains the charge out of the NEMS switch. Figure 4.3.B shows SPICE simulations of the entire system, with a 9-stage charge pump and a 770 aF NEMS switch. With a 10 pF charge store capacitor, the NEMS switch is able to cycle on and off 400 times before the charge pump must turn back on. When the charge pump does turn back on, it takes a very short amount of time to recharge the charge-store back to its original voltage.

NEMS Switches:

For this Chapter, a selection of switches was chosen from [6] with a variety of mechanical delays and activation voltages. Table 4.1 presents these switches. For all switches, the drain gap is 30 nm, the gate gap is 50 nm, and the beam thickness is 200 nm. The total capacitance is the sum of the parallel plate capacitance between the switch and gate and the parasitic capacitance from the gate to an assumed ground shield above and below the gate (estimated using the equations in [36]). For all NEMS simulations, it will be assumed that the contact resistance is set such that K_D of the functional unit is 1.20. Through charge pump simulations, the activation energy of each switch was determined, and, together with the switching speed, this forms a high level model of a NEMS switch that is suitable for system-level models and simulations. As for cycle count delay, two additional clock cycles were added to the mechanical delay of the switch to account for the time it takes to activate the system in Figure 4.3.

Table 4.1: Physical characteristics of the switches used in this study.

ID	Beam W (nm)	Beam L (μm)	Speed (ns)	Activation Voltage	Total Cap. (aF)
A	100	10	81	3.2	1,595
B	50	5	41	5.0	741
C	75	5	27	7.8	770
D	50	3	15	13.0	445
E	50	2.5	10	19.5	371

NEMS Endurance:

Unlike transistors, which can support a seemingly endless number of switchings, NEMS switches have a finite lifetime. For a fine-grained functional unit power-gating system, if the policy selected is too aggressive, the NEMS switches may cycle fast enough that the device lifetime becomes too limited to be practical. To explore this, the aggressiveness of the Flag Policy, controllable through the N parameter, was varied and it was determined what effect this has on the lifetime of the device. This was done with the 'basicmath' benchmark

and it was assumed that a switch dies after 1E12 on/off cycles, which is on the high end of reported lifetimes [21]. For each value of N, the functional unit with the highest switch rate was determined. From there, it was determined how many days the device can last if the benchmark is constantly processing. In addition, N was set to 2048, which ensures a device lifetime on the order of multiple years, and it was determined what effect this has on the energy savings. Both a NEMS-only and a NEMS/transistor hybrid system were examined.

4.5 Functional Unit Simulation Environment

This section details the simulation environment for the CPU and functional units, including the cycle-accurate simulator, the benchmarks, and the energy and timing of the various functional units.

Cycle Accurate Simulation:

The M5 simulator [3] was used for cycle accurate simulation of a 100 MHz 5-stage 32-bit in-order processor, which is a typical configuration of low-voltage embedded processors such as the ARM 7. The simulated processor has an 8 kB separate data and instruction cache, and a 512 kB L2 cache. There are 7 functional units: an integer ALU, shifter, multiplier, and divider, a pipelined 4-stage floating point ALU, a pipelined 7-stage floating multiplier, and a floating point divider. It was assumed that each pipeline stage of the functional units has its own NEMS-switch. It was also assumed that waking up functional units is the responsibility of the CPU pipeline and that, when transistor power-gates are used, it was done early enough to prevent stalling, as is described in [29, 31]. Therefore, stalling only occurs when a NEMS switch is being activated.

Benchmarks: Eight benchmarks were tested, all of which are from the Mibench suite [12], except for Float FIR, Fixed FIR, and Hamming, which were added to stress certain functional units. Four of the benchmarks are floating point (Float FIR, FFT, Susan, Basicmath), while

four are integer only (Fixed FIR, Rawcaudio, GSM, Hamming). Additionally, Rawcaudio and Hamming only make use of the integer ALU and shifter.

Functional Unit Energy Model:

The functional units were synthesized with an industry standard cell library and simulated in SPICE with a 130 nm industry technology library. These simulations were performed across a range of V_{DD}s and V_{TH}s, after which the various parameters of the energy model in Section 4.2 were fitted to the measured data using Least-Mean-Squares fitting. The integer units were generated using Eudoxus [2], while the floating point units were OpenSPARC units that were stripped down to single precision. To accurately determine switching energy, cycle-accurate simulator traces were analyzed to determine average activity factors of the integer functional units. The values for C_B were also determined through SPICE simulations by measuring the amount of energy it takes to charge the power networks to V_{DD}. Finally, for a given range of relative widths (W_H) and V_{DD}s, the values for K_D and K_L were measured and entered into a 2D lookup table by simulating a large group of ring oscillators that represent an actively processing circuit.

4.6 Results

4.6.1 NEMS Energy Analysis

Table 4.2: Results of charge pump and NEMS switch simulations.

Switch ID (Table 4.1)	**# Charge Pump Stages**	**Pump Bootup Time (μs)**	**Switch Speed (cycles)**	**Energy per switching (pJ)**
A (slow)	4	7.59	11	0.114
B	6	19	7	0.146
C	9	38.1	5	0.468
D	15	77.7	4	1.38
E (fast)	22	147	3	2.43

Table 4.2 presents the results of the charge pump simulations for each of the five NEMS switches. Since each switch has a different activation voltage, they require a different number of stages in the charge pump. It is important to note that these results are different from the previous chapter. In the previous chapter, each time the FFT processor was woken up, the charge-pump had to charge the output all the way from 0 V to the activation voltage, which consumes a considerable amount of energy. With the system in this chapter, a charge store is used to maintain the output voltage at a high voltage. As such, the energy-per-activation is considerably lower.

Also listed is the time it takes for the charge pump to charge the output from 0 V to V_{DD}. This is only applicable if the entire processor is woken from a deep sleep state, since during normal operation, the charge pump maintains the charge store at the activation voltage (Figure 4.3.B). Since charging the capacitance of each NEMS switch to its activation voltage using an ideal charge pump would consume at most a few femtojoules, it is evident that a vast majority of the switching energy comes from charge pump overhead.

4.6.2 Ideal Policy

Table 4.3: Comparison of ideal power-gating policies to no power-gating with respect to percent drop in total functional unit energy.

Benchmark	**No Func. Power-Gating**	
	$\mathbf{V_{OPT}}$	$\mathbf{E_T}$ **@ G. Opt. (J)**
Float FIR	0.55	5.04E-6
Fixed FIR	0.67	8.11E-7
Rawcaudio	0.80	2.82E-7
FFT	0.60	7.59E-6
Susan	0.65	4.41E-6
GSM	0.76	9.56E-7
Hamming	0.73	1.03E-7
Basic Math	0.74	6.05E-7
Global Opt.	**0.72**	

Table 4.4: Comparison of ideal power-gating policies to no power-gating with respect to percent drop in total functional unit energy.

Benchmark	Ideal Transistor-Based			Ideal NEMS-Based		
	V_{OPT}	E_T @ G. Opt. (J)	% Drop	V_{OPT}	E_T @ G. Opt. (J)	% Drop
Float FIR	0.53	4.08E-6	18.9	0.49	3.70E-6	26.5
Fixed FIR	0.59	6.36E-7	21.6	0.59	5.99E-7	26.1
Rawcaudio	0.70	1.67E-7	40.6	0.63	1.28E-7	54.7
FFT	0.56	6.16E-6	18.8	0.55	5.59E-6	26.3
Susan	0.58	3.67E-6	16.8	0.47	3.46E-6	21.7
GSM	0.69	7.51E-7	21.4	0.67	7.87E-7	17.6
Hamming	0.62	7.41E-8	28.1	0.55	6.78E-8	34.2
Basic Math	0.65	4.74E-7	21.7	0.60	4.28E-7	29.2
Average	**0.64**		**23.5**	**0.59**		**29.5**
Ideal Transistor Global Opt: $V_{DD} = 0.64$ V, $W_H = 0.076$ **Ideal NEMS Global Opt:** $V_{DD} = 0.59$ V, NEMS Switch B						

With the ideal power-gating policy, the scheduler has perfect knowledge of the future and will only shut off a unit if 1) it can be turned back on before the unit is needed again, and 2) the energy savings outweigh the energy overhead. Table 4.4 presents the energy model optimization results for the ideal policy. The local optimum voltage and total functional unit energy at the global optimum is shown for each benchmark (see Section 4.3 for how the global optimum is determined). Also for the power-gated systems, the percentage drop in energy is shown, as compared to no functional unit power-gating, which is shown in Table 4.3. It should be noted that both the optimal V_{DD} and V_{TH} are chosen simultaneously such that energy is minimized and the critical delay is 10 ns (for a 100 MHz clock frequency).

There are three significant aspects of these results:

- Power-gating reduces the run-time leakage energy of the processor, which can lead to substantial drops in total energy for benchmarks such as Rawcaudio, which mainly use the adder and shifter. By reducing leakage across the board, a lower V_{DD} and V_{TH} can be selected, with the lower V_{DD} benefiting benchmarks such as FFT, which is highly active and rarely shuts units off.

- The average drop in total energy with the transistor system is 23.5%, whereas with NEMS it is 29.5%. While the difference is not overwhelming, it must be taken into account that NEMS switches are truly advantageous for long idle period. These results demonstrate that the high activation energy and delay of NEMS switches do not hinder their use when targeting shorter idle periods.

- The optimum NEMS switch is B, which is the second slowest switch. This implies that the stalling energy is actually very small. Having B be the optimum switch also has the benefit of requiring a relatively small 6-stage charge pump.

4.6.3 Flag Policy

Table 4.5: Comparison of Flag Policy to no power-gating with respect to percent drop in total functional unit energy.

Benchmark	Flag Transistor-Based			Flag NEMS-Based		
	V_{OPT}	E_T @ G. Opt. (J)	% Drop	V_{OPT}	E_T @ G. Opt. (J)	% Drop
Float FIR	0.53	4.08E-6	18.9	0.55	3.76E-6	25.3
Fixed FIR	0.60	6.46E-7	20.3	0.57	6.15E-7	24.1
Rawcaudio	0.70	1.68E-7	40.6	0.63	1.29E-7	54.4
FFT	0.55	6.16E-6	18.8	0.56	5.70E-6	24.9
Susan	0.61	3.70E-6	16.1	0.59	3.59E-6	18.7
GSM	0.69	7.59E-7	20.6	0.67	7.99E-7	16.4
Hamming	0.62	7.43E-8	28.0	0.55	6.14E-8	40.4
Basic Math	0.65	4.80E-7	20.7	0.60	4.39E-7	27.4
Average	**0.64**		**23.0**	**0.60**		**28.9**
Flag Transistor Global Opt: $V_{DD} = 0.64$ V, $N = 64$, $W_H = 0.076$ **Flag NEMS Global Opt:** $V_{DD} = 0.60$ V, $N = 32$, NEMS Switch C						

Table 4.5 presents the optimization results for the Flag Policy. For the Flag Policy, NEMS switches still outperform transistors by a narrow margin. This is significant because NEMS switches are considerably slower than transistors, which can be considered a major weakness in the context of frequent and fine-grained power-gating. The ideal policy has perfect knowledge of the future, so it is able to mask the slowness, but the Flag Policy does not and

the processor will briefly stall when a unit is needed. As these results show, the effect of this stalling is very insignificant. It is also noteworthy that the Flag Policy results are very close to the ideal results. Despite its simplicity, the Flag Policy is a very effective power-gating policy.

4.6.4 NEMS Endurance Results

Table 4.6: Effect of limiting NEMS switch rate by setting N to 2048.

Benchmark	Flag NEMS-Based			Flag Transistor/NEMS Hybrid		
	V_{OPT}	E_T @ G. Opt. (J)	% Drop	V_{OPT}	E_T @ G. Opt. (J)	% Drop
Float FIR	0.55	4.00E-6	20.6	0.53	3.63E-6	27.8
Fixed FIR	0.58	6.12E-7	24.5	0.57	5.82E-7	28.2
Rawcaudio	0.64	1.28E-7	54.6	0.61	1.18E-7	58.2
FFT	0.57	6.05E-6	20.3	0.54	5.52E-6	27.3
Susan	0.61	3.62E-6	17.9	0.57	3.50E-6	20.6
GSM	0.68	7.18E-7	24.8	0.65	7.33E-7	23.3
Hamming	0.60	6.92E-8	32.8	0.53	5.94E-8	42.4
Basic Math	0.69	5.75E-7	5.0	0.63	4.49E-7	25.8
Average	**0.63**		**25.1**	**0.59**		**31.7**
Flag NEMS Global Opt: V_{DD} = 0.63 V, N = 2048, NEMS Switch C **Hybrid Global Opt:** V_{DD} = 0.59 V, N_{TR} = 32, N_{NM} = 2048, NEMS Switch C						

Of critical importance to a fine-grained NEMS-based power-gating system is how long the system will last before a NEMS switch fails. With the Flag Policy, the time-to-failure can be controlled by setting N, or the value at which the counter checks the functional units, then resets. Assuming that a NEMS switch lasts 1E12 on-off cycles, Figure 4.4.A presents the time till a switch fails, with respect to N. For each point on the graph, the functional unit with the highest switch rate is determined. At $N = 8$, one of the switches would fail before 10 days, yet at $N = 2048$, the switches would last for multiple years.

Increasing N, however, means that the policy is less aggressive and may miss opportunities to put units to sleep. Figure 4.4.B shows the total functional unit energy for one run of 'basicmath', with respect to N. Both the NEMS-only policy and the hybrid NEMS-transistor

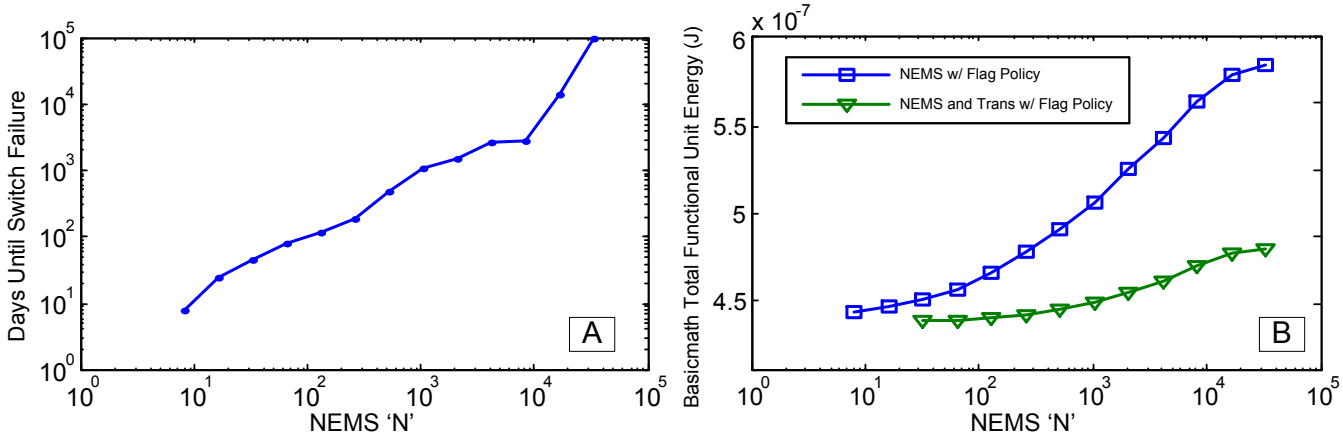

Figure 4.4: A. Effect of Flag Policy aggressiveness (N) on the number of days until failure for the 'basicmath' benchmark. B. Comparing energy of the NEMS-only power-gating policy to the NEMS-transistor hybrid system with respect to N for the 'basicmath' benchmark.

policy are shown. Increasing N for the NEMS-only system causes the energy to increase to the point where it nearly equivalent to no power-gating at all. This is where the hybrid system shows a marked advantage, in that the transistors are able to handle the shorter sleep events, while the NEMS switches handle the longer sleep events and therefore switch less. With the hybrid system, increasing the NEMS N from 32 to 2048 greatly increases the lifetime and results in only a small increase in total energy.

It should be noted, however, that when limiting N to large numbers, the difference in energy between the NEMS-only and the hybrid system is only applicable to certain benchmarks. Table 4.6 presents the differences for all the benchmarks between NEMS-only and hybrid when the NEMS N is limited to 2048. While 'basicmath', shows a large drop in energy, other benchmarks show smaller or no drops at all. With the NEMS-only flag system and $N = 2048$, the average drop in energy is very close to the transistor-only system, but these (and all other) results need to be taken in the context that NEMS switches are best suited for long sleep periods, which this chapter does not consider. If the processor is constantly running for its entire lifetime, this study has demonstrated that, despite their slow switching speed and high activation voltage, NEMS switches can rival transistors as power-gates in terms of energy savings. If there are any longer idle periods, however, as is almost always

the case with embedded systems, Chapter 3 demonstrated that NEMS power-gates would be vastly superior to transistor power-gates.

4.7 Conclusion

To examine the effect of power-gating granularity on the optimum supply and threshold voltage, this chapter investigated a microprocessor with functional-unit level power-gating, considering both transistors and NEMS switches. We presented a detailed simulation framework that includes circuit-level simulations of NEMS switches, benchmark simulations on a cycle accurate simulator, and SPICE-based energy models of the various functional units of the processor. A hardware-based power-gating policy that has very little energy overhead and requires no interface or intervention from the operating system was also presented. Additionally, we formulated a detailed analytical framework for functional unit power-gating that allows for optimization of various circuit and system parameters including supply voltage, threshold voltage, and power-gating aggressiveness.

Our results showed that with an ideal oracle-based scheduler, NEMS-based functional unit power-gating yields a 29.5% average reduction in total functional unit energy with respect to no functional unit power-gating, while an ideal transistor-based system yields a 23.5% reduction. With the more realistic hardware-based Flag Policy, NEMS power-gating yields a 28.9% drop in energy, while transistor power-gating yields a 23.0% drop in energy, which is very close to the ideal results. When scaling back the aggressiveness and limiting the NEMS Flag Policy to an on/off rate that allows for multiple years of operation, the energy savings drops to 25.1%. A NEMS/transistor hybrid policy, however, which adds transistors to this system for short-term power-gating, brings the average energy savings up to 31.7%. The most dramatic gains are in the 'basicmath' benchmark, which increases from a 5% to a 25.8% reduction in total functional unit energy.

The primary focus of this chapter was regional granularity: the microprocessor was broken down into power-gating regions based on functional units, and the scheduler was designed in such a way that hardware-overhead was minimized, which allows a large number of regions. Due to device-lifetime issues, the (non-ideal) architecture with the most promising results was the NEMS-transistor hybrid architecture, which used transistors to target short ideal periods and NEMS switches for long idle periods. That being said, the break-even cycle count of the functional units still ranged from 20 to 100 clock cycles. This is mainly due to the energy-overhead of power-gating, which limits the temporal granularity of the power-gating policy. In the next section, temporal granularity will be examined, and it will be shown how through the use of a logic-style with significantly lower power-gating overhead, idle periods far lower than 20 clock cycles can be effectively targeted.

Chapter 5

Ultra-Fine-Grained Power-Gating with Sense-Amplifier Pass-Transistor Logic

5.1 Introduction

So far, this thesis has presented an exploration of system-wide power-gating of an FFT architecture during long idle periods, and an exploration of fine-grained power-gating, where individual functional units of a microprocessor were shut-down for brief amounts of time. The duration of the idle periods that were targeted with the studied policies, however, tended be above 20-100 clock cycles due to the high energy overhead associated with power-gating. This leaves behind many short idle periods that cumulatively add up to a large number of untargeted idle clock cycles. Consider, for example, the integer multiplier while the GSM benchmark is running. Of the 107,166 clock cycles that the multiplier is idle, 73% of the idle clock cycles occur during idle periods that are 10 cycles or less. If it were possible to target these idle periods, then the functional units could spend a significantly longer time powered down. What stops those idle periods from being targeted, however, is the overhead associated with power-gating a unit on and off. When a unit is turned, charge drains out of the circuit, which must be restored upon power-up. Likewise, the sleep-transistors themselves have a

considerable area, thus the energy required to activate the sleep network is non-negligible. These two factors together lead to the previously discussed *break-even cycle count*, or the point at which the leakage savings outweighs the power-gating overhead. Unfortunately for static-CMOS, the break-even cycle count is anywhere from tens to hundreds of clock cycles, making these idle periods difficult to target.

This chapter proposes that the solution to targeting these very short idle periods is to move away from static-CMOS entirely and find a new style of logic that has a much lower power-gating overhead. Of course, many new styles of logic have been presented in the past few decades, but none have been able to unseat CMOS, save for some very specific applications. The style of logic that this chapter focuses on, however, was born out of necessity due to the inadequacies of CMOS with regards to the high process variation at both ultra-low voltages and state-of-the-art processes. This new logic style, Sense-Amplifier Pass-Transistor Logic (SAPTL), is completely compatible with any CMOS process, can operate at extremely low voltages, is competitive with CMOS in terms of delay, switching energy, and leakage, and can gracefully handle extreme process variation through the use of self-timed logic. As this chapter will show, however, SAPTL has an additional previously unnoticed benefit: the power-gating overhead is significantly lower than CMOS; in some cases, by an order of magnitude.

This low overhead opens the door for a new unexplored topic in low power design: *Ultra-Fine-Grained* power-gating. With this style of power-gating, very brief idle periods can be targeted, which is possible due to the lower power-gating overhead and break-even cycle count. The major drawback of SAPTL, however, is that it is still immature and lacks the well-established design, synthesis, and layout automation processes that accompany CMOS. To get around this, this chapter details small-circuit experiments that use post-layout simulations to accurately compare CMOS and SAPTL. The novel SAPTL synthesis framework presented in Appendix A is also used to yield rough comparisons of larger circuits. Finally, the results from the small and large circuit comparisons are integrated into the simulation framework presented in the previous chapter to analyze the potential energy reduction that

can be achieved on a microprocessor with Ultra-Fine-Grained functional unit power-gating. The key findings of this chapter are as follows:

- Simulations of small, hand-optimized circuits reveal that the power-gating overhead of SAPTL is significantly smaller than CMOS. Compared to a logically equivalent CMOS circuit, a 6-input SAPTL lookup table requires a header that is one tenth the width and consumes up to 40 times less energy upon boot-up.
- Comparisons are made between CMOS and SAPTL 32-bit integer ALU units using a novel SAPTL synthesis approach presented in Appendix A. The trend remains the same: for a 32-bit integer multiplier, 5 times fewer headers are needed, while the boot energy is 15 times lower.
- After substituting the SAPTL power-gating overhead into the functional unit power-gating simulation in the previous chapter, and implementing a new scheduler that immediately shuts off functional units after being used, the drop in functional unit energy increases from 23.0% to 37.6% for a transistor-only system, and 22.3% to 44.0% for a transistor-NEMS hybrid system.

The rest of this chapter is organized as follows: Section 5.2 presents a background on SAPTL operation, while Section 5.3 provides an analysis of why SAPTL has significantly lower power-gating overhead. Section 5.4 presents the methodology used for the experiments in this chapter, while Section 5.5 presents the results.

5.2 Sense-Amplifier Pass-Transistor Logic:

Sense-Amplifier Pass-Transistor Logic (SAPTL) is a relatively new form of logic that has been designed to address the drawbacks of operating with ultra-low voltages [1]. First, SAPTL logic is able to push the boundaries of the energy-delay trade-off. With CMOS logic, it is

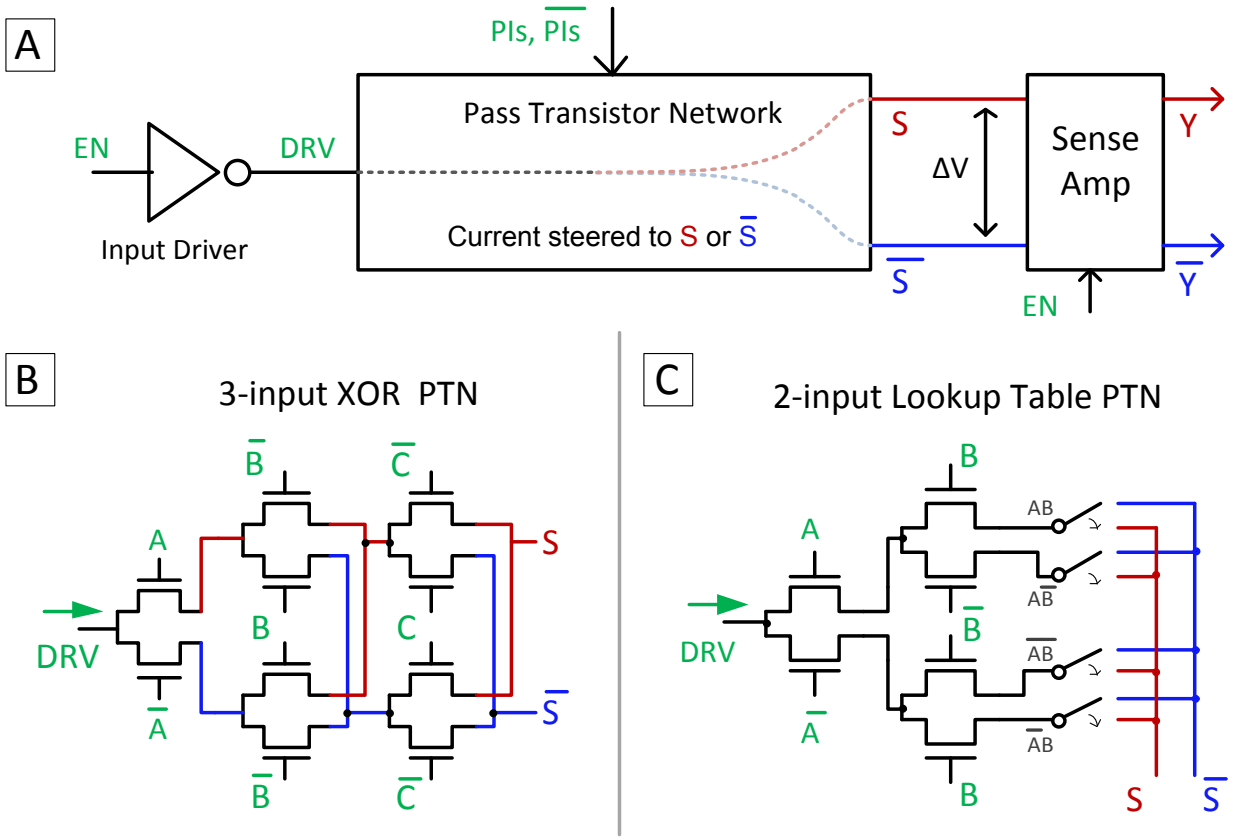

Figure 5.1: A) Overview of an SAPTL Cell, including input driver, pass-transistor network, and sense amplifier. B) Pass-transistor network that implements a 3 input XOR gate. C) Pass-transistor network that implements a 2 input lookup table.

well-known that energy can be reduced while sacrificing delay. This eventually reaches a limit, however, when the CMOS gates lose their positive gain and the logic fails. SAPTL, on the other hand, uses sense amplifiers to amplify the gain and can push the energy-delay trade-off much further. This is greatly beneficial for applications with very tight energy constraints, but virtually no timing constraints. Also, through the use of automatically triggered sense-amps, it is straightforward to adapt SAPTL to asynchronous self-timed logic. Considering that process variation is getting more severe with newer processes, and the severity is magnified by operating in the ULV region, asynchronous SAPTL logic would allow designers to reclaim the large overly-pessimistic timing margins that stem from operating with such high variation.

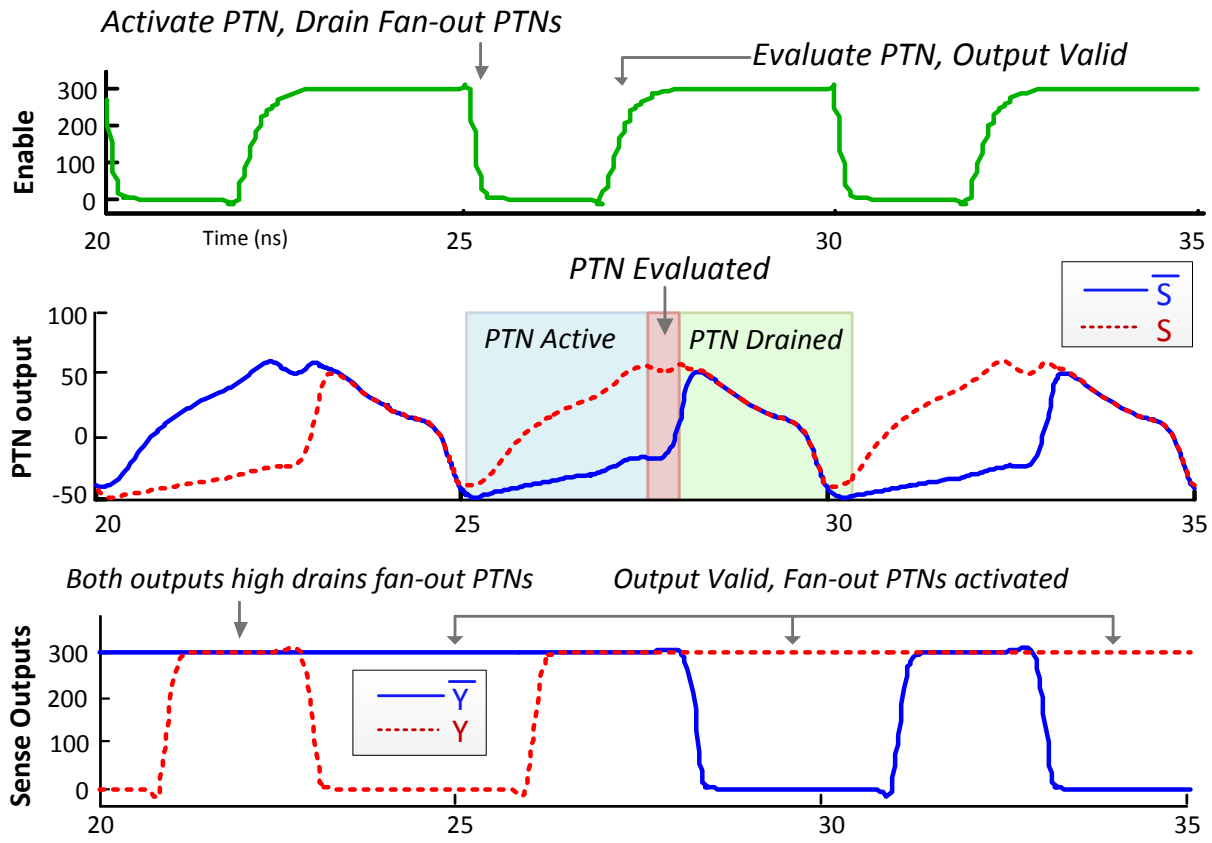

Figure 5.2: Waveforms of an SAPTL cell in action.

Overview of operation:

Figure 5.1 presents an overview of SAPTL operation. Figure 5.1.A presents the implementation of a single SAPTL cell, which evaluates both the output, Y, and its complement, $\overline{Y}$, of a multi-input function. At the heart of an SAPTL cell is a Pass-Transistor Network (PTN) which evaluates the logic function by using NFETs to steer current supplied by the input driver to either S, if the output is 1, or $\overline{S}$, if the output is 0. Two PTNs are shown in Figure 5.1. Figure 5.1.B shows a PTN for a 3-input XOR gate, while Figure 5.1.C shows a PTN for a programmable 2-input look-up table (LUT). These are just simple examples; it is possible to implement, for example, a 20-input XOR gate within a single SAPTL cell.

The PTN essentially is a transistor representation of the binary decision diagram (BDD) of a logic function, which is a now-ubiquitous graph representation of a circuit that is used in synthesis and verification applications [1]. A node of a BDD has two outputs and takes in

a primary input. The traversal through a BDD node is steered through one output if the input is positive, and the other output if the input is negative. Eventually the traversal reaches logic one or logic zero, which represents the output of a function for a given set of inputs. Likewise with the SAPTL PTN, the traversal starts at the root of the PTN, where an input driver supplies current. When the current reaches a node, if the input to that node is one, the uncomplemented transistor steers the current one way. If the input is zero, the complemented transistor steers the current the other way. Eventually, the current reaches either S or $\overline{S}$, representing an output of one or zero. The pass-transistors degrade the voltage at the output of the PTN, so a sense amplifier is used to latch the result, restore the logic level to V_{DD}, and drive the succeeding fan-out cells.

Due to the clocked sense amplifier, the timing of SAPTL is not as straightforward as combinatorial static-CMOS. SAPTL Figure 5.1.D shows a SPICE simulation of a 9-input XOR SAPTL cell. The top trace is the enable signal, which drives both the input driver and the sense amplifier. When the enable signal is low, the input driver is activated and the primary inputs steer the current in the PTN to either the S or $\overline{S}$, shown in the middle trace, depending on whether the output is 1 or 0. The rising of the enable signal triggers the sense amplifier to evaluate the voltage difference of S and $\bar{S}$ and latch the value to the outputs, Y and $\overline{Y}$, of the SAPTL cell. During this time, the inputs to the PTN are all set to high, and the input driver is set to low. This causes the accumulated charge in the PTN to drain, which allows for predictable operation in the following clock cycle. A detailed explanation of operation, including clocking methods (both synchronous and self-timed asynchronous), is available in [1]. Additionally, a novel synthesis approach for SAPTL is presented in Appendix A. It is the first-known synthesis framework for non-programmable SAPTL logic.

The numerous advantages of SAPTL logic in the ULV region have been extensively studied by the designers of the logic. This chapter instead examines another previously unnoticed advantage of SAPTL: compared to CMOS, the energy-overhead of power-gating an SAPTL circuit is significantly lower than CMOS; in some cases, by more than an order of magnitude. This has strong implications for ultra-fine-grained power-gating, where a functional unit may

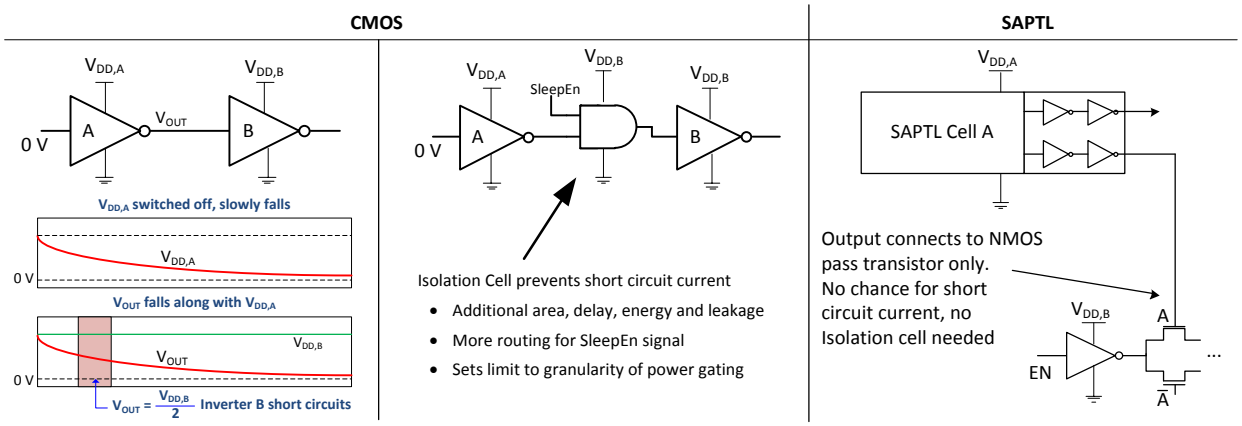

Figure 5.3: CMOS circuits require isolation cells between power-gating regions, while SAPTL circuits do not.

be rapidly turned on and off.

5.3 Low Power-Gating Overhead with SAPTL

SAPTL has three distinct advantages when it comes to power-gating: first, since the input driver completely controls current flow, there is no need for isolation cells between power-gating domains; second, the current spikes are much less severe, meaning much smaller headers can be used; finally, the relative lack of PFETs and simplicity of the power networks means the boot-up energy is significantly smaller. Together these factors make SAPTL an attractive choice for the goal of this chapter: ultra-fine grained, fast switching power-gating of functional blocks.

5.3.1 Isolation Cells

Isolation cells are required for CMOS power-gating between voltage domains, as is demonstrated in Figure 5.3.A. Inverter A is connected to power-network $V_{DD,A}$ while Inverter B is connected to power-network $V_{DD,B}$. If the input of Inverter A is 0 and $V_{DD,A}$ is powered down, then V_{OUT} falls along with $V_{DD,A}$. Once V_{OUT} reaches $\frac{V_{DD,B}}{2}$, both the NFET and

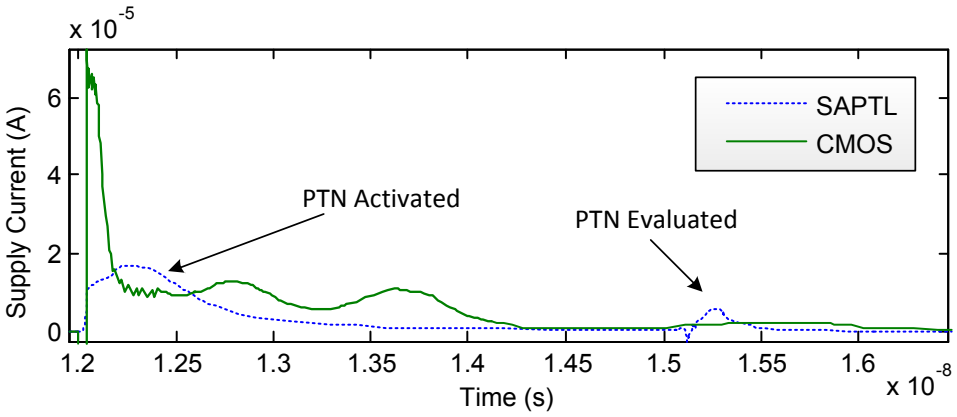

Figure 5.4: CMOS circuits draw larger current spikes than SAPTL. This requires more headers to limit the V_{DD} drop.

PFET in Inverter B will be turned on, which causes a large spike of short-circuit current. The industry-standard solution to this is to insert isolation cells, shown in 5.3.B. When the sleep enable signal is asserted (to logic 0) for $V_{DD,A}$, the AND gate output is driven to zero, which prevents the short-circuit current (the series NFETs also prevents the short circuit current spike in the AND gate). The downside of isolation cells is that the AND gates contribute to area, energy, leakage power, and load capacitance on the sleep network. Additionally, they limit the granularity of the voltage domains. If more voltage domains require more isolation cells, then eventually there will be diminishing returns.

With SAPTL, there is no need for isolation cells. As is demonstrated in Figure 5.3.C, all of the outputs of an SAPTL cell connect to NFETs in a pass-transistor network. If SAPTL Cell A were to be powered down, at no point would a node at $\frac{V_{DD}}{2}$ connect to both a PFET and NFET in series on a separate power-domain. Thus, the granularity of the voltage domains can be increased without the isolation cell overhead normally associated with CMOS.

5.3.2 Current Spikes

When the inputs to a circuit suddenly change, such as from clocked inputs, there is a sudden on-rush of current as the logic switches. The on-rush is the most severe immediately after the inputs are changed (for example, after a clock pulse), because both the short and long paths

are being evaluated. If power-gating is being used, this is the point at which the voltage drop created by the power-gates is the largest. When the circuit is operating in the ULV region, this voltage drop is especially severe because of the exponential dependence of delay on supply voltage. One of the benefits of SAPTL logic is that the magnitude of the current spike is significantly less than CMOS. When the input driver turns on, the pass transistor network slowly charges up to a voltage that is much less than V_{DD}, meaning that, with the exception of the input driver itself switching, there is no sudden on-rush of current.

Figure 5.4 presents the on-rush current of a CMOS 6-input LUT and an SAPTL 6-input LUT. In order to clearly compare the difference in shape of the current spike, the SAPTL cell has a V_{DD} of 0.7 V, while the CMOS LUT has a V_{DD} of 0.35 V. When the CMOS inputs are first changed, there is a severe current spike that can potentially cause a large voltage drop. SAPTL, on the other hand, has a much softer spike which lessens the voltage drop, and therefore mitigates the exponential delay penalty caused by that voltage drop. The end result is that SAPTL cells need fewer power-gates in order to achieve the same delay drop. This leads to less area, as well as less energy needed to activate the sleep network. As will be shown later, this can have a significant positive impact on the effectiveness of ultra-fine-grained power-gating.

5.3.3 Reduced Power-Network Capacitance

A second major source of overhead stems from having to recharge the power networks, which slowly discharge to ground after a unit has been power-gated. This energy is stored in the metal lines that make up the power network, the power-gate drain diffusion, and the PFET source diffusion of the gated circuit[1]. Also, since a CMOS network is complementary, for

[1]N-wells contribute a large amount of capacitance to the power networks, so to make a fair comparison,it is assumed that the N-wells are contacted with an un-gated power line. In the event that the N-Well power line was power-gated, SAPTL would have a huge advantage since the N-well area is much smaller than CMOS.

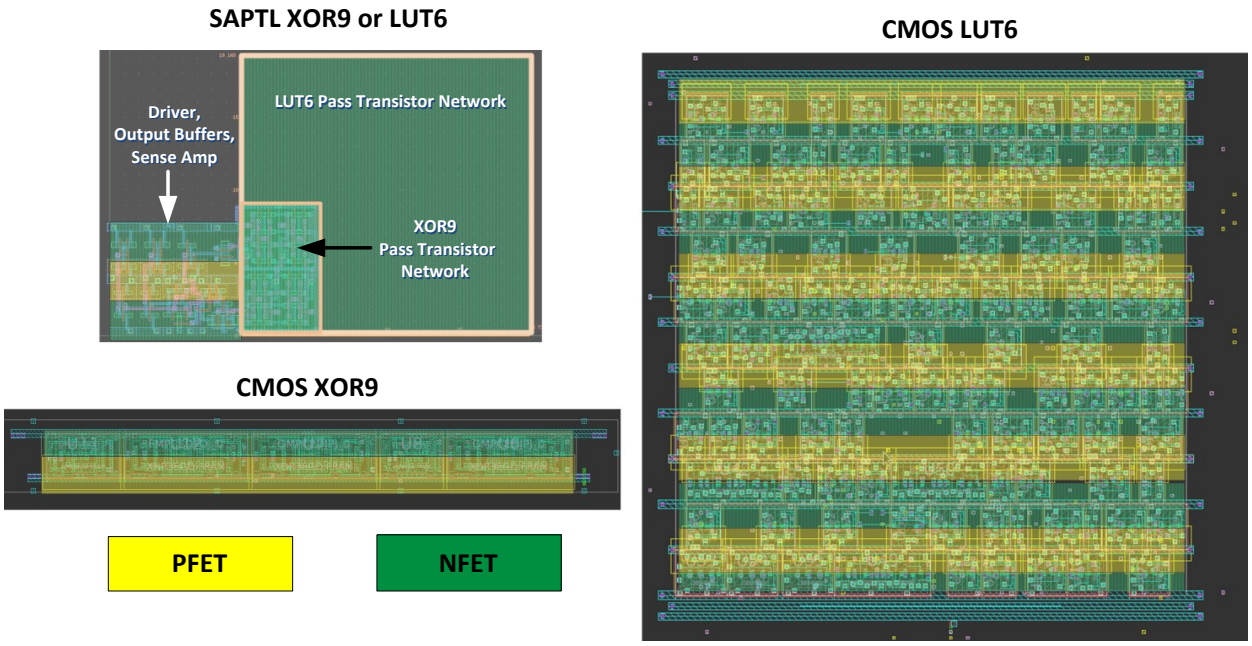

Figure 5.5: SAPTL cells have significantly fewer PFETs than CMOS. This greatly reduces the power network capacitance, and therefore the boot-up energy.

any given input, roughly half the gates in the network are at logic one and half are at logic zero. When the circuit is powered down, the charge stored in the gate capacitance of the nodes at logic one discharges, and assuming the input vector is kept constant, this charge must be restored upon power-up.

A primary source of capacitance on the power network is PFET source diffusion, which in modern processes can be nearly as large as the gate capacitance. With a CMOS network, there is a roughly equal balance between NFET and PFET transistors, so there ends up being a substantial amount of PFETs connected to the power network. With SAPTL, on the other hand, the logic is implemented with an NFET pass transistor network, which is neither connected to power nor ground. The only PFETs in an SAPTL cell are from the input driver, sense amp, and output buffers. Figure 5.5 gives a comparison of CMOS and SAPTL in this regard by highlighting the NFET and PFET regions of their layouts (130 nm technology; SAPTL is a custom-layout, CMOS is a standard cell layout). The SAPTL layout implements the XOR9 function, but the estimated area for a LUT6 function is also shown. The CMOS XOR9 has around 2-3 times higher PFET area than the SAPTL cell,

while the CMOS LUT6 has a substantially larger PFET area. Since a significant amount of these PFETs have a source terminal on the power network, this represents a very large capacitance that must be recharged during each power-gating event. It should also be noted that, with the PFETs confined to a small area in the SAPTL cell, the power network can be more sparse and smaller. This would reduce the wire capacitance of the power network.

The other source of boot-up energy overhead mentioned previously was recharging the gate capacitance of the nodes that are at logic one. This stabilization of the CMOS network consumes a significant amount of energy and is unavoidable. With SAPTL, on the other hand, by carefully setting the state of the cell before it is shut down, a majority of the circuit stabilization energy can be avoided. There are two steps to evaluating a function implemented with SAPTL. The first step is to completely drain the charge out of the pass transistor network. The second step is to drive the root of the pass transistor network high and evaluate the logic function. If the circuit is shut down when the root of the pass transistor network is driven low, then the pass transistor network will be empty of charge. When the SAPTL cell is booted back up, the pass transistor network will still be empty of charge, meaning the cell is effectively already at the second step, and the logic function is ready to be evaluated again. Unlike CMOS, there is no need to drive parts of the logic network to logic one, which leads to lower boot-up energy compared to CMOS.

The lower current spikes and boot-up energy capacitance suggest that the power-gating overhead of SAPTL is significantly lower than CMOS. This would allow for a functional unit that can turn on and off rapidly without incurring much energy overhead, which means a power-gating scheme can target the frequent, yet short idle periods of microprocessor functional units. With that in mind, the rest of this chapter outlines an experimental methodology that explores whether a reduction in energy is achievable by switching to SAPTL and targeting these very short idle periods.

5.4 Experimental Methodology

Comparing one logic style to another is difficult, especially considering how well established static-CMOS is, and how new and immature SAPTL is. In this chapter, a two pronged approach was taken to achieve a fair comparison between the two styles of logic in regards to power-gating. First, two small circuits were compared, of which the SAPTL versions were designed by hand. Second, a primitive, yet functional synthesis framework was used to synthesize SAPTL-based 32-bit integer functional units. The results of the first two experiments were then integrated into the functional unit power-gating methodology used in the previous chapter, along with a new power-gating policy. From there, it was determined what impact the much lower power-gating overhead of SAPTL had on power-gating energy reduction.

Small Circuit Comparison:

The switching energy, leakage power, delay, and power-gating overhead of two small circuits, either of which fits into a single SAPTL cell, were compared in a 130 nm process. The first is a nine-input XOR gate, while the second is a six-input lookup table, which is programmable and can implement any function of six variables. The CMOS versions were synthesized using a standard cell library, while the SAPTL circuits were designed and optimized by hand. The SAPTL circuits were designed according to the dual-clock synchronous methods in [1], but additional circuitry was added that drove both the input drive signal and enable signal low, thus ensuring the PTN stacks were drained during sleep mode. For realistic timing, both the CMOS and SAPTL circuits had properly shaped inputs and were loaded by a fan-out-of-four of the same circuit. Post-layout parasitics were integrated into the SPICE simulations, and Monte Carlo simulations of the SAPTL cells were performed to ensure the sense amplifier was sized large enough to prevent mismatch errors.

Large Circuit Comparison:

Just comparing small, hand-crafted circuits is not a fair comparison because it does not reflect the tremendous amount of time and effort that has been invested in CMOS ASIC design over the past few decades. To make the comparison fair, larger circuits must be synthesized and compared using both a CMOS flow and an SAPTL flow. To make matters difficult, at the time of this thesis, there is only one published method for synthesizing SAPTL circuits. This method assumes that the entire circuit will be implemented using programmable 4-6 input SAPTL LUTs. As such, it uses an FPGA synthesizer, the output of which can be mapped directly into these LUTs.

For this chapter, however, a direct comparison to CMOS is being made. Considering that CMOS circuits are non-programmable and mapping into programmable logic is inherently inefficient, in order to make a fair comparison to CMOS, a hard-wired, non-programmable SAPTL synthesis approach is needed. To this end, an SAPTL synthesis framework was developed for this investigation, the details of which are in Appendix A. The synthesis framework first uses an FPGA synthesizer, and then transforms the resulting mappings into reduced binary decision diagrams (BDDs). A delay model is applied to the BDDs, which yields an estimate of the critical delay of the SAPTL circuit. The synthesis framework was applied to the integer adder, shifter, multiplier, and divider units. The synthesis results, along with the small circuit results were then used to estimate the boot-up energy and header width required to limit the critical delay drop to 20%.

The SAPTL synthesis approach is implemented using off-the-shelf tools that were not designed or optimized for SAPTL. While it does produce functionally equivalent circuits, the approach is crude and inefficient, and the results represent a lower bound of the true potential of SAPTL. To address this, Appendix A presents several potential improvements to the approach that warrant further investigation. Nevertheless, as it will be shown in the next section, even with the crude synthesis approach, SAPTL has vastly lower power-gating overhead compared to CMOS.

Functional-Unit Power-Gating

As a point of comparison, the results of the previous two experiments were integrated into the functional unit power-gating framework that was used in the previous chapter on fine-grained NEMS power-gating. Since the synthesis framework reports the total number of SAPTL cells needed, this number, combined with measured results of the small circuit SAPTL simulations was used to determine boot-up energy of the SAPTL-based functional units. Likewise, the synthesis framework also reports the worst-case number of SAPTL cells switching in a single clock cycle, which was used to determine the header-width requirement.

To determine the impact of the lower SAPTL overhead on the overall effectiveness of the functional unit power-gating scheme, the C_B and W_H values used in the simulation framework (Chapter 4) will be derived from the synthesis results of the SAPTL circuits. It is important to note that, in order to specifically examine the effect of power-gating overhead, the delay, energy, and leakage remained unchanged between CMOS and SAPTL. This way, only the specific impact due to lower power-gating overhead can be examined, without being affected by the other differences between the two styles of logic. Since the SAPTL synthesis framework is not yet able to handle sequential logic, the floating point units used power-gating overhead values based on the integer multiplier results, but scaled in proportion to the gate area. Also, for power-gating structures, both a transistor-only and a transistor-NEMS hybrid system was examined. The NEMS portion of the hybrid system used the Flag Policy presented in Chapter 4 with $N = 2048$.

Immediate Policy

To take advantage of the ultra-fine-grained switching that is achievable with SAPTL, the Immediate Policy was developed. This power-gating policy takes a highly aggressive stance by shutting down a functional unit almost immediately after it is used. This greatly simplifies the hardware overhead of the policy because it can be implemented using the existing pipeline control logic. In this policy, the pipeline waits an additional clock cycle after usage in order to check for another incoming request. If there is no request, the unit is shut down. Like the

Flag Policy, it is assumed that it is up to the pipeline control logic to wake the unit back up. This policy can easily be integrated into the energy model present in Chapter 4. The number of sleep cycles is simply

$$n_{sleep}(FU, m) = \sum_{j=2}^{\infty} S_{FU,j}(j-2) \tag{5.1}$$

This policy switches on and off very frequently, and thus the power-gating overhead must be small for it to be effective. As the results will demonstrate, the power-gating overhead of CMOS is on the high side, and this method is no better, and in some instances much worse, than the Flag Policy. With the overhead at SAPTL levels, however, this incredibly simple policy becomes highly effective.

5.5 Results

Small Circuit Comparison:

Table 5.1: Small-Circuit Comparison of CMOS to SAPTL.

	LUT6		**XOR9**	
	CMOS	**SAPTL**	**CMOS**	**SAPTL**
Supply Voltage (V)	0.45	0.45	0.45	0.45
Threshold Voltage (V)	0.47	0.365	0.487	0.365
Critical Delay (ns)	4.1	4.1	6.0	6.0
Leakge Power (nw)	20.2	12.9	13.8	9.21
Switching Energy (fJ)	40.7	8.78	10.8	4.93
Boot Energy (fJ)	101	2.89	8.73	1.91
$\mathbf{\Sigma}$ Header Width (μm)	8.34	0.80	2.8	0.20

Table 5.1 presents the comparison results between CMOS and SAPTL using post-layout simulations of XOR9 and LUT6 circuits. A striking result is the dramatic reduction in switching energy and leakage power with SAPTL logic. These large reductions are not the

focal point of this chapter, however. The purpose of this chapter is to examine the effect of drastically lower overhead on the effectiveness of ultra-fine-grained power-gating. As such, for the SAPTL functional-unit power-gating experiments, the switching energy, leakage, and delay values for CMOS will be used.

As predicted in Section 5.3, the power-gating overhead is significantly lower with SATPL. With CMOS, moving from an XOR9 circuit to a LUT6 circuit dramatically increases the boot capacitance, the cause of which is clear by looking at Figure 5.5: the PFET area and power network area increases significantly. When moving from a XOR9 SAPTL cell to a LUT6 SAPTL cell, however, the only change in the PFET area is from having larger output buffers and a larger input driver. Also of note is the fact that SAPTL cells also require headers that are roughly 10 times smaller, leading to a significantly smaller amount of energy needed to activate the sleep network.

Large Circuit Comparison:

Table 5.2: Large-circuit Comparison of CMOS to SAPTL.

V_{DD}=0.45, V_{TH}=0.365	**Adder**	**Shifter**	**Multiplier**	**Divider**
Total # of SAPTL Cells	67	69	796	579
SAPTL Network Depth	2	2	15	25
Max Inputs per Cell	16	20	12	9
Delay - SAPTL (ns)	2.79	1.89	45.2	17.0
Delay - CMOS (ns)	8.02	17.2	119	328
Σ Header Width - SAPTL (μm)	14	8.9	47.9	28.2
Σ Header Width - CMOS (μm)	100	92	250	285
Boot-up Energy - SAPTL (fJ)	159	157	1,671	1,530
Boot-up Energy - CMOS (fJ)	492	2,040	24,907	13,770

The integer functional units used in Chapter 4 were synthesized using the SAPTL synthesis framework presented in Appendix A. The results are presented in Table 5.2. The synthesis framework produces a logically equivalent SAPTL network, so the total number of cells and

the total network depth, and the number of cells at each network level, all of which are used to calculate power-gating overhead, can be reported with high confidence. The SAPTL delay, on the other hand, relies on relatively simple analytical estimations, and thus represent rough approximations. Nevertheless, the purpose of reporting the delay is to show that, at a minimum, SAPTL with larger circuits is not drastically slower than CMOS. Even if the actual delay is four times higher than the estimate, SAPTL is still extremely competitive in this respect. This is especially true for the shifter and divider, for which SAPTL seems particularly well suited.

The SAPTL synthesis framework was assembled using off-the-shelf tools intended for other applications and is thus crude and inefficient. Even with this consideration, the power-gating overhead of the larger SAPTL networks is significantly lower than their CMOS counterparts. By improving the synthesis framework, whether through the approaches suggested in Appendix A, or through other techniques, the relative difference can only improve.

Functional Unit Power-Gating with SAPTL:

In order to take advantage of the lower power-gating overhead of SAPTL, a new power-gating policy was developed that simply shuts down a unit immediately after it was used. Table 5.3 presents the results of this policy, using the simulation framework developed in the previous chapter with CMOS functional units. The approach of finding the global optimum V_{DD} remains unchanged, and the percent drop shown is in reference to the non-power-gated results in the previous chapter. Results are shown for both transistor-only and transistor-NEMS hybrid power-gating approaches.

As these results show, with CMOS functional units, the Immediate Policy is no better, and in some cases worse than the Flag Policy. This is because the scheduler is far too aggressive for the power-gating overhead of CMOS; in the case of the GSM benchmark for the transistor-only system, the Immediate Policy actually leads to an significant *increase* in energy, compared to no power-gating. In particular, note how the global optimum V_{DD} is around 0.56 V, whereas with the transistor Flag Policy in Chapter 4, the optimum V_{DD} is

0.64. Most likely, the optimization framework chooses a lower V_{DD} in an attempt to mitigate the high power-gating overhead.

Table 5.4 presents the drop in energy when the power-gating overhead has been reduced to SAPTL levels. Again, it is important to note that the purpose of this experiment was to examine the impact of power-gating overhead, so the delay, switching energy, and leakage power of the functional units remain unchanged from the CMOS experiment. With the SAPTL power-gating overhead, the Immediate Policy becomes a much more effective scheduler: not only is it incredibly simple to implement, but it is also able to target the very brief idle periods. The results are also much more consistent than the Flag Policy. With the Flag Policy, the floating point benchmarks only yielded around a 20% drop in energy, whereas the adder/shifter-only benchmarks yielded a 30%-40% drop. With the Immediate Policy, on the other hand, almost all benchmarks achieved around a 40% drop in total functional unit energy. It is worth noting that these results were obtained using a synthesis framework that was assembled using off-the-shelf tools; by developing a synthesis approach that is tailored

Table 5.3: Comparing the Immediate Policy to both the Flag Policy and no power-gating for CMOS functional units.

Benchmark	**Immediate Policy (CMOS) Transistor**				**Immediate Policy (CMOS) Transistor/NEMS Hybrid**			
	V_{OPT}	**E_T (J) @ 0.56 V**	**% Drop No P.G.**	**% Drop Flag**	**V_{OPT}**	**E_T (J) @ 0.55 V**	**% Drop No P.G.**	**% Drop Flag**
Float FIR	0.50	3.97E-6	21.1	2.7	0.50	3.90E-6	22.5	-7.3
Fixed FIR	0.54	6.62E-7	18.3	-2.5	0.54	6.50E-7	19.8	-11.6
Rawcaudio	0.60	1.63E-7	42.3	2.8	0.57	1.46E-8	48.3	-23.8
FFT	0.50	5.99E-6	21.0	2.7	0.50	5.89E-6	22.5	-6.7
Susan	0.55	3.61E-6	18.2	2.6	0.55	3.56E-6	19.3	-1.6
GSM	0.60	1.12E-6	-17.3	-47.8	0.59	1.11E-7	-16.1	-51.5
Hamming	0.53	6.81E-8	33.9	8.3	0.50	6.42E-8	37.7	-8.1
Basic Math	0.56	4.71E-7	22.1	1.8	0.55	4.56E-7	24.6	-1.6
Average	**0.56**		**20.0**	**-3.7**	**0.55**		**22.3**	**-14.0**

to SATPL, these results can only get better.

5.6 Conclusion

This chapter presented a new frontier in low-power design: *Ultra-Fine Grained* power-gating. With this approach, functional units are almost immediately shut-off after use in order to mitigate leakage power. In order to make this approach feasible, a new style of logic was investigated, Sense-Amplifier Pass-Transistor Logic (SAPTL). This logic was specifically designed for ultra-low-voltage operation in situations where the environmental and process variation is very high. An additional advantage that was detailed in this chapter was its low power-gating overhead, which can be orders of magnitude lower than CMOS. This low power-gating overhead makes ultra-fine-grained power-gating feasible by greatly decreasing the break-even cycle count of the functional units.

Table 5.4: Comparing the Immediate Policy to both the Flag Policy and no power-gating for SAPTL functional units.

Benchmark	**Immediate Policy (SAPTL) Transistor**				**Immediate Policy (SAPTL) Transistor/NEMS Hybrid**			
	V_{OPT}	**E_T (J) @ 0.53 V**	**% Drop No P.G.**	**% Drop Flag**	**V_{OPT}**	**E_T (J) @ 0.45 V**	**% Drop No P.G.**	**% Drop Flag**
Float FIR	0.40	3.13E-6	37.9	23.4	0.39	2.73E-6	45.9	25.0
Fixed FIR	0.45	5.21E-7	35.7	19.3	0.41	4.44E-7	45.3	23.8
Rawcaudio	0.62	1.51E-7	46.6	10.1	0.59	1.38E-8	51.0	-17.3
FFT	0.40	4.70E-6	38.1	23.8	0.40	4.11E-6	45.8	25.5
Susan	0.48	2.69E-6	39.0	27.3	0.45	2.43E-6	45.0	30.7
GSM	0.61	7.63E-7	20.2	-0.5	0.61	7.64E-7	20.1	-4.2
Hamming	0.52	5.76E-8	44.1	22.4	0.45	4.78E-8	53.7	19.6
Basic Math	0.52	3.66E-7	39.5	23.8	0.49	3.30E-7	45.5	26.6
Average	**0.53**		**37.6**	**18.7**	**0.45**		**44.0**	**16.2**

Simulations of small, hand-optimized circuits reveal that the power-gating overhead of SAPTL is significantly smaller than CMOS. Compared to a logically equivalent CMOS circuit, a 6-input SAPTL lookup table requires headers that are one tenth the width and consumes 40 times less boot-up energy. Analysis of larger integer ALU units, which were synthesized using a novel SAPTL synthesis approach presented in Appendix A, demonstrated that the trend continued: the SAPTL circuits required 5 to 10 times fewer headers, while consumes 3-15 times less energy on boot-up. After substituting the SAPTL power-gating overhead into the functional unit power-gating simulation in the previous chapter, and implementing a new scheduler that immediately shuts off functional units after being used, the drop in functional unit energy increased from 20.0% to 37.6% for transistor-only power-gating, and 22.3% to 44.0% for transistor-NEMS hybrid power-gating.

Further reductions in energy can be achieved in two ways. First, the SAPTL synthesis approach can be improved. The approach outlined in Appendix A produces logically equivalent circuits, yet is inefficient for reasons that are outlined in the appendix. Improving the synthesis framework can reduce the number of necessary SAPTL cells, which in turn would lower the power-gating overhead. The second way to achieve further reductions in energy is to increase the granularity of the functional units. As of now, an entire functional unit is turned on, even if only a certain number of least significant bits are needed. Turning on only the bit-slices that are needed would greatly benefit benchmarks such as GSM and Rawcaudio, which perform a large number of 8- or 16-bit computations. This would bring the overall average V_{DD} down as well, which would benefit the other benchmarks. In this regard, power-gating can be *Ultra-Fine-Grained* in two respects: time, by targeting very small idle periods, and area, by subdividing functional units and allowing individual bit-slices to turn off.

This chapter concludes our investigation into emerging power-gating techniques. This investigation began by exploring applications with very long idle periods and demonstrated how NEMS switches can completely eliminate off-state leakage and drastically lower the average power consumption. More fine-grained power-gating was then explored, including a micro-

processor with the ability to individual power-gate functional units through the use of a hardware-only power-gating policy. This investigation concluded by introducing *Ultra-Fine-Grained* power-gating, which uses SAPTL, an emerging ultra-low voltage logic, to greatly lower the overhead of power-gating, allowing very short idle periods to be targeted. Throughout the course of this investigation, it was demonstrated that, through aggressive leakage control, it is possible to select both a low threshold voltage and a low supply voltage, meaning a ULV circuit can be low-energy without being low-performance.

Chapter 6

Conclusion

In this thesis, emerging technologies were presented that can greatly increase the effectiveness of power-gating, which has a broad impact on the energy efficiency of ultra-low voltage (ULV) circuits. Present power-gating implementations suffer from limitations including non-zero off-state leakage, which can aggregate to a large amount of wasted energy during long idle periods, and high energy overhead, which limits their use to long-term system-wide sleep modes. As was shown, however, by vastly increasing the effectiveness of power-gating through the use of emerging technologies, and implementing aggressive hardware-oriented power-gating policies, leakage in microprocessors can be eliminated on a large scale. This allows the threshold voltage to be lowered, leading to ULV microprocessors with both low switching energy and high performance.

The first emerging technology investigated was the Nanoelectromechnical-Systems (NEMS) switch, which is a CMOS-compatible mechanical relay with zero off-state current and low on-resistance. A detailed simulation framework for NEMS switches was presented that integrated published data from previously fabricated switches, along with SPICE simulations of the charge-pump system needed to activate the switches. The results demonstrated that for devic1es with long idle periods, huge drops in power consumption were possible. For example, a NEMS-power-gated architecture performing an FFT per hour consumes 30 times

less power than a transistor-power-gated architecture, while lowering power-gating area overhead by 36-83%. By eliminating the off-state leakage, it was also shown that architecture selection becomes more straightfoward. With non-zero off-state leakage, the optimum FFT architecture, in terms of complexity, was shown to be highly dependent on the length of the idle periods. With the zero off-state leakage of NEMS switches on the other hand, no matter how long the idle periods are, the optimum architecture is generally the one that is the most efficient in terms of switching energy.

NEMS switches were also compared to transistors in the context of a fine-grained power-gating system that targets shorter idle periods and individual functional units of a microprocessor. Despite having a longer activation delay, higher activation energy, and limited device lifetime, NEMS switches outperformed transistors across the board in terms of energy reduction. Using the discussed hardware-based Flag Policy, the NEMS-power-gated system yielded a 28.9% drop in energy, while the transistor-power-gated system only yielded a 23.0% drop in energy. While the difference is not overwhelming, taking into account that the NEMS switches are vastly superior for longer idle periods, these results demonstrate that the drawbacks of NEMS switches do not hinder their use when targeting shorter idle periods with a more fine-grained power-gating system.

The second technology targets the high energy overhead associated with powering a circuit on and off. CMOS logic suffers high power-gating overhead for three reasons: high capacitance on the power-network, severe current spikes that require large headers to prevent performance loss, and the need for isolation cells at the boundaries of power-gating regions. This thesis demonstrated that a new logic style specifically designed for ULV operation, Sense Amplifier Pass Transistor Logic (SAPTL), has significantly lower power-gating overhead. Compared to static-CMOS, SAPTL circuits consume up to 15 times less energy on boot-up, and require 8 to 10 times fewer headers. These results were determined through both post-layout SPICE simulations of small circuits and large circuit comparisons, using a novel SAPTL synthesis approach. Microprocessor simulations demonstrated that a fine-grained power-gating policy, along with the drastically lower overhead of SAPTL, can result in up to a 44% drop in

energy.

This thesis also provided an energy estimation framework built around a cycle-accurate microprocessor simulator, which allows a wide range of circuit and power-gating parameters to be optimized, including supply voltage, threshold voltage, power-gating aggressiveness, and power-gate dimensions. This framework implements two hardware-based power-gating schedulers that are completely invisible to the OS, and have extremely low hardware overhead. This paves the way for future research into power-gating systems with much finer regional granularity. All together, this thesis presents the tools that can analyze a microprocessor with many fine-grained power-gated regions, the policies necessary to implement this system with minimal hardware overhead, and emerging technologies that promise to greatly increase the efficiency of power-gating, allowing for unprecedented levels of energy efficiency and performance in ULV circuits.

6.1 Recommendations for Future Work

The next step would be to fabricate integrated circuits that implement the techniques in this thesis. Though the most accurate simulation methods possible were used, including SPICE simulations with post-layout parasitics, cycle-accurate simulation, and NEMS models based on fabricated switches, a fabricated chip would provide definite proof of the effectiveness of the proposed techniques. The SAPTL experiments are perhaps the easiest to fabricate since layouts are already completed, and the logic style is CMOS compatible. NEMS switch fabrication, however, would be much more difficult because it requires foundry cooperation, which is difficult in a university setting. A good intermediate step to eventual fabrication in a foundry would be to use university fabrication facilities to produce the switches on a separate die, then use 3D integration techniques to bond them to a CMOS die.

There is also much more work to be done on the design and analysis of fine-grained power-gated microprocessors. One aspect that this thesis did not address is breaking up the func-

tional units into more power-gating regions. For example, the integer units could be split up into bit-slices: if the upper 16 bits of a multiplier are not needed in a calculation then they do not have to be turned on. This would require additional design work in the pipeline logic in order to determine how many bit-slices are actually needed. Likewise, the individual pipeline stages of the floating-point units could be targeted. This thesis also did not address the control-oriented functional units of a microprocessor, including the pipeline-control logic, I/O modules, branch unit, exception handler, etc. as well as the logic portion of the memory caches. These units do not resemble the data-oriented functional units examined in this thesis, and it would therefore be more challenging to develop a power-gating policy for them. Nevertheless, it has been demonstrated thoroughly in this thesis that highly aggressive power-gating yields large reductions in energy, so no stone should be left unturned.

In all, these recommendations highlight the overall theme of this thesis: eliminate leakage as much as possible by increasing both the temporal and regional granularity of the power-gating policy, while at the same time greatly reducing the overhead and limitations associated with power-gating. In doing so, the supply and threshold voltage can be lowered, which allows for low energy processing with levels of performance that can greatly expand the capabilities of energy-constrained devices.

Appendix A

SAPTL Synthesis Framework

When new styles of logic like Sense-Amplifier-Pass-Transistor-Logic (SAPTL) are developed, often they are compared to the ubiquitous standard, CMOS, by a series of small hand-crafted circuits. Often this provides an unfair comparison because it does not reflect the decades of effort put into the CMOS design flow, including the ability to synthesize and analyze circuits with potentially millions of gates. This dissertation proposes that very fine-grained leakage control is possible by selecting a style of logic such as SAPTL with drastically lower power-gating overhead. In order to provide for a fair comparison, the synthesis of large circuits is necessary because it may be the case that mapping SAPTL to large circuits results in large inefficiencies that are not reflected in small hand-crafted circuit comparisons. There are no publicly available tools to synthesize RTL to SAPTL cells, so this appendix presents a method for synthesis that uses a combination of analytical modeling and off-the-shelf tools. As will be shown, this approach contains many inefficiencies that need to be addressed in the long run, but it allows a very conservative upper bound to be obtained on the final delay and area of an SAPTL circuit.

A.1 Overview

SAPTL is distinct from static-CMOS in that an individual SAPTL cell is able to implement large logic functions of many variables within a single block. Thus it differs from CMOS synthesis, where a logic function is decomposed and mapped into a set of predefined gates in a library. The style of logic that is closest to SAPTL and well established is a Field-Programmable Gate Array (FPGA), which implements an N-input lookup table (LUT) that can implement any logic function of N inputs. The similarity stems from the fact that both synthesis approaches would start with a circuit implemented as a series of simple gates, then group them together into larger cells with many inputs. It therefore stands to reason that FPGA tools are a good place to start when implementing an off-the-shelf SAPTL synthesizer.

An overview for the synthesis approached used in this thesis is as follows: First, an open source FPGA synthesis tool is used to transform gate-level Verilog into an FPGA LUT mapping. Next, for each LUT, the equation for that LUT is transformed into a binary decision diagram (BDD). A binary decision diagram is a well-established graph-based representation of a circuit and is used extensively in synthesis and verification applications. It also just so happens that the pass transistor network of an SAPTL cell is structurally identical to the BDD of the implemented logic function, which means the well-established BDD techniques can be used for synthesis [1] .

After synthesis, an analytical delay model is applied to the BDD of each node to determine the worst-case delay of the SAPTL cell. Finally, given the network of SAPTL cells and their associated delay, the total delay of the network can be determined. The resulting node count and delay can be used as feedback to adjust the parameters of the original FPGA synthesis tool in order to find settings that produce a good balance between node count and delay.

This approach does produce a functional, logically equivalent circuit, however FPGAs and SAPTL networks are vastly different in one respect: an FPGA LUT implement any N-input logic function without there being any difference in the resulting power, delay, or area of

the LUT. Thus, FPGA synthesizers strive to minimize the total LUT count of the circuit and the number of LUT stages on the critical path. With SAPTL, on the other hand, a single cell can implement a large logic function of many inputs, but the size of the Binary Decision Diagram (BDD) of that function plays a large role in the power, delay, and area of the SAPTL cell. An FPGA synthesizer is unaware of this fact, so this represents a major inefficiency of using an off-the-shelf FPGA synthesizer. This inefficiency, as well as other potential improvements are presented at the end of the appendix.

A.2 FPGA Synthesis and BDD Reduction

The FPGA synthesizer used was the open-source ABC synthesis package [4]. The reason for choosing this package is that it allows for FPGA synthesis with LUT sizes up to 32 inputs. This is useful since it is feasible for an SAPTL cell to implement logic functions with that many inputs, whereas state-of-the-art FPGAs currently generally only implement 6-7 input LUTs, thus ruling out commercial tools. Additionally, the open source nature allows for future modification of the FPGA algorithms for the purpose of tailoring them more towards SAPTL. The first step to synthesis is to convert the Verilog RTL to gate-level Verilog using Synopsys Design Compiler and a small library of basic gates. From there, the ABC tool reads in the gate-level netlist, converts it to an intermediate structure, and runs the FPGA synthesis algorithm for a given LUT size. For a K-LUT run, the end result is a network of nodes, with each node represented as a logic equation having K or less inputs. These equations are then transformed to BDDs using the BDD tools internal to ABC.

Transforming a logic equation to a BDD is not a straightforward procedure, since the size of the resulting BDD is heavily determined by the order at which the input variables are extracted. In some cases, the optimum variable ordering leads to a BDD size that increases linearly with circuit size, whereas the least optimum variable ordering leads to an exponentially increasing BDD size. As such, fast, yet complicated heuristics exist to determine a

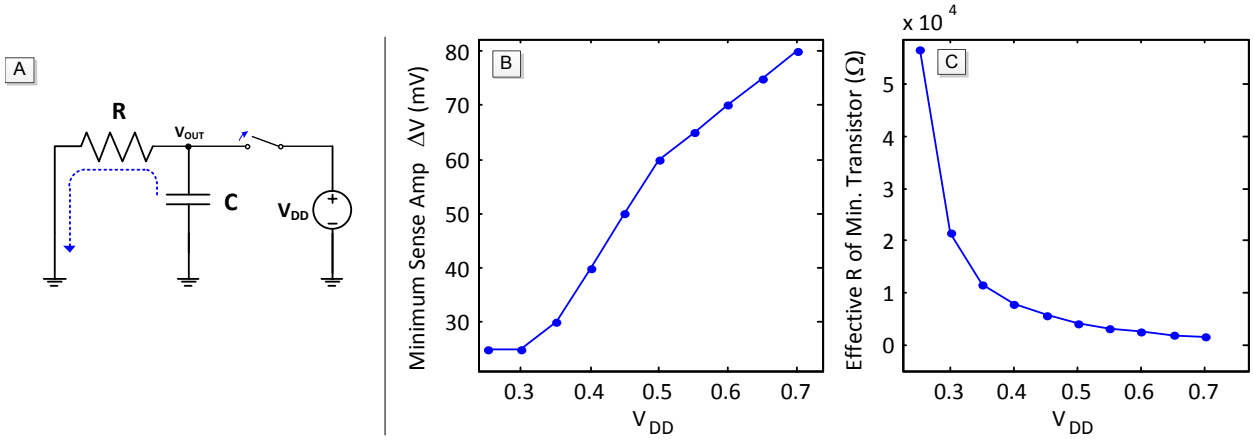

Figure A.1: A. First-order discharging RC circuit. B. Minimum ΔV required for sense amp to register correct value, with respect to V_{DD}. C. Effective resistance of a W=280 nm, L=120 nm transistor.

good variable ordering. The SAPTL framework uses the built in BDD reorder command of the ABC package, which works reasonable well. As will be detailed later, however, the BDD reorder command only tries to minimize the number of nodes in the BDD. In the context of SAPTL, nodes in a BDD can be rearranged in such a way that the total node count remains unchanged, but the delay and energy are significantly impacted. This is not considered by ABC, and is an area for potential improvement.

A.3 Delay Estimation

An important part of any synthesis framework is a method for quickly determining the critical delay of a resulting logic network. In most synthesis frameworks, there is a large number of controllable parameters, and quick delay estimations allow for the parameter space to be explored in a reasonable amount of time. In this section, a simple RC-based delay estimation method is presented that can quickly and accurately determine the critical delay of a synchronous SAPTL network.

The delay estimation framework starts by substituting simple resistors and capacitors for the NFETs in the pass transistor network. This is a similar approach to the methods used for hand-analysis of static-CMOS circuits. The capacitors are used to represent the diffusion capacitance of the NFETs, while the resistors represent the effective resistance of the channel when it is on. Unlike CMOS circuits, the on-path in the network of an SAPTL pass-transistor network contains no branches and only diffusion capacitance, so the resulting RC network resembles a simple RC ladder. The Elmore Delay Model, a long-standing approximation for RC ladder networks, can be used to achieve an accurate approximation of the rise time at the output. The Elmore Delay Model is based on the equation for the discharge time of the first-order RC network shown in Figure A.1.A. The output voltage with respect to time is given as

$$V_{OUT} = V_{DD} e^{\frac{-t}{RC}} \tag{A.1}$$

Solving for t yields

$$t = -RC\, ln\left(\frac{V_{OUT}}{V_{DD}}\right) \tag{A.2}$$

With SAPTL logic, the pass-transistor network drives a sense amp that senses the differential voltage across S and $\bar{S}$. Through extensive Monte Carlo simulations, it is possible to determine a minimum ΔV value required between S and $\bar{S}$ for the sense amplifier to correctly evaluate the output. This value is dependent on V_{DD}, as is shown in Figure A.1.B, which plots $\Delta V(V_{DD})$ for a 130 nm sense amplifier. Substituting $\Delta V(V_{DD})$ into (A.2) yields

$$t = -RC\, ln\left(\frac{\Delta V(V_{DD})}{V_{DD}}\right) \tag{A.3}$$

which is the time it takes the output voltage of the first order network in Figure A.1 to reach ΔV. Equation (A.4) can be further simplified by defining effective resistance, R', as

$$R' = -R\, ln\left(\frac{\Delta V(V_{DD})}{V_{DD}}\right) \tag{A.4}$$

This effective resistance is generally constant for a given transistor sizing, ΔV, and V_{DD} and can be determined through SPICE-based curve fitting. Figure A.1.C gives the effective

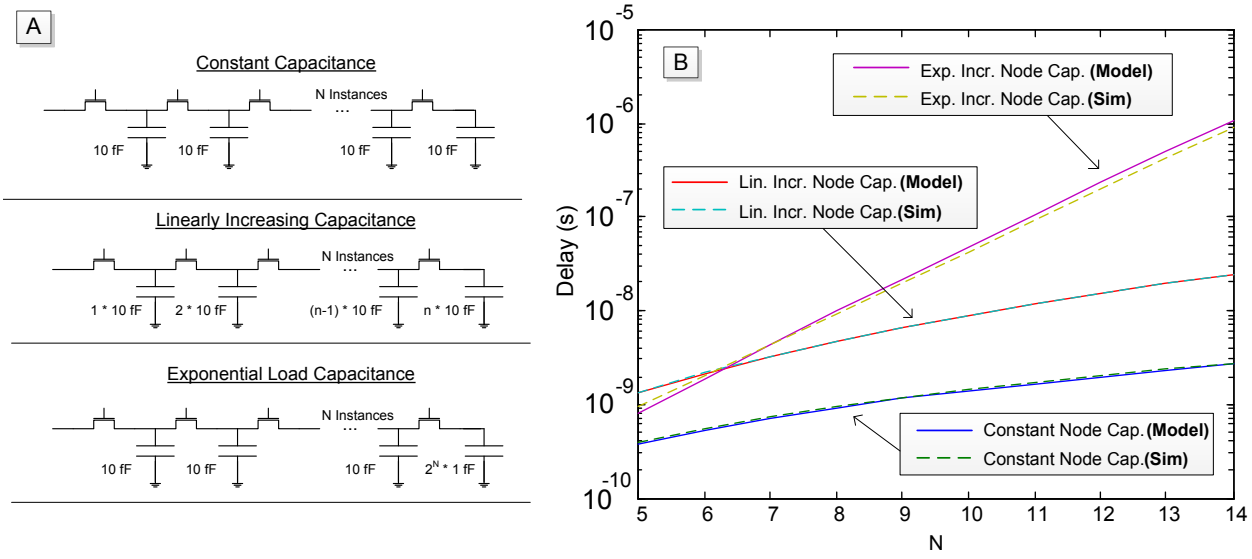

Figure A.2: A. Three parameterized RC ladders used to validate the analytical model. B. Simulation vs. model of RC ladder delay.

resistance of a minimum sized 130 nm transistor, with respect to V_{DD} and the ΔV values in Figure A.1.B. The equation for delay of the pass transistor network then becomes

$$t = R'C \tag{A.5}$$

These equations are only valid for a first order network, which is a single resistor and capacitor. The equivalent RC network of the on-path would have many more elements, yet the exact equations for higher order systems quickly become too complicated to work with. This is where the Elmore Delay Model is used, which is a simple yet accurate delay approximation for higher order RC networks. The equation for delay at the output is simply

$$\sum_{\forall i} R_i C_i \tag{A.6}$$

where R_i is the total resistance from node i to the source and C_i is the capacitance at i. It should be noted that, according to (A.5), as long as the pass-transistor network delay can be given in terms of $R \cdot C$, that it is on-resistance of a minimum sized NFET times source/drain diffusion capacitance of a minimum sized NFET, the results can be compared in a technology- and supply voltage-independent manner.

```
Function BDD-WC-delay
Input: BDD
Output: Worst case (WC) delay of BDD (in terms of RC)
delayTo = hash ( key: node value: WC delay to vertex )
resTo = hash ( key: node value: source/node resistance of WC path )
For each node n in topological_sort( BDD )
  For each edge (n, w)
      if( !exists delayTo{ w } )
        delayTo{ w } = resTo{ w } = 0
      newDelay = delayTo{ v } + cap_at( w ) * ( resTo{ v } + 1 )
      if( delayTo{ w } <= newDelay )
            delayTo{ w } = newDelay
            resTo{ w } = resTo( v ) + 1
return max( delayTo{ bdd_one }, delayTo{ bdd_zero } )
```

Figure A.3: Pseucode used to determine worst-case delay of an SAPTL cell.

To validate the delay model, three parameterized N-stage on-path NFET networks shown in Figure A.2 were simulated in SPICE with a 130 nm library and a V_{DD} of 0.5 V. The first network has a constant capacitance of 10 fF at each node. The second network has a capacitance at each node equal to $d \cdot 10$ fF, where d is the number of NFETS between that node and the source. The third network has a constant capacitance of 10 fF at each node, with the expectation of the last node, which has a capacitance of 2^N fF. All of the transistors were sized W = 280 nm, L = 120 nm. Figure A.2 compares the model to the SPICE simulations with respect to N. Considering the simplicity and speed of the model, the error is sufficiently small for use with the synthesis framework.

A.3.1 Worst Case Delay

The previous method determines the delay of a single on-path, but more work is needed to find the critical delay of an SAPTL cell, i.e. the set of inputs that leads to the longest delay. This can be done by performing the longest path algorithm on the BDD representation of the pass transistor network. It is assumed that minimum sized transistors are used for pass transistor network and the resistance of each transistor is identical. Weights are assigned to the edges equal to the number of source or drain terminals exposed to the node. The longest path algorithm is then run on the resulting network according to the pseudocode in

Figure A.3. Sneak-paths allow current to flow back up the pass transistor network, which effectively increases the capacitance of a node and therefore the delay. The *cap_at* function searches for sneak leakage paths by backtracking up the off-paths of the network.

Once the worst-case delay of the individual SAPTL cells are known, the delay of the entire SAPTL network can be determined. In this thesis, a pessimistic approach is taken. Since each synchronous SAPTL cell latches it output, the entire SAPTL network operates similar to a pipeline. Assuming there is only a single global clock, this means the overall clock period is limited by the delay of the slowest SAPTL cell. The total critical delay of the entire network is therefore the maximum depth of the network, i.e. the path from input to output that contains the most SAPTL cells, multiplied by the delay of the slowest SAPTL cell.

A.4 Results

To analyze the effectiveness of the SAPTL synthesis framework, a 32-bit adder and 32-bit shifter were synthesized using the framework. Figure A.4.A-C present the results for the adder and Figure A.4.D-F presents the results for the shifter. Figure A.4.A and Figure A.4.D present the total number of SAPTL cells in the network for the adder and shifter, respectively. As the maximum number of allowable inputs in an SAPTL cell (or LUT, to the FPGA synthesizer) is increased, the total number of required nodes decreases. Presented in Figure A.4.B and A.4.E is the maximum network depth, which also decreases as the maximum LUT inputs increase. This first two numbers are what the FPGA synthesizer is trying to reduce since, to an FPGA-based circuit, total number of nodes and logic depth directly impact the delay and power consumption of an FPGA. After running the FPGA synthesizer, each SAPTL cell is converted to a BDD and reduced using variable reordering. From there, the worst-case RC delay of each BDD is calculated.

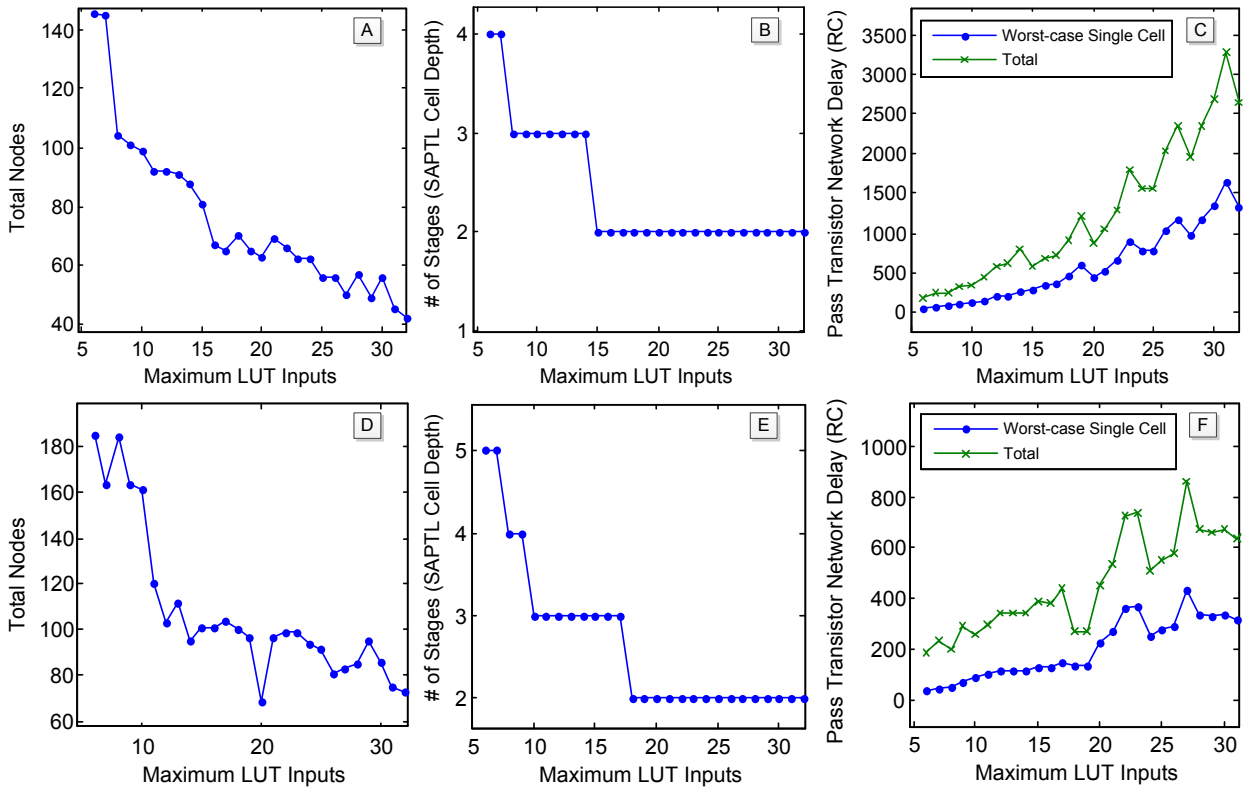

Figure A.4: All graphs are with respect to maximum single-cell SAPTL inputs allowed by FPGA synthesizer. A. Total SAPTL nodes of synthesized 32-bit adder. B. Number of stages in 32-bit SAPTL adder. C. Worst-case single-cell delay and total delay of 32-bit SAPTL adder. D. Total SAPTL nodes of a synthesized 32-bit shifter. E. Number of stages of in 32-bit SAPTL shifter. F. Worst-case single-cell delay and total delay of 32-bit SAPTL shifter.

All delay numbers in Figure A.4 are given in terms of RC. For delay results specific to a technology, see Chapter 5. Both the worst case single cell RC delay and the total RC delay are plotted for the adder and shifter in Figure A.4.C and Figure A.4.F, respectively. It is clear that, as the number of LUT inputs increases, the total delay increases drastically. This is primarily due to the inherent inefficiencies of using an FPGA synthesizer, which are detailed in the next section. In attempt to balance area, leakage, and delay, the 16-input result for the adder and 20-input result for the shifter can be considered sweet spots.

A.5 Potential Improvements

The goal of the synthesis framework was to use off-the-shelf FPGA tools to produce functionally correct SAPTL circuits. The downside of the off-the-shelf approach is that SAPTL is a fairly unique logic style and has many differences from FPGAs. In this section, sources of inefficiency are identified and potential solutions are suggested that can greatly improve the overall efficiency of SAPTL synthesis.

A.5.1 Delay-Aware Critical Path Mapping

The most glaring inefficiency is the fact that the FPGA synthesizer does not consider the internal delay of an SAPTL cell. This is because it models the LUT delay as constant, regardless of what logic function it implements. The synthesizer instead attempts to minimize the LUT count and the logical depth of the LUT network [30]. The solution to this problem should come in two parts. First, additional circuitry should be created that allows for an SAPTL cell to be evaluated in two or more clock cycles. This way, if an SAPTL cell does not reside on the critical path of a network, it can have a large delay without slowing down the entire circuit. The second part should focus on putting smaller SAPTL cells with less delay on the critical path. This is similar to the way conventional synthesis algorithms operate.

It may be possible to improve the results by adding a delay-based cost function to the FPGA synthesizer. The difficulty with this, however, is that FPGA synthesis algorithms have been optimized over the years with complicated heuristics that are very specific to FPGAs. To get truly effective results, a new algorithm needs to be developed that is tailored specifically to SAPTL. A starting point could be to formulate the problem as a minimum-cost binate covering problem, as is detailed in [30], with the cost being a metric based on delay, area, and energy. Exact solutions for binate covering problems take a very long time to solve, so efficient heuristics would have to be built on top.

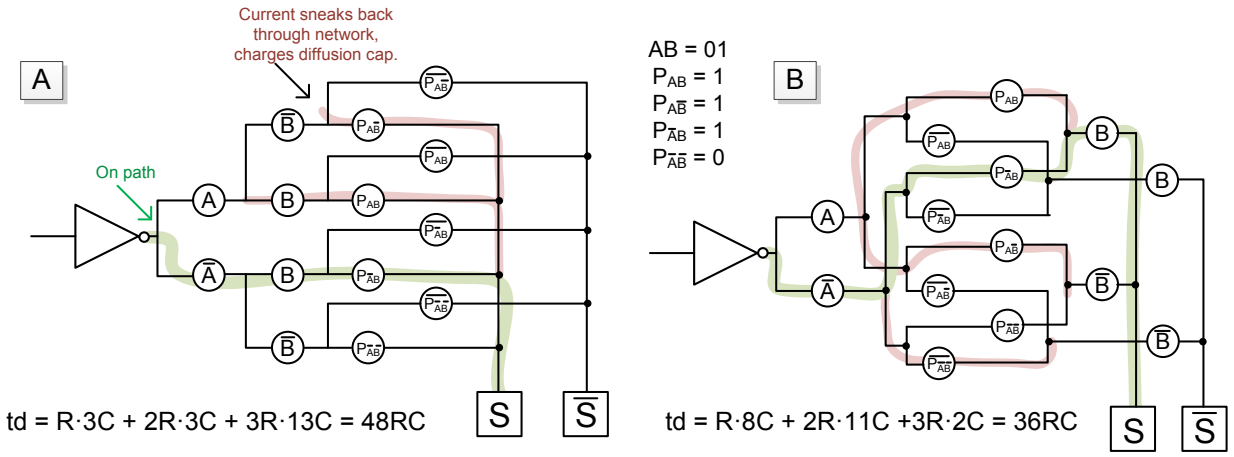

Figure A.5: A. Resulting network of a 2 input LUT using built in BDD reorder package of ABC. B. Rearranging the nodes by hand yields an RC delay that is 25% lower.

A.5.2 BDD Remapping to Mitigate Sneak-Paths

Binary decision diagrams are used to determine the pass transistor network for a given logic equation. During the BDD reduction phase, input variable reordering algorithms are used to determine a variable order that minimizes BDD node count. What is not considered in this step is the delay of the resulting BDD, as it is possible to have two logically equivalent SAPTL networks with the same node count, but with different delays. Figure A.5 presents two logically equivalent pass transistor networks for an SAPTL 2-input look-up tables. On the left is the resulting BDD from the ABC synthesis package, while on the right is a rearranged network. Because two wires branch into the B and $\overline{B}$ node, it does not technically implement a BDD.

With the given input vector, there exists sneak paths (shown in read) where current actually flows back into the pass transistor network, leading to additional delay. Based on the Elmore Delay Model, moving the point at which these sneak paths start closer to the source can decrease overall delay. In the example shown, by reorganizing, the RC delay is reduced by 25%. To facilitate this, a post-BDD algorithm is needed that can examine how reordering nodes can mitigate sneak leakage paths.

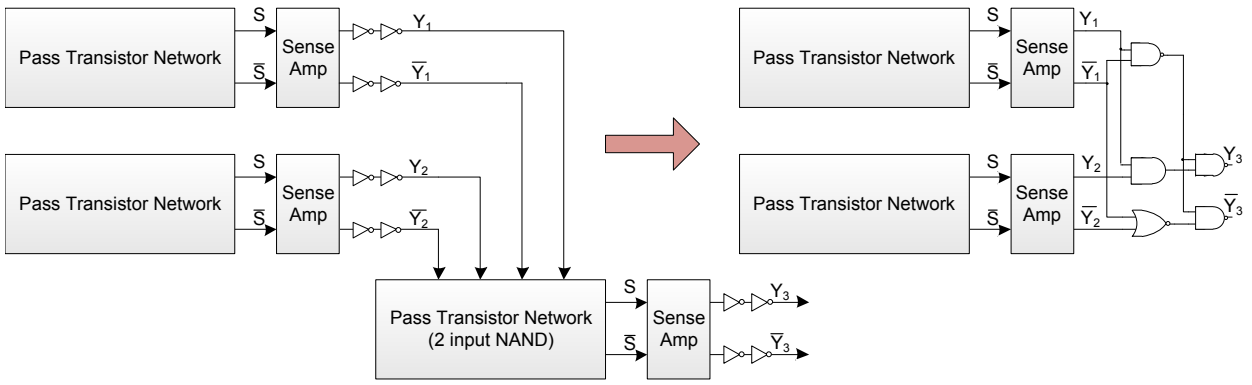

Figure A.6: SAPTL cells that implement very simple functions can be implemented as CMOS logic instead. The logic can be rolled into the output buffers, thus saving transistors.

A.5.3 Small Node Conversion to CMOS

Since SAPTL is hard-wired logic, additional static-CMOS gates can be inserted between SAPTL stages in order to reduce the overall node count. This differs from a FPGA, where all logic must be implemented using reprogrammable lookup tables or mapped to specific special-purpose gates. This useful for the many cases where the FPGA synthesizer creates an SAPTL cell for a very simple logic function. Figure A.6 illustrates this concept. Rather than create an entire SAPTL cell (including buffers, sense amp, driver, and pass transistor network), for a NAND2 gate, the logic is simply implemented using static-CMOS. Note that two additional NAND gates must be generated to ensure the fan-out pass transistor networks are drained after evaluation (see Chapter 5). By doing this, many transistors can be potentially saved. In the example in Figure A.6, the twelve inverters that are used to drive output loads are reduced to three NAND gates, a NOR gate, and an AND gate (which can also be sized to drive output loads). This is a reduction from 24 transistors to 22, just from the output buffers alone. The decision to convert small SAPTL nodes to CMOS should be based on whether or not the resulting CMOS logic has less area, delay, and/or energy than the circuitry needed to implement the SAPTL cell.

A.6 Conclusion

This appendix presented a method for synthesizing SAPTL circuits using an off-the-shelf FPGA synthesizer and a binary-decision-diagram (BDD) package. The FPGA synthesizer is capable of grouping together simple gates into larger cells with multiple inputs, which is a similar goal to what an SAPTL synthesizer would do. This results in a conservative estimate of the final delay and area of an SAPTL network. One major downside to this method is that FPGA synthesizers are ignorant of internal cell delay, due to the fact that the delay of FPGA LUTs are independent of the implemented logic function. This leads to large inefficiencies, thus the resulting delay can be considered an upper bound of SAPTL potential. Also presented were approaches that can be taken to improve SAPTL synthesis. This includes adding a cost function to the gate grouping algorithm that takes into account internal cell delay, modifying the BDD input reordering to account for delay, and converting small SAPTL cells to CMOS gates.

Bibliography

[1] L. Alarcon, T.-T. Liu, and J. Rabaey. Sense amplifier-based pass transistor logic (saptl), January 2009.

[2] D. Bakalis, K. Adaos, D. Lymperopoulos, G. Alexiou, and D. Nikolos. Eudoxus: a www-based generator of reusable arithmetic cores. In *Rapid System Prototyping, 12th International Workshop on, 2001.*, pages 182 –187, 2001.

[3] N. Binkert, R. Dreslinski, L. Hsu, K. Lim, A. Saidi, and S. Reinhardt. The m5 simulator: Modeling networked systems. *Micro, IEEE*, 26(4):52 –60, July 2006.

[4] R. Brayton and A. Mishchenko. Abc: An academic industrial-strength verification tool. In T. Touili, B. Cook, and P. Jackson, editors, *Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science*, pages 24–40. Springer Berlin / Heidelberg, 2010.

[5] G. K. Chen, D. Blaauw, T. Mudge, D. Sylvester, and N. S. Kim. Yield-driven near-threshold sram design. In *ICCAD '07: Proceedings of the 2007 IEEE/ACM international conference on Computer-aided design*, pages 660–666, Piscataway, NJ, USA, 2007. IEEE Press.

[6] D. A. Czaplewski, G. A. Patrizi, G. M. Kraus, J. R. Wendt, C. D. Nordquist, S. L. Wolfley, M. S. Baker, and M. P. de Boer. A nanomechanical switch for integration with cmos logic. *Journal of Micromechanics and Microengineering*, 19(8):085003, 2009.

[7] H. Dadgour, M. Hussain, C. Smith, and K. Banerjee. Design and analysis of compact ultra energy-efficient logic gates using laterally-actuated double-electrode nems. In *Design Automation Conference (DAC), 2010 47th ACM/IEEE*, pages 893 –896, 2010.

[8] H. F. Dadgour and K. Banerjee. Design and analysis of hybrid nems-cmos circuits for ultra low-power applications. In *Proceedings of the 44th annual Design Automation Conference*, DAC '07, pages 306–311, New York, NY, USA, 2007. ACM.

[9] R. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge. Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits. *Proceedings of the IEEE*, 98(2):253 –266, 2010.

[10] C. C. Enz, F. Krummenacher, and E. A. Vittoz. An analytical mos transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications. *Analog Integrated Circuits and Signal Processing*, 8:83–114, 1995. 10.1007/BF01239381.

[11] A. Fruehling, S. Xiao, M. Qi, K. Roy, and D. Peroulis. Nano-switch for study of gold contact behavior. In *Sensors, 2009 IEEE*, pages 248 –251, 2009.

[12] M. Guthaus, J. Ringenberg, D. Ernst, T. Austin, T. Mudge, and R. Brown. Mibench: A free, commercially representative embedded benchmark suite. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pages 3 – 14, dec. 2001.

[13] S. Hanson, M. Seok, Y.-S. Lin, Z. Y. Foo, D. Kim, Y. Lee, N. Liu, D. Sylvester, and D. Blaauw. A low-voltage processor for sensing applications with picowatt standby mode. *Solid-State Circuits, IEEE Journal of*, 44(4):1145 –1155, april 2009.

[14] S. Hanson, B. Zhai, M. Seok, B. Cline, K. Zhou, M. Singhal, M. Minuth, J. Olson, L. Nazhandali, T. Austin, D. Sylvester, and D. Blaauw. Exploring variability and performance in a sub-200-mv processor. *Solid-State Circuits, IEEE Journal of*, 43(4):881 –891, 2008.

[15] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose. Microarchitectural techniques for power gating of execution units. In *Proceedings of the 2004 international symposium on Low power electronics and design*, ISLPED '04, pages 32–37, New York, NY, USA, 2004. ACM.

[16] J. Jeon, V. Pott, H. Kam, R. Nathanael, E. Alon, and T.-J. K. Liu. Perfectly complementary relay design for digital logic applications. *Electron Device Letters, IEEE*, 31(4):371 –373, Apr. 2010.

[17] H. Kaul, M. Anders, S. Mathew, S. Hsu, A. Agarwal, R. Krishnamurthy, and S. Borkar. A 320mv 56 uw 411gops/watt ultra-low voltage motion estimation accelerator in 65nm cmos. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 316 –616, 2008.

[18] S. Kulkarni, D. Sylvester, and D. Blaauw. A statistical framework for post-silicon tuning through body bias clustering. In *Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on*, pages 39 –46, 2006.

[19] J. Kwong, Y. Ramadass, N. Verma, M. Koesler, K. Huber, H. Moormann, and A. Chandrakasan. A 65nm sub-vt microcontroller with integrated sram and switched-capacitor dc-dc converter. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 318 –616, Feb. 2008.

[20] A. Lungu, P. Bose, A. Buyuktosunoglu, and D. J. Sorin. Dynamic power gating with quality guarantees. In *Proceedings of the 14th ACM/IEEE international symposium on Low power electronics and design*, ISLPED '09, pages 377–382, New York, NY, USA, 2009. ACM.

[21] J. Maciel. Rf mems switches are reliable: a comprehensive technology overview. *MEMS Investor Journal*, 2009.

[22] N. Muralimanohar and R. Balasubramonian. Cacti 6.0: A tool to model large caches.

[23] R. Nathanael, V. Pott, H. Kam, J. Jeon, and T.-J. K. Liu. 4-terminal relay technology for complementary logic. In *Electron Devices Meeting (IEDM), 2009 IEEE International*, pages 1 –4, 2009.

[24] G. Palumbo and D. Pappalardo. Charge pump circuits: An overview on design strategies and topologies. *Circuits and Systems Magazine, IEEE*, 10(1):31 –45, 2010.

[25] R. Parsa, K. Akarvardar, J. Provine, D. Lee, D. Elata, S. Mitra, H. Wong, and R. Howe. Composite polysilicon-platinum lateral nanoelectromechanical relays. In *14th Solid-State Sensors, Actuators, and Microsystems Workshop*, pages 7 – 10, Jun. 2010.

[26] M. Putic, L. Di, B. Calhoun, and J. Lach. Panoptic dvs: A fine-grained dynamic voltage scaling framework for energy scalable cmos design. In *Computer Design, 2009. ICCD 2009. IEEE International Conference on*, pages 491 –497, 2009.

[27] J. Rabaey. *Low Power Design Essentials*. Series on Integrated Circuits and Systems. Springer, 2009.

[28] A. Raychowdhury, B. C. Paul, S. Bhunia, and K. Roy. Computing with subthreshold leakage: device/circuit/architecture co-design for ultralow-power subthreshold operation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(11):1213–1224, Nov. 2005.

[29] S. Roy, N. Ranganathan, and S. Katkoori. A framework for power-gating functional units in embedded microprocessors. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 17(11):1640 –1649, 2009.

[30] A. Sangiovanni-Vincentelli, A. El Gamal, and J. Rose. Synthesis method for field programmable gate arrays. *Proceedings of the IEEE*, 81(7):1057 –1083, jul 1993.

[31] N. Seki, L. Zhao, J. Kei, D. Ikebuchi, Y. Kojima, Y. Hasegawa, H. Amano, T. Kashima, S. Takeda, T. Shirai, M. Nakata, K. Usami, T. Sunata, J. Kanai, M. Namiki, M. Kondo, and H. Nakamura. A fine-grain dynamic sleep control scheme in mips r3000. In *Computer Design, 2008. ICCD 2008. IEEE International Conference on*, pages 612 –617, oct. 2008.

[32] M. Seok, S. Hanson, D. Sylvester, and D. Blaauw. Analysis and optimization of sleep modes in subthreshold circuit design. In *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, pages 694 –699, 2007.

[33] R. Swanson and J. Meindl. Ion-implanted complementary mos transistors in low-voltage circuits. *Solid-State Circuits, IEEE Journal of*, 7(2):146 – 153, Apr. 1972.

[34] A. Wang, B. Calhoun, and A. Chandrakasan. *Sub-threshold design for ultra low-power systems*. Series on Integrated Circuits and Systems. Springer, 2006.

[35] N. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison-Wesley Publishing Company, USA, 4th edition, 2010.

[36] S.-C. Wong, G.-Y. Lee, and D.-J. Ma. Modeling of interconnect capacitance, delay, and crosstalk in vlsi. *Semiconductor Manufacturing, IEEE Transactions on*, 13(1):108 –111, Feb. 2000.

[37] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45nm design exploration. In *ISQED '06: Proceedings of the 7th International Symposium on Quality Electronic Design*, pages 585–590, Washington, DC, USA, 2006. IEEE Computer Society.