Reducing Subthreshold Leakage Power Through Hybrid MOSFET-NEMS Power Gating

David G. Kindel

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

Master of Science
in
Computer Engineering

Leyla Nazhandali, Chair
Michael S. Hsiao
Wu-chun Feng

May 4, 2016
Blacksburg, Virginia

Keywords: NEMS, Power Gating, Low Power, Simulator, Computer Architecture

Reducing Subthreshold Leakage Power Through Hybrid MOSFET-NEMS Power Gating

David G. Kindel

Academic Abstract

Modern devices such as smartphones and smartwatches spend a large amount of their life idle, waiting for external events. During this time, they are expending energy, using up battery life. Increasing power consumption is a rising concern to users and researchers alike. Power gating, turning off a blocks of hardware when idle, reduces static power consumption. The Metal-Oxide-Semiconductor Field-Effect Transistors (MOSFETs) currently employed in processors leak current. Even in power gated circuits, MOSFET power gating may only save between 60-80% of power. A different type of switch, a Nanoelectromechanical Systems (NEMS) switch, presents an air gap between the source and drain while in the off state, eliminating subthreshold leakage current. The NEMS switch is slower to operate and only has a finite number of switching before breaking. They should be switched with caution. Proposed in this thesis is a hybrid power gating model wherein a MOSFET is placed in series with a NEMS switch. Power gating the Floating Point Unit (FPU) of a processor is studied through the use of modern open source computer architecture simulators. Each switch type is used to model power gating to observe energy savings and performance costs. The hybrid power gating model is more flexible across a variety of applications. Energy savings are comparable to single NEMS switch power gating for applications with low FPU activity. Any performance loss remains low, matching that of MOSFETs. Processor electrical costs are heavily reduced while devices remain operating at a near-optimal speed.

Reducing Subthreshold Leakage Power Through Hybrid MOSFET-NEMS Power Gating

David G. Kindel

General Audience Abstract

Modern devices such as smartphones and smartwatches spend a large amount of their life idle, waiting for external input. During this time, they are expending energy, using up battery life. The transistors that are inside of them, the minuscule electronics that make these devices work, are not perfect and "leak" current even when not in use. Another type of switch, a mechanical one, has been under development over the last decade. This mechanical switch is slower to operate and is not as reliable as current transistors yet yields a complete disconnection when turned off. Thus, no energy is wasted when a device is sitting idle. While this saves more energy, using a mechanical switch also has the potential to degrade a device's performance due to its slow operation. In this thesis, the effectiveness of combining the two types of transistors into one process is analyzed. The fast switching times of the currently used transistors can be used in situations where it is difficult to determine whether shutting down a piece of hardware is a good decision. If it has been determined that the circuit may be put to sleep for a long amount of time, the slower but more energy efficient mechanical switch may be used. With this hybrid operation, each transistor is only used in a mode that suits them most appropriately.

Contents

List of Figures

List of Tables

List of Abbreviations

ALU	Arithmetic Logic Unit
CIPC	Curved In-Plane Cantilevered
CMOS	Complementary Metal-Oxide Semiconductor
FLOP	Floating Point Operation
FLOPS	Floating Point Operations Per Second
FPU	Floating Point Unit
FU	Functional Unit
GALS	Globally Asynchronous/Locally Synchronous
ITRS	International Technology Roadmap for Semiconductors
MEMS	Microelectromechanical System
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistors
NEMS	Nanoelectromechanical System
NMOS	N-Channel MOSFET
PMOS	P-Channel MOSFET
SPDT	Single-Pole Double-Throw
SWCN	Single-Walled Carbon Nanotube

Chapter 1

Introduction

With the increasing popularity of multicore and manycore computer architectures and the ever growing number of transistors placed on a die, the industry has long since reached a time where power consumption of a processor must be taken into account. Processors currently contain more transistors than can be turned on at once. If it was not for these portions of "dark silicon," a processor could overheat and break. Even when not operational, transistors consume a small amount of leakage power.

Furthermore, the prevalence of mobile devices such as smartphones and smartwatches are rising. A commonality among these devices is that they spend a majority of their time in an idle state, waiting for external input from users or sensors. During this time, portions of the hardware may be safely put to sleep. With current implementations, sleeping may reduce power between 60 and 80%. The rest of the power is wasted. The user ends up

paying for the costs of wasted power in the form of reduced battery life in mobile devices or increased electricity usage. Mitigating this power consumption is a great concern to the modern computer architecture field.

There are many different ways currently implemented in today's technology to reduce power consumption. These focus on the two types: static power and dynamic power. Static power is power that is consumed due to the transistor simply existing on the die while powered on. If any current flows through a transistor, it consumes power since there is some resistance and a voltage drop across it, no matter the magnitude. Dynamic power, also called switching power, is the power consumed when a transistor makes a change from a logic level 0 to a logic level 1 or vice versa. For decades, dynamic power was the only type of power considered of any importance. The clock frequency of a processor had been steadily increasing up until the mid-2000s, increasing dynamic power. There were fewer transistors on a die, a higher transistor threshold voltage, and longer channel length, all of which ensured a low static power and the dominance of dynamic power. Since then, transistors have gotten smaller and more plentiful while the clock frequency has stayed the same. The rise in static power has shifted old perspectives. Without proper consideration for static power, researchers will be unable to shrink process sizes without overheating and destroying devices.

The research discussed in this paper addresses methods to further decrease static power consumed in modern processors. One of the most popular methods is power gating. Power gating is a power saving method wherein particular regions of a die are turned off when not in use. Ideally, without being connected to any source or ground, transistors within

this region will not consume power. This disconnection is performed by introducing what is known as a sleep transistor between either the source or ground line and the power gated region. This method is extremely effective in reducing static power. It could be better, however, since power cannot be completely removed with currently used MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors). There is still a physical connection between the source and the drain. The bulk of a MOSFET acts as a large resistor between the source and drain and some amount of leakage current flows through. That amount is often relatively small but it increases with the size of the transistor. With MOSFETs, some leakage power is unavoidable, despite many methods attempting to minimize the impact.

Mechanical switches are a proposed solution to the leakage problem. NEMS and MEMS (Nanoelectromechanical System and Microelectromechanical System) switches introduce an air gap between the source and drain that is closed when the transistor is switched on. This air gap provides an infinite resistance between the source and drain and provides no leakage current and no power leakage. From the perspective of saving power, this is a highly beneficial attribute. Contrariwise, every currently proposed type of mechanical switch has a longer latency and a very large turn on voltage relative to the voltages currently seen in a processor. Due to the mechanical nature, they also have a limited lifetime before they break. For these reasons, NEMS switches are firstly being considered as complements to CMOS (Complementary Metal-Oxide Semiconductor) transistors instead of replacements.

In analyzing the effects that NEMS have when compared to MOSFETs, modern applications, power gating schemes, properties of different switches, and the surrounding architecture

must be taken into account. The evaluation of these different switches is performed with consideration for each aspect.

1.1 Contributions

This thesis provides an analysis of power gating schemes with different sleep transistors. Analyses were performed by running various benchmarks on a common multicore system. The FPU (Floating Point Unit) was analyzed for various applications. The FPU is a component of the processor that is used infrequently for many complex operations but has a large area [15], consumes more power [11], and is more latent than other ALU functional units [4]. This analysis is performed using gem5, a cycle-accurate computer architecture simulator [4], and McPAT, a power, area, and timing simulator [15]. gem5 was modified to account for latencies of the sleep transistors and to provide different methods of power gating. McPAT was expanded to calculate dynamic and static power using gem5's runtime statistics and processor configuration.

All of these details were gathered to display a holistic view of how an FPU's performance, energy, and power consumption is affected by different sleep transistors. It is shown that with the proper power gating schemes, even the most latent NEMS switch has the potential to save energy, though often at the cost of its lifetime. These energy savings apply to a variety of benchmarks, even those designed to stress the FPU. In all of the proposed power gating schemes, the limited lifetime of a NEMS switch can also be controlled through changes

in hardware configurations.

Power gating of a processor core was also analyzed at a high level for various assumed activity levels and execution periods. This research provided information on how the length of idle periods and the activity frequency of sleep transistor switching affect the magnitude of energy savings and any performance drop.

The most important contribution in this thesis is the introduction of hybrid MOSFET-NEMS power gating. Hybrid power gating includes a shallow MOSFET sleep transistor and a deep NEMS sleep transistor. The low latency of the MOSFET is used to quickly power down and wake up the functional unit if the unit will not be asleep for a large amount of time. It helps to further reduce leakage while waiting for a more appropriate time to switch off the NEMS switch. This ensures that the NEMS switch is in use over long periods of time, increasing its effectiveness per sleep session. The lifetime of the NEMS switch is also preserved as it is not being switched as often.

It is shown that, for a certain benchmark and power gating scheme, a NEMS switch saves the FPU about 49.63% energy over no power gating and a PMOS header saves 36.45%. The hybrid model saves 38.14% energy compared to no power gating. This is done while the hybrid model matches the PMOS performance drop of less than 2% as opposed to the NEMS performance drop of nearly 5%. The hybrid model also lasts about 13 times as long as a standalone NEMS switch. These numbers have been gathered from a high FPU activity benchmark and aggressive power gating scheme. While other high activity benchmarks show that hybrid power gating can perform slightly worse for energy savings than a PMOS-only

implementation, low activity applications can receive immense benefits. The hybrid model is flexible in nature and accommodates any type of application a user may decide to run.

1.2 Organization

Beginning with Chapter 2, MOSFET and NEMS transistor background will be presented. This includes an analysis of their electrical properties. A general analysis of power and previous research in power gating wrap up this chapter.

Chapter 3 introduces specific properties of sleep transistors in power gating. It also introduces details of a hybrid MOSFET-NEMS power gating layout followed by equations used to gather power and energy numbers. Concluding Chapter 3 is a discussion on the testing environment, specifically with information on gem5 and McPAT parameters and the benchmarks used.

Developed power gating schemes are introduced in Chapter 4. Flowcharts are presented for each scheme that show how power gating decisions are made. This is done for single transistor schemes and a hybrid scheme.

Results are presented in Chapter 5 along with a discussion on their impact. These results are centered around how benchmarks, schemes, and sleep transistor types affect energy, performance, and NEMS lifetimes.

The work of this thesis is concluded in Chapter 6.

Chapter 2

Background

2.1 MOSFET Analysis

The MOSFET is the standard transistor used in modern processors. This type of transistor can be placed as a sleep transistor in power gating layouts. As such, it is necessary to understand its operation and electrical properties.

A MOSFET operates in one of three regions: subthreshold, linear, and saturation. Each region can be described by the drain to source current through the MOSFET in equation 1.

$$I_{DS} = \begin{cases} \mu C_{ox} \frac{W}{L} e^{\frac{V_{GS}-V_{th}}{(nV_T)}} (1 - e^{\frac{-V_{DS}}{V_T}}) V_T^2 e^{1.8} & V_{GS} < V_{th} \quad \text{(Subthreshold)} \\ \mu C_{ox} \frac{W}{L} [(V_{GS} - V_{th} - \frac{V_{DS}}{2}) V_{DS}] & V_{DS} < V_{GS} - V_{th} \quad \text{(Resistive)} \\ \mu C_{ox} \frac{W}{L} \frac{(V_{GS}-V_{th})^2}{2} & V_{DS} > V_{GS} - V_{th} \quad \text{(Saturation)} \end{cases} \tag{2.1}$$

In Equation 2.1, μ is the mobility of electrons if the MOSFET is n-type or electron holes if the MOSFET is p-type, C_{ox} is the gate oxide capacitance per unit area, W is the width, L is the length, V_{th} is the threshold voltage, V_T is the thermal voltage, and n is the subthreshold swing coefficient.

If a MOSFET is in the subthreshold, or cutoff, region, it is turned off and the channel between the MOSFET source and drain is only lightly populated with electrons (or electron holes). Only a very small amount of current is allowed to flow between the source and the drain. This subthreshold leakage current flows through the bulk of the MOSFET, which acts as a very large resistor. There is some subsequent voltage drop across the transistor. This subthreshold current is typically a small value for minimum sized MOSFETs but larger widths and smaller threshold voltages increase this current dramatically.

When $V_{GS} > V_{th}$ and $V_{DS} < V_{GS} - V_{th}$, the MOSFET operates in the linear region. In this mode, the channel between the source and drain conducts current. There is a smaller voltage drop across the source and drain but the current is directly related to that voltage drop as well as the voltage applied to the gate. As the gate voltage is increased, the amount of current flowing through the channel increases. The MOSFET acts as a resistor.

If V_{DS} is a large enough value, the MOSFET operates in the saturation, or pinch-off, region. In this space, the transistor is on and is highly dependent on the voltage applied at the gate but not the voltage drop across the MOSFET itself. Conduction is performed through a broad region and no longer through a narrow channel.

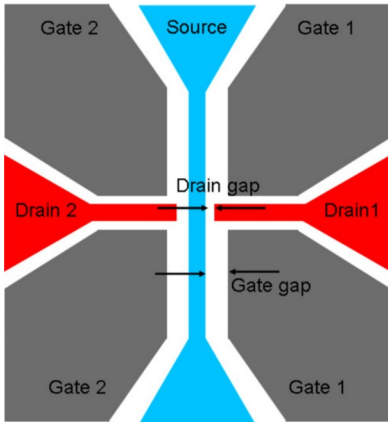


Figure 2.1: A Single-Pole Double-Throw NEMS switch.

2.2 NEMS Switch Analysis

NEMS switches are mechanical switches with a movable arm working as the source. A voltage applied to the gate influences the source to physically move and to make contact with the drain. When contact is made, the switch is in the on state. The on current is dependent on the resistance of the metal connection between the source and drain, though the on current consistently is reported to be lower than its CMOS counterpart [10]. A designer can control this resistance by changing the metal contact material [11].

An example of a NEMS switch from [5] can be seen in Figure 2.1. When a voltage is applied to a gate, the source beam will deform. At a large enough voltage (actuation voltage), the beam deforms the distance of the drain gap and makes contact with the drain. At this point, the connection is made and current flows.

The physical connection makes them unique when compared to other modern transistors employed in the field today. They can be used to make unique simple gates that can be

used in conjunction with CMOS circuitry [16]. Additionally, they make particularly good candidates for sleep transistors for power gating circuitry. Due to the air gap between the source and drain, Off state leakage reports have been at the bottom range of measurement instruments, in the femtoamp range [11, 7, 14]. The only current through the off state is the vacuum tunneling effect and Brownian motion displacement [6], effectively making the leakage current zero when compared to CMOS technologies. The presence of the air gap dramatically changes the effect the transistor has on the system. With no leakage current, there will be no leaked power through the block and, likewise, no energy consumed.

NEMS switches have been previously built so they can be fabricated on the same die as traditional CMOS logic [11, 5]. Because of this, special production for NEMS is not necessary and it is attractive for fab houses and designers looking to make use of NEMS alongside CMOS logic.

There are some properties of NEMS that researchers report make them difficult to include in processors. The largest issue facing NEMS is that they have a limited lifetime. They can only be switched a finite number of times before the switches are effectively broken. They can break due to general wear and tear, burn-out, or stiction. The lifetime of NEMS vary by the process type and even within the process itself. Variables affecting how long a NEMS will last include the material used, the thickness of the beam, and the gate actuation voltage [16, 26].

The actuation voltage also affects the usefulness of NEMS. Most low power devices that can benefit from a NEMS switch will not have the larger voltage source necessary to charge

a NEMS gate to actuation. Typical processors operate around 1V and as the process size scales down, the voltage scales down as well. To reach the necessary actuation voltage, a charge pump must be placed on the die. When used in conjunction with a capacitor to store charge, the charge pump consumes a comparatively low amount of power [11].

2.3 Power Analysis

Regardless of its mode of operation, there is a current and a resistance through a MOSFET. Therefore, whether active or off, it consumes power. The mode of operation determines how much current flows. Static and dynamic power may be consumed at any point. Static power refers to power consumed when a transistor is in a steady state, when there is no change to the voltage applied to the transistor at any of the nodes. Dynamic power, often called switching power, refers to the power it takes for a transistor to switch from its off state to on state and vice versa.

$$P = P_{static} + P_{dynamic} \tag{2.2}$$

$$P_{static} = I_{DS}V_{DS} \tag{2.3}$$

$$P_{dynamic} = CV_{DS}^2 f \tag{2.4}$$

In Equation 2.3, it's shown that the static power can be calculated by the current flowing through the transistor and the voltage drop across it. In the subthreshold region, this is particularly important since every transistor will have some small current and potentially

large voltage drop. The cumulative static power is the summation of every individual transistor's static power seen in Equation 2.5. The same is true for dynamic power, as seen in Equation 2.6.

$$P_{static,total} = \sum_{i=1}^{n} P_{static}(i) \tag{2.5}$$

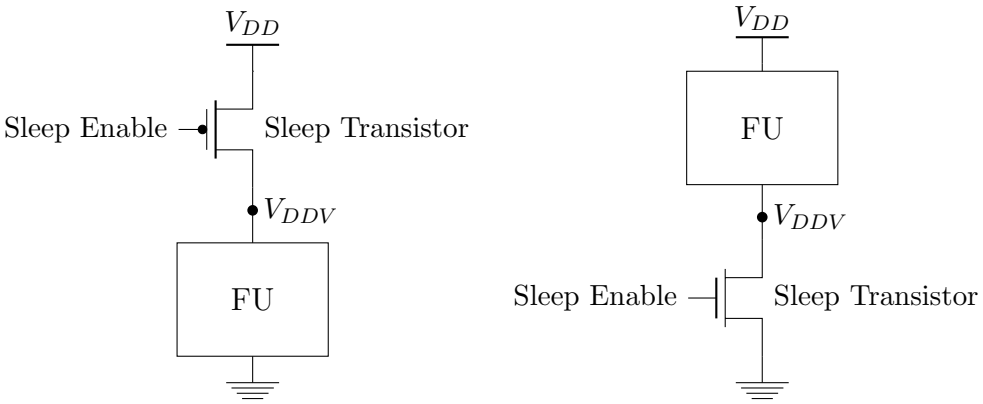
$$P_{dynamic,total} = \sum_{i=1}^{n} P_{dynamic}(i) \tag{2.6}$$

When comparing how well two entities perform over time, it may be best to consider energy instead of power. Calculating an application's energy savings takes its time to execute into account as well. This is shown explicitly in Section 3.1. A lower power consumption does not guarantee lower energy consumption.

$$E = \int_0^T P dt \tag{2.7}$$

2.4 Power Gating

Power gating is a method currently employed by computer architects to limit the amount of power consumed by a region of transistors on a die when is is not in use. A sleep transistor may be placed in between a logic block and the V_{DD} line as a header switch or between the logic block and ground as a footer switch, as seen in Figure 2.2. In either case, when the switch is moved into the subthreshold region, the circuit is effectively left open and no current should flow through the entire logic block, cutting static power. However, as discussed in



(a) Header Power Gating (b) Footer Power Gating

Figure 2.2: The layout of a header and footer sleep transistor used for a power gated functional unit. The voltage drop across the functional unit is shown as V_{DDV} (V_{DD} virtual). The voltage drop across the sleep transistor is V_{DS} where $V_{DS} = V_{DD} - V_{DDV}$.

Section 2.1, a MOSFET cannot completely eliminate static power because there is always some leakage between the source and drain through the bulk. These sleep transistors leak as well and become inefficient when gating a large macro. Both the voltage drop and the subthreshold leakage current through the sleep transistor must be analyzed to make effective use of power gating. These values are limited by many factors discussed in Section 3.1.

Many power gating techniques have been previously proposed including sleep transistor design [20, 24], sleep transistor placement and layout [17], and power gating schemes [23, 12]. Most research has been performed on traditional P-Channel and N-Channel MOSFETs. These are easy to size and easy to arrange on a die since designers already have thoroughly tested processes available for these types of transistors. PMOS transistors tend to be less leaky than NMOS, mainly due to the mobility of electrons as opposed to the smaller mobility of holes, but NMOS often consume a smaller area. For systems where area is a strict concern, NMOS may be employed since any amount of power gating is often desirable. However, if

area is not an issue, of the two transistors, PMOS is likely the candidate of choice.

There have been developments in recent years in the area of NEMS switches that reduce their activation voltage and increase their overall lifetime. NEMS switches have never been more primed for use in power gating and they will continue to get better as research continues. The motivating factor for the use of NEMS switches in power gating schemes is the infinite off resistance due to the air gap between the source and the drain. While MOSFETs leak current over time even when the circuit isn't in use, a NEMS switch will have effectively no current flowing and therefore will consume minimal power.

The latency involved with activating a NEMS switch drastically reduces its effectiveness, however. Considering an execution unit of a processor, if power gating is applied, the rest of the pipeline might require that data before it is able to proceed. The longer a unit is stalling on a piece of data, the less useful work it is doing. At a certain point, energy consumption might get worse with power gating than without as it could take considerably longer to execute a program's instructions. It is crucial to ensure that when a processor is asleep, it will be asleep long enough to have an overall gain in energy savings. Power Gating Schemes help to achieve this goal and each are designed using different methods to maximize energy savings with respect to performance and NEMS lifetime. The particular schemes developed for this research in power gating are described in detail in Chapter 4.

If a net energy savings has not been seen then a break-even point has not been reached. The break-even point is defined as the point in time where the total energy saved by turning off a unit is greater than any dynamic energy consumed by shutting down and waking up a

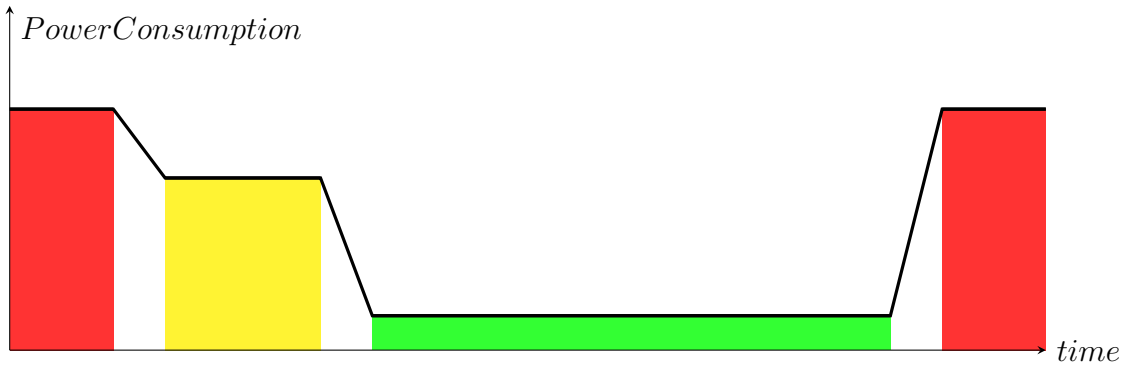


Figure 2.3: A timeline of a functional unit in a traditional power gating scheme. There are 3 periods of note: active (red), idle (yellow), and asleep (green).

unit or static overhead energy from the power gating hardware itself. [12] models the break-even point of a MOSFET sleep transistor for execution units. This is presented with typical MOSFET parameters. [11] presents a model for the break-even point of NEMS switches. According to these two resources, the break-even point of MOSFETs is typically around 10 cycles while the break-even point of NEMS switches ranges between tens and hundreds of cycles, depending on the specific switch type. A MOSFET switch is more efficient for fast switching and short periods of inactivity whereas NEMS switches will typically only be viable for periods of longer inactivity.

A given power gating timeline might look similar to Figure 2.3. When a functional unit is active, it operates with a certain dynamic and static power, consuming a large amount of power. If idle, it no longer consumes dynamic power but still consumes static power, which can still be very large. As noted in Section 1, this is only expected to get worse. In periods such as these, a unit may decide to sleep. In sleep mode, there is the potential to save vast amounts of energy as the power consumption is reduced closer to the floor.

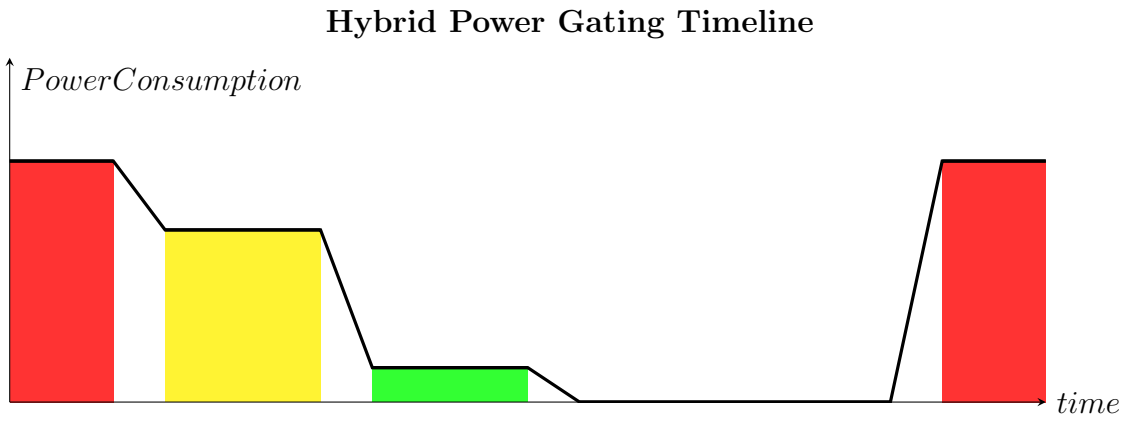


Figure 2.4: A timeline of a functional unit in a hybrid power gating scheme. There are 4 periods of note, as opposed to 3 in a traditional scheme: active (red), idle (yellow), and asleep (green), and deep asleep (at floor).

There is room to improve the benefits of power gating through the infinite off resistance of a NEMS switch. Those benefits are shown in Figure 2.4. Instead of keeping the off power close to the floor, the NEMS switch reduces the power consumption to 0 and the circuit is truly open. This type of power gating has the potential to be extremely beneficial for applications with large gaps in execution while maintaining the flexibility to save power in high activity applications. Applications see near-minimal performance drops regardless of its activity with hybrid power gating. If an idle period exists that is deemed too short for a NEMS switch to take advantage of, the unit remains in a shallow sleep instead of moving into a full deep sleep. This will provide the unit a reduced performance drop and will still have reached its break-even point. Mispredictions on deep sleep sessions can still be a concern, however, and it is the role of the power gating scheme to minimize them.

2.4.1 Power Gating Granularity

Power gating can be done in two different levels of granularity, either coarse or fine-grained. Fine-grained power gating adds sleep transistors to small logic cells on a die. This can be more costly in terms of area overhead and introduce a high IR drop across the sleep transistor. This is often seen at the level of functional units, memory cells, etc. Coarse-grained power gating is done at a broader level and is often done for large blocks such as whole processor cores.

Fine-grained power gating is often more complex and is integrated into the design of the cell itself. Integrating the power gating with the logic cell becomes expensive to design and construct since more detailed knowledge of the unit in question is often necessary. Specific details of the macro are needed to make the power gating function effectively. If done properly, it is more beneficial in terms of power and energy. Units can be asleep for shorter periods of time before a break-even point is reached since the capacitance of smaller units is inherently lower and it takes less energy to wake up or shut down. Likewise, they have shorter wake up and shut down latencies. The entire impact of shutting down a unit is not as costly as it would be in a coarse-grained setup.

Since power gating schemes timing decisions can be tailored to the unit being power gated, the schemes can be designed such that they turn off optimally based off of both seen and expected execution patterns. This can be set to dynamically predict through predictive hardware and statically by setting the sizes of counters or other properties of a scheme.

Performing fine-grained power gating also allows all units to work independently. Gated units can make sleep decisions based off of their own workload and other functional units are not affected by a sleeping unit for anything but normal data dependencies. That is, if an FPU is used infrequently but the integer ALU is highly utilized, the FPU power gating is not bound to the execution of the ALU. The integer ALU may continue execution while the FPU is put to sleep. When considering coarse-grained power gating, a processor core as a whole will be shut down instead of selected pieces. This fails to take advantage of many potential idle periods.

Fine-grained power gating with respect to PMOS, NEMS, and the combination of the two is analyzed in the majority of this research. Coarse-grained power gating is also analyzed, but is monitored at a high level with respect to potential loads instead of specific execution patterns based on experimental data. Coarse-grained analyses are done with respect to gating a single core of a multicore architecture. The differences between MOSFETs and NEMS switches drastically affect how each switch performs in power gating at each level.

Power gating the core of a processor can have some different consequences than the relatively fine-grained power gating seen for the FPU. These consequences are more severe for cores due, mainly, to a much larger total capacitance. A core has a larger capacitance because of its large area and greater number of transistors. A larger total capacitance means that the core will take longer to charge when woken up, thus reducing performance. Furthermore, the dynamic power consumption is larger. As discussed in Section 2.3, dynamic power is directly related to the capacitance of the unit being switched. The transistor that is used

for gating the unit also needs to be larger, as more current is required to flow through a core than a functional unit. Power calculations from Section 3.1 show that a larger transistor is also required, yielding a larger dynamic power due to sleep transistor switching as well.

The larger power consumption and worse performance mean that switching a core comes at a much higher cost than switching a smaller functional unit. Putting a unit to sleep too often could quickly result in a severe negative performance impact and consume more energy than not power gating. As will be seen in the coming sections, this reaction can occur with MOSFET, NEMS and hybrid power gating.

Chapter 3

Methodology

This chapter will discuss the methods by which results were gathered. Modern architectural simulators were used to analyze specific power gating sleep transistors and power gating schemes in terms of their performance and energy savings. Multithreaded applications were run against these simulated multicore architectures. Properties of the specific types of sleep transistors were configured in each power gating scheme. MOSFET properties were based on ITRS (International Technology Roadmap for Semiconductors) data built into the McPAT simulator [15] and NEMS switch properties were based on various recent publications.

One of the most important qualities of the sleep transistor is the ability to allow for the maximum possible current to flow when the unit is on while also reducing both the voltage drop across the transistor and the subthreshold leakage current through the transistor. There are models available and research performed for discovering these tradeoffs in a MOSFET

through sizing [21, 13]. NEMS switches, however, are more complicated. There are many different types of NEMS switches, expected values are often different from measured values, and processes are evolving as more research is performed [16, 5]. These properties of sleep transistors are discussed in Section 3.1.

Differences between traditional MOSFETs and newer NEMS switches were monitored using active open source simulators gem5 [4] and McPAT [15]. gem5 is a cycle-accurate computer architecture simulator capable of running benchmarks built for specific ISAs against customized architectures or those taken straight from examples. Multiple heterogeneous cores can be implemented, each with different pipelines and hardware complexity if desired. Statistics are reported at the end of executions detailing the performance of the system as a whole along with individual pieces of hardware. Customized statistics can also be implemented in the system in order to gather more specific details about the execution of any piece of the architecture.

Unfortunately, gem5 currently does not have a power estimator built in. Results were funneled into McPAT, a power, area, and timing simulator, to judge power efficiency with respect to the hardware and runtime performance seen in gem5. McPAT reports both dynamic and leakage power in a hierarchical form, from basic functional units to the entire processor.

The Floating Point Unit (FPU) of a processor was chosen as the unit to be studied for power gating. In practice, the FPU is an optimal candidate. Many common programs tend to operate on integer units wherever possible to avoid the unfortunate latencies that floating point operations incur. This results in many extended idle gaps in execution time that

can be taken advantage of in power gating. During this time of no useful work, the FPU might be successfully gated to reduce most of its static power. Scientific applications operate on the opposite spectrum and often use floating point execution for complex mathematical operations. In the worst case scenario, if a power gating scheme inappropriately shuts down a unit early and often, scientific applications might even consume more energy and take longer to execute. Using these tools, the performance of different types of applications can be seen with regards to different power gating configurations.

In order to accurately simulate how our chosen benchmarks might perform in real execution environments, a consistent processor model was used across all tools. Both gem5 and McPAT provide built in support for an ARMv7-a profile model of the ARM architecture. The model also includes definitions for an out of order model processor with all instructions in the ISA and a group of functional units with appropriate latencies. Specifically, support is included for the ARM VFP FPU and the NEON extension.

3.1 Sleep Transistor Properties

Sleep transistors are designed as header switches for this research. This is due mainly to PMOS headers being less leaky, though typically larger, than their NMOS counterparts. In NEMS switch analysis, using it as header or footer does not make a difference.

There are two methods used to place a header into a circuit. The first method introduces a larger voltage at the source to compensate for the voltage drop across the sleep transistor.

The second method keeps the voltage drop of the entire system, sleep transistors included, at the original V_{DD}. There are certain performance, power, and complexity tradeoffs that must be considered when designing for acceptable electrical properties surrounding sleep transistors.

If a higher voltage is applied to the source, the attributes of the macro are all kept identical. Performance is not affected and the macro's static and dynamic power consumption remains the same. The sleep transistor may also be sized, placed, and routed independently of the macro. There is a larger voltage drop, $V_{DS} + V_{DDV}$, across the sleep transistor and functional unit that results in a larger total power consumption. A larger power means this design has the potential to be a less than ideal configuration.

If the voltage V_{DD} remains the same, the IR drop across the header will determine how fast the functional unit can operate. Assuming the FPU is in the critical path, there exists a performance drop across the entire system. Given an allowable performance drop for the system, Equation 3.1 shows how the desired voltage drop can be calculated. Typical allowable performance drops are 5% or 10%.

$$V_{DS} = PerformanceDrop * (V_{DD} - V_{th}) \tag{3.1}$$

Unless operating in a Globally Asynchronous/Locally Synchronous (GALS) environment [25], the performance drop determines the system clock frequency. Due to the design overhead and second order effects of introducing a GALS system, the system clock has been reduced for simplicity. The magnitude of the reduction is simply Equation 3.2.

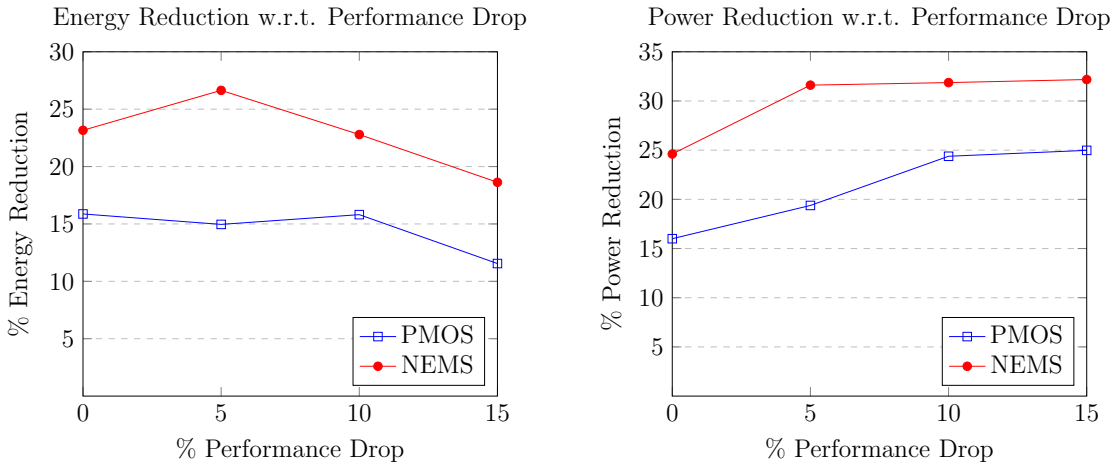


Figure 3.1: Relationship between allowable performance drops and power/energy: Energy and power are concerns when modifying V_{DS} and/or V_{DD} to fit a particular allowable performance drop. If performance is decreased too far, the amount of saved energy begins to fall as well despite power savings still being improved.

$$NewClock = OldClock * (1 - PerformanceDrop) \tag{3.2}$$

There is no clear answer for an optimal universal value for the allowable performance drop or if any performance drop should be tolerated at all. It is possible, however, to gauge the effectiveness of these differences by running a benchmark against the different systems with a chosen power gating scheme. Figure 3.1 provides an idea of how these executions look in terms of performance, energy, and power savings. This analysis is performed for both a chosen NEMS switch and PMOS header. Both have some effect on the performance drop and result in a certain amount of energy/power savings.

In this aspect, energy is a concern. Despite a system performing slower to save power, the energy savings was hindered. Equation 2.7 shows that the longer a unit is active, the more energy it consumes. Even if power is reduced by a large margin, a system might perform

so slowly that it is no longer beneficial to reduce power. Figure 3.1 shows this in regards to our PMOS and NEMS headers. Due to this relationship between power and time, a unit that performs longer at a lower power might not necessarily operate more efficiently. Energy and time are taken into consideration for the tests in Chapter 5. Power simply influences energy consumption and is not entirely indicative of the utility of power gating and thus is not considered on its own.

In Figure 3.1, the points with the best performance (0% performance drop) have been supplied an additional voltage at the source of the header such that $V_{DS} = 0.1$ and $V_{DDV} = 1$. While there are severe relative differences in power savings, the energy savings between the two methods of power gating are quite comparable. Because of this, and to avoid any secondary effects of slowing the entire system and reducing the voltage supplied to the FPU, an additional voltage was supplied to the power gating block for all tests.

3.1.1 MOSFET Header Analysis

The sizing of the sleep transistor is critical to the MOSFET effectiveness in power gating. Sizing is performed by attempting to keep the off current as small as possible while still providing enough on current to the functional unit to ensure its normal operation.

The peak "on" current and typical MOSFET properties are known through executions of McPAT for a particular process (45nm High Performance embedded ARM processor). Using these values, the linear equation from Equation 2.1 can be used to size the transistor. A sleep

transistor operates in the linear region when switched on. The MOSFET size can be found by rearranging the linear region current equation into Equation 3.3. The current through the MOSFET in its off state can be found by inserting the $\frac{W}{L}$ ratio into the subthreshold formula in Equation 2.1.

$$\frac{W}{L} = \frac{I_{on}}{\mu * c_{ox} * [(V_{GS} - V_{th} - \frac{V_{DS}}{2}) * V_{DS}]} \tag{3.3}$$

A certain voltage, $V_{DS} + V_{DDV}$, was chosen to be applied to the header where V_{DS} is the voltage drop across the header and V_{DDV} is the original macro V_{DD}. This V_{DS} was some value smaller than V_{DDV}. From Equation 3.3, as V_{DS} becomes small, since a constant I_{on} is desired, the $\frac{W}{L}$ ratio will inevitably be larger. This causes I_{off} to become larger as well. If the opposite occurs, V_{DS} is large, $\frac{W}{L}$ becomes smaller and the subthreshold current is minimized.

While it is impossible to wholly tell which option is best for a system, a reasonable one can be chosen based on empirical evidence guided by Equation 2.1. When V_{DS} is low, the voltage overhead is reduced so both static and dynamic power are reduced in normal operation but static power in the off mode is increased. The effects of a larger V_{DS} is reversed. A larger V_{DS} consumes more static and dynamic power during normal operation but will consume much less static power if the unit is switched off. A value for V_{DS} must be chosen that attempts to balance power consumption across a variety of workloads and that will capture a net savings due to power gating.

To discover that value, tests were run to look at how a sample execution might look for

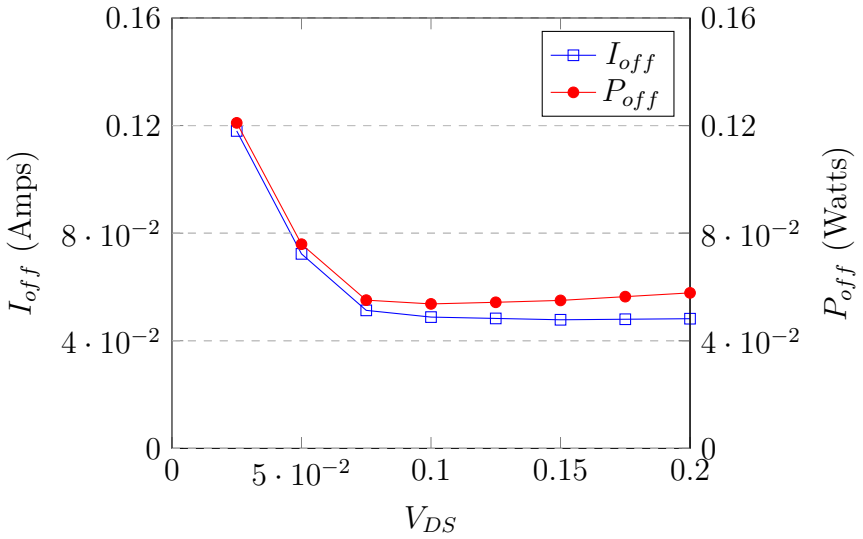


Figure 3.2: The effect of changing the sleep transistor V_{DS}: V_{DS} has a very direct affect on the circuit. A larger V_{DS} will create a smaller $\frac{W}{L}$ ratio and a smaller subthreshold leakage but at a point, the returns decrease and both power and current increase. In this scenario, $V_{DD} = 1$.

varying values of V_{DS}. The resulting graph, Figure 3.2, looks vaguely parabolic in shape for off power. At a certain threshold, the voltage became very large and current stopped reducing. Power began to climb higher at that point. A value in the middle appeared optimal for this configuration and $V_{DS} = 0.1$ was the best choice of those tested. This ultimately became the V_{DS} used in all single-transistor future tests.

3.1.2 NEMS Header Analysis

The NEMS switch is variable in nature. Many different types of NEMS switches exist [16] and even within those processes, what is constructed is can be very different from what is theorized [5]. In choosing a NEMS switch, there are often different lifetimes, actuation voltages, fabrication difficulties, and switching latencies, each of which must be taken into

Table 3.1: NEMS switch properties

NEMS type	Lifetime (switches)	Activation Voltage	Switching Speed	Notes
Single-Walled Carbon Nanotube	10^{12}	6V	10ns	[18][22][19]
Curved In-Plane Cantilevered Thin Film	10^{8}	10.3V	25ns	[8][1]
Single-Pole Double-Throw	10^{9}	5V	41ns	[11][5]
Near Ideal	10^{12}	1V	1ns	

account when designing a circuit. A representative collection of different NEMS switches and their properties are shown in Table 3.1. Every NEMS switches was assigned a specific lifetime, activation voltage, and switching speed based off of prior research. These numbers attempt to present an average case for different NEMS switch types.

Compared to MOSFETs, NEMS switches require a much higher voltage needed to switch. This is due to the physical movement required to make the source beam deform to the drain. On virtually every modern processor, this higher voltage will not be readily available. A charge pump must be placed next to the switch in order to provide a voltage source for the NEMS switch to use. According to [11], these pumps consume a proportionally low amount of energy when compared to the rest of the system. As such, charge pump analysis is not studied further.

The voltage drop, V_{DS}, is also assumed to be the same value as is set for the PMOS implementation. This was decided because a NEMS resistance is variable due to placement and routing of a the switches, the metal contacts of the switches, and the process variability. Different processes might use different metals that provide a variety of resistances [16]

and the metals themselves can be changed on a NEMS switch [11]. For these reasons, the assumption is made to maintain the same V_{DS} as the MOSFET transistor for simplicity and more accurate comparison of each switch.

In order to physically move the source to make contact with the drain, it takes a certain amount of time. This switching time is often much larger for any NEMS switch than it is for a MOSFET and it introduces a much longer latency. It is critical that a unit is not shutting down inappropriately or too often. If it is, system performance could be severely hampered. The next section will show that the hybrid power gating model alleviates some pressure on the NEMS switch in its power gating scheme. If wrongly shut down, the wake up time will be shorter and the lifetime of the NEMS switch extended.

3.2 Hybrid Power Gating

The goal of a hybrid power gating layout it to take advantage of the low latency switching and infinite lifetime of a MOSFET transistor as well as the very low off current of a NEMS switch. A PMOS header is placed in series with a NEMS switch as in Figure 3.3. In this arrangement, the shallow sleep transistor (PMOS) and deep sleep transistor (NEMS) have separate sleep enable lines and can be shut off independently of each other.

The power gating scheme in Section 4.2 attempts to save overall energy consumption by powering down the PMOS transistor when short idle periods are encountered and turning off the NEMS switch for longer ones. This helps alleviate the two largest drawbacks of a

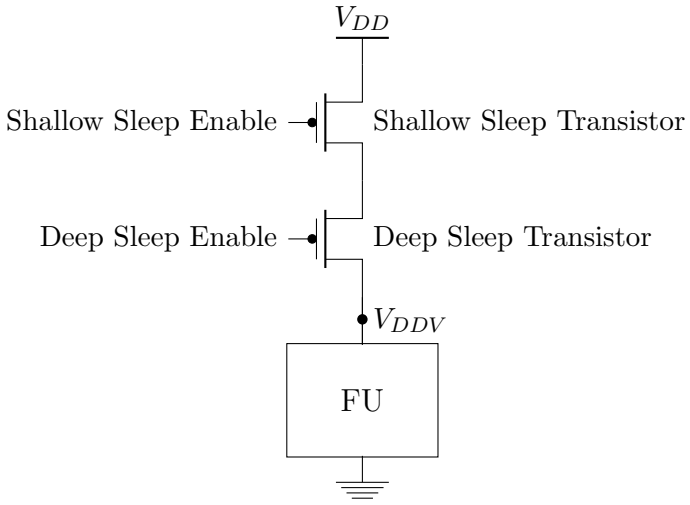


Figure 3.3: Hybrid power gating layout: Sleep transistors gating a functional unit in a hybrid scheme are placed in series. The power gating scheme will send a sleep enable signal to both shallow and deep sleep transistors. A MOSFET will likely be the shallow transistor and a NEMS will be the deep transistor.

NEMS switch, the long latency and limited lifetime, while still power gating during as many idle cycles as possible.

Due to having two transistors in series, the collective voltage drop across them is judged similarly to Section 3.1. Doubling the collective V_{DS} to 0.2 instead of 0.1 as it is in a single transistor layout will create a considerably larger voltage drop across the entire unit. Alternatively, each transistor's V_{DS} can be halved so the entire voltage drop, V_{DD}, remains the same. Choosing which V_{DS} to apply heavily affects the leakage power consumed by the MOSFET in its off state.

Figure 3.4 shows the effect that different types of voltage drops have on the overall energy consumption. The conclusion is that a lower V_{DS} is more beneficial across the sleep transistors. With a low V_{DS}, more energy is saved. This is true across multiple different power

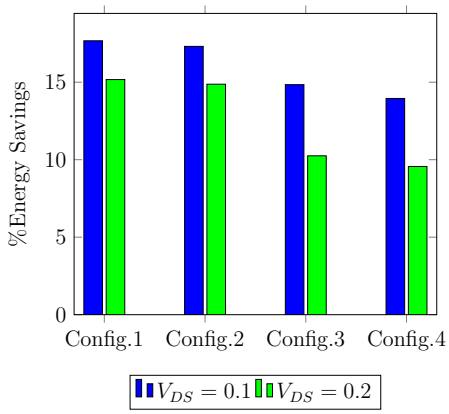


Figure 3.4: Differences in the hybrid scheme V_{DS}: Executions of PARSEC's Swaptions benchmark for different power gating configurations with a NEMS switch. V_{DS} is the collective voltage drop across both sleep transistors.

gating configurations. Configurations of power gating schemes are expanded upon in Chapter 4. The reason a lower V_{DS} is desirable is largely due to two occurrences. The first is that there are short idle periods in the shallow sleep state. Any increase in the subthreshold leakage current for the shallow sleep state have a greatly reduced effect on the overall power consumption in a hybrid model. The second occurrence is that a larger V_{DD} increases the static power consumption when a functional unit is awake. The zero off leakage of the NEMS switch allows V_{DS} to remain low.

3.3 Calculating Power and Energy

In order to find the energy savings of a floating point functional unit in the face of a hybrid power gating scheme, details of the execution traces are required. A new average static power consumption can be found by Equation 3.4. P_{static} is the static power consumption

with respect to the new voltage drop, V_{DS} over the sleep transistor(s). P' and P'' are the static power consumption of a shallow power gated transistor and a deep sleep transistor, respectively. s and d are the number of cycles in the shallow sleep and deep sleep states, respectively, and t is the total number of cycles of application execution. For a NEMS switch, the static power consumption for its power gated region is 0 and for any power gating scheme not operating in a hybrid model, $d = 0$ as well.

$$P_{static,new} = (P_{static} * \frac{t - s - d}{t}) + (P' * \frac{s}{t}) + (P'' * \frac{d}{t}) \tag{3.4}$$

Alongside a reduced static power, power gating introduces increased dynamic power consumption due to the switching of the sleep transistor. Equation 3.5 is used to find the total additional dynamic power by multiplying the number of sleep transistor transitions from logic level 0 to a logic level 1 and vice versa by the energy consumption per wakeup. The energy consumption per wakeup is the combination of the energy it takes to switch the sleep transistor and to charge or discharge the power gated macro.

$$E_{dyn,new} = E_{wakeup} * n \tag{3.5}$$

$$P_{dyn,new} = P_{dyn} + \frac{E_{dyn,new}}{time} \tag{3.6}$$

Finding the new dynamic power consumption is done by using the new found energy and dividing that by the total execution time, in seconds, and adding it to the dynamic power consumption of the FPU as a standalone unit. The dynamic power due to the unit be-

ing power gated is a value provided by McPAT. McPAT provides power numbers for the embedded FPU being analyzed as well as the processor core for the coarse-grained analysis.

With the new total power, the addition of the new dynamic and static powers, each single execution can be compared to any other execution, whether power gated or not. Tests in Chapter 5 use this to compare power gating effectiveness.

3.4 Test Environment

All tests were performed by running gem5, a cycle-accurate computer architecture simulator, for a specific architecture and power gating scheme. ARM benchmarks were compiled and loaded into a customized disk image to be loaded into the gem5 environment. gem5 also loaded a linux kernel, version 3.13.0-rc2, with which to make system calls. These system calls include the use of the pthreads library in order to make use of multiple processor cores defined by gem5. While gem5 also has ways of mimicking a thread scheduler and all system calls, it was decided to use the linux kernel to give more realistic credence to the results.

3.4.1 Benchmarks

Benchmarks were chosen from two different suites. The main one that was analyzed was the PARSEC Benchmark Suite [2]. Applications were chosen to attempt a worst case analysis for high performance computing. Each application stresses the power gating hardware to

heavily test the NEMS switch reliability and efficiency.

The second suite chosen was the MiBench embedded systems benchmark suite [9]. Since the MiBench suite was built for embedded systems, applications included in this would more closely match what applications can be seen in low power systems and those that could benefit from NEMS switch power gating. Monitoring these types of applications and their feasibility is critical to building a realistic notion of the utility of our hybrid model.

PARSEC

Benchmarks were chosen from the PARSEC Benchmark Suite version 3.0 and compiled for the ARM ISA. PARSEC is comprised of multithreaded applications that are diverse and capture execution patterns of emerging workloads. Specific applications were chosen to get a set of benchmarks that heavily utilize the FPU. This enables the monitoring of a "worst case" scenario where power gating mispredictions may be common.

The Swaptions, Blackscholes, and Ferret benchmarks were chosen specifically from the suite in order to see how the devised power gating schemes operate with a large percentage of FPU operations. The Freqmine benchmark was also chosen in order to test how a baseline application with nearly no FLOPs (Floating Point Operations) operates against each scheme. According to [3], for the large simulation working set developed by PARSEC developers, the values in Table 3.2 are expected. The small working set was used in order to reduce gem5 execution time. It was anticipated that large and small sets would include a comparable

Table 3.2: PARSEC instruction makeup

Benchmark	**Instructions (Billions)**		**%FPU**
	Total	FLOPS	
Blackscholes	2.67	1.14	42.69%
Swaptions	14.11	2.62	18.56%
Ferret	23.97	4.51	18.81%
Freqmine	33.45	0.00	0.00%

Table 3.3: PARSEC measured FPU activity

Benchmark	**Cycles (Millions)**		**%FPU**
	Total	FPU Busy	
Blackscholes	99.7	24.5	24.61%
Swaptions	240	42.2	17.60%
Ferret	445	92.6	20.83%
Freqmine	646	0.00	0.00%

number of FLOPs. Results of FPU activity times were tracked and are presented in Table 3.3. Each benchmark, except Freqmine, is heavily affected by power gating due to their high FPU usage. Each are highly susceptible to different FLOP execution patterns. Because of their high FPU activity, if benchmarks such as these benefit from power gating with minimal performance issues, many applications that users might encounter can benefit as well.

One aspect of each application that is not taken into account in Table 3.3 is the ordering of instructions. Even with a high overall density of FPU accesses, some periods of time may witness higher activity than other periods. The low activity periods are more amenable to power gating while periods of high activity are not. In periods of high activity, controlling sleep sessions is critical. Increased sleep transistor switching yields worse performance, increased dynamic power, and for a NEMS device, decreased lifetime, though there is also a potentially lower static power.

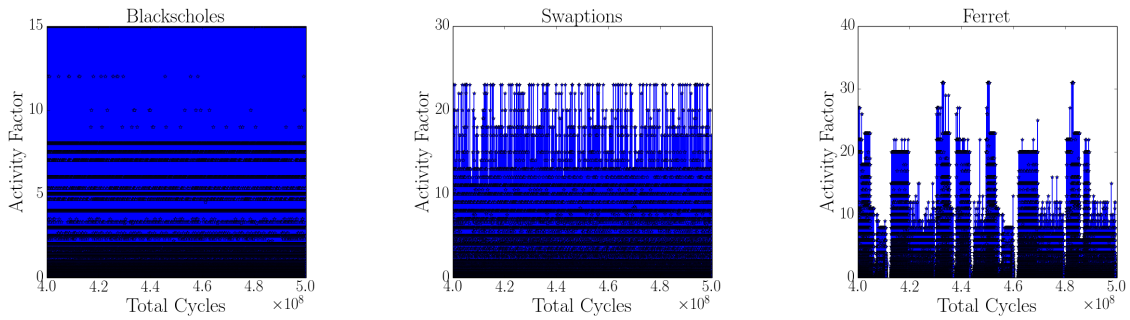


Figure 3.5: PARSEC activity factors: Each of the studied PARSEC benchmarks (excepting Freqmine) in terms of their FPU activity during an execution snapshot. Activity factor is defined as $\frac{BusyCycles}{IdleCycles}$ for a given time period.

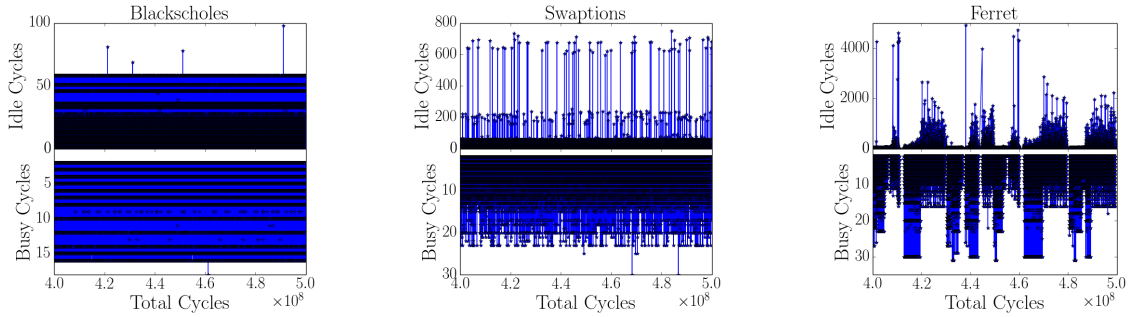


Figure 3.6: PARSEC idle and busy cycle comparisons: Each of the studied PARSEC benchmarks (excepting Freqmine) in terms of their Idle and Busy cycle counts during an execution snapshot.

Figures 3.5 and 3.6 show execution patterns for three PARSEC benchmarks, Blackscholes, Swaptions, and Ferret. Freqmine was not analyzed here due to its extremely low FPU usage. Its execution pattern is relatively trivial and the FPU was off nearly 100% of the time, regardless of power gating architecture. This can be seen in Chapter 5.

Blackscholes is shown as an application with a very consistent rate of FPU accesses. Swaptions is still consistent but has larger gaps where power gating may take place. Ferret lends itself nicely to power gating as there are many periods of inactivity. The frequency and

length of these periods of inactivity are more important to power gating than the magnitude of each application's busy time. Long idle times that are interrupted infrequently by accesses yields a large reduction in static power and minimizes the negative impact of waking up the FPU more than necessary.

MiBench

The MiBench benchmark suite is built for embedded systems [9]. The benchmarks in this suite are subdivided into different categories, each for a different area of the embedded market. These categories are Automotive and Industrial Control, Consumer Devices, Office Automation, Networking, Security, and Telecommunications. Each of these areas have applications that have different execution patterns that can lend themselves differently to power gating. Not all of them utilize the FPU at the same level as those chosen from PARSEC either.

To look at a wide range of execution patterns, one application was chosen from each of the categories to test its power gating effectiveness. Gathering applications from each category provided a good general overview of embedded system performance as a whole with respect to the power gating configuration. The specific benchmarks chosen for testing are Basicmath from Automitive, Lame from Consumer, Dijkstra from Network, Rsynth from Office, SHA from Security, and FFT from Telecommunications. Each of these applications have varying levels of FPU activity. In the Lame application, over 20% of the instructions are floating point while Dijkstra has less than 1%.

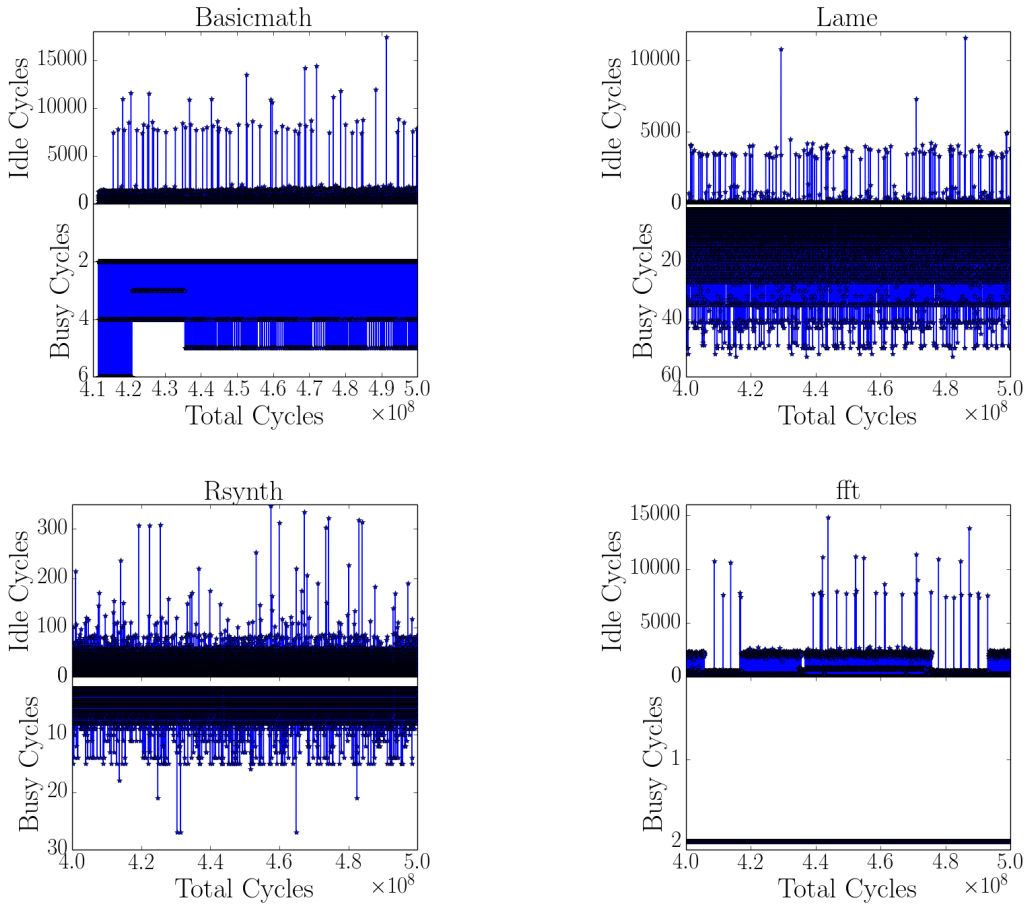


Figure 3.7: MiBench idle and busy cycle comparisons: Each of the studied MiBench benchmarks (excepting SHA and Dijkstra) in terms of their Idle and Busy cycle counts during an execution snapshot.

Figure 3.7 shows the idle cycles and busy cycles over a certain period of execution for several of the MiBench benchmarks. The SHA and Dijkstra benchmarks were not included because they did not execute a significant number of floating point instructions. Idle cycle periods of these two applications were very large while their busy cycle periods were short and infrequent.

A commonality among these applications when compared to their PARSEC counterparts is that each of these applications generally have significant, consistently long, idle periods. These periods make the applications more amenable to power gating, and specifically power gating in a deep sleep mode for hybrid power gating. A longer idle period makes for more efficient power gating for any configuration.

3.4.2 Testing Tools

gem5 is capable of simulating an entire architecture including wire latencies, cache designs, and branch predictors. Specific processor configurations can be found in Table 3.4. A high performance out of order processor with seven pipeline stages and multiple layers of caches was decided as the architecture to expand on. Each core of the processor contains a separate instruction and data cache and its own set of functional units. This configuration was chosen to be indicative of a modern high performance ARM processor.

All of the PARSEC benchmarks were run on a processor from Table 3.4 with four cores while the MiBench benchmarks were run on a processor with a single core. This is due to

Table 3.4: gem5 processor configuration

Core Properties		**Caches**		**Functional Unit Pool**	
ISA	ARM	ICache Size	32 kB	Int ALUs	2
Clock Speed	1 GHz	ICache Replacement	LRU	Int MUL/DIV units	1
Pipeline Depth	7	ICache Associativity	2 Way	FPU/SIMD Units	2
Instruction Order	Out of Order	DCache Size	64 kB	Mem Load Units	1
Branch Prediction	Bi-Mode	DCache Replacement	LRU	Mem Store Units	1
Integer Registers	128	DCache Associativity	2 Way		
FP Registers	192	L2Cache Size	2MB		
		L2Cache Associativity	8 way		
		L2Cache Replacement	Random		

the multithreaded nature of the PARSEC benchmark suite while the MiBench suite only contains singlethreaded applications.

Statistics were provided by gem5 at the end of benchmark execution. These, along with the processor hardware configuration, were funneled into an XML input file for McPAT through scripts. For PARSEC, since the experiments ran on a multicore platform running multithreaded applications, all results at the end were averaged across all the cores. To get a realistic view of functional unit power consumption, the statistics across functional units were also averaged in the case that one unit is heavily used over another.

McPAT was used to provide the complete power consumption for our particular architecture simulated with gem5. This includes all simulated functional units, the pipeline, caches, branch predictors and more. Only the hardware configuration and runtime statistics for the FPUs are of any importance since those are the only units whose power and performance are altered. MOSFET technology parameters are also included in McPAT's CACTI-P implementation. The parameters in this implementation were used in the current and voltage calculations in Section 3.1.1.

Chapter 4

Power Gating Schemes

The proposed power gating schemes dictate when sleep transistors will shut an FPU down. Each of them attempts to create a balance between static power reduction and wake up penalties. Wake up penalties include increased dynamic power, increased latencies, and decreased NEMS switch lifetimes. Other concerns a designer might consider is hardware complexity and area overhead. Such aspects are not considered in this thesis but are an inevitable concern to designers.

Of the seven stages of the pipeline, modifications were made in the issue stage. In the issue stage, waiting instructions are fetched from a queue and "acquire" the required functional unit if any is needed. At this point, in order to acquire a unit, the flow chart in Figure 4.1 is followed.

Preference for functional units is given to those that are awake or waking up and attempts

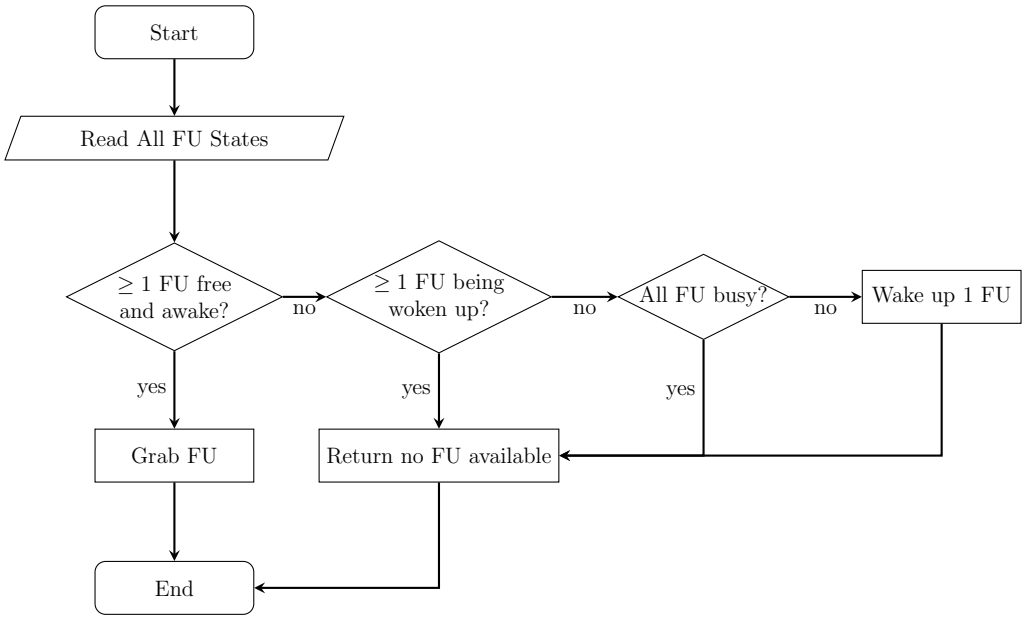


Figure 4.1: Acquiring a functional unit: The flowchart a processor must follow when assigning a functional unit to a specific instruction.

are made to avoid unnecessarily waking up units. Initially, a processor attempts to grab any open, available functional units. If unable, it stalls to wait for functional units to awaken. Only if there is a functional unit that is not working and is fully asleep, does it attempt to wake up that unit.

Each of the schemes operate with a counter or a set of counters. Each counter increments to a certain value N. For every scheme, a smaller N means the switch will turn off sooner. This results in a lower static power consumption yet higher dynamic power consumption. For NEMS switches, this also results in a much lower life expectancy. This is an important parameter with an effect on every result that is received. Because of this, it is analyzed across all executions. A lower N might be desired for executions where a macro will have a lot of downtime because it will be shut down quickly to take advantage of as many idle

cycles as possible. A higher N is more conservative and could be used in environments where the macro utilization is higher or unpredictable. In the context of a hybrid power gating scheme, multiple counters might be used, each of which have a different N. This provides some flexibility in preserving the lifetime of a NEMS switch and keeping any performance degradation low.

All of the discussed power gating schemes work based on the idea of temporal locality. If a unit has been used recently, it is likely to be used again in the near future. The cycle-by-cycle activity history of individual functional units are monitored to predict future usage.

Power gating logic occurs in two different instances. The first is on every clock cycle and the second is on an FU (Functional Unit) access. The specifics of the logic in each scheme are detailed in the following sections.

The high level block diagrams provided show each power gating scheme for a single sleep transistor. To modify a scheme for a hybrid layout, each sleep transistor is given its own internal units. That is, if an individual functional unit had a single counter in a single transistor layout, it has two in a hybrid layout, one dedicated counter for each sleep transistor. The deep sleep devices are used in the same manner as the shallow sleep devices yet a unit can only be put into deep sleep if it had already been in shallow sleep. The hybrid model and comparative differences are discussed in Section 4.2.

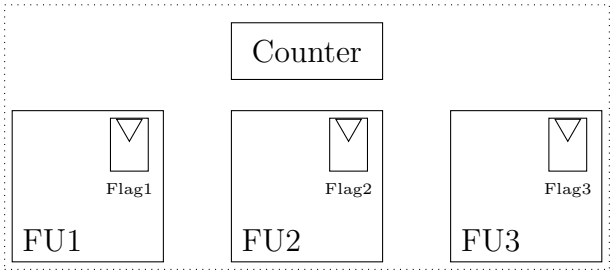


Figure 4.2: Flag based layout: The high-level block diagram for a Flag Based power gating scheme.

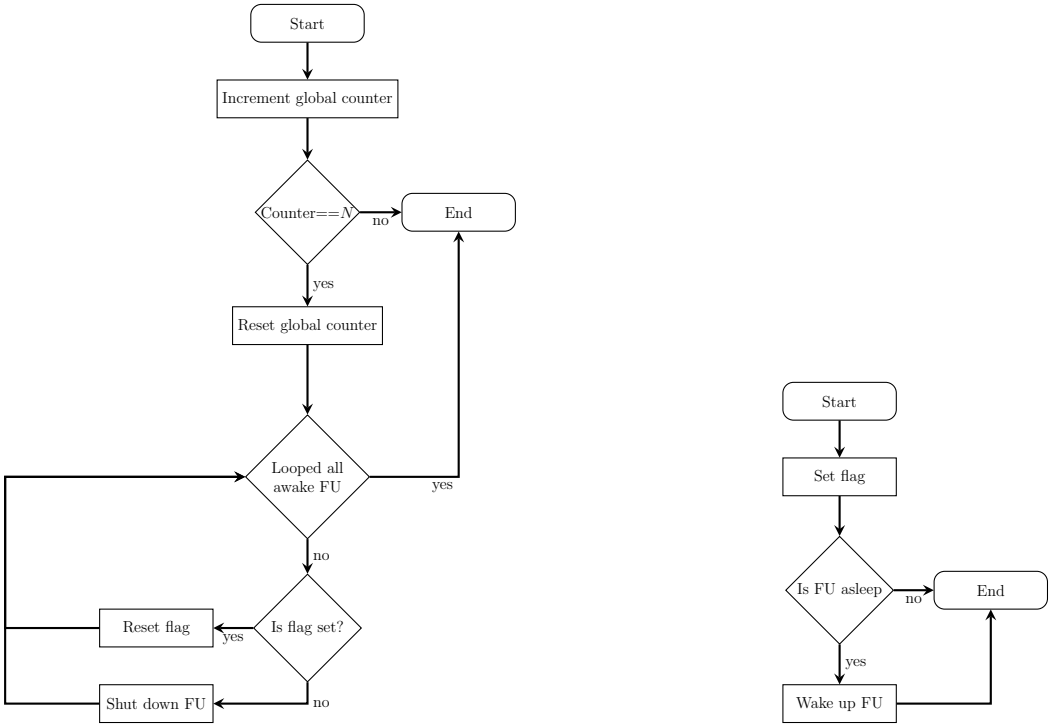
4.1 Single Transistor Power Gating

4.1.1 Flag Based

The flag based scheme was adapted from [11]. It consists of a global counter owned by the functional unit pool and a single flag dedicated to each power gating enabled functional unit. The flag is an indicator of whether or not the functional unit has been accessed recently and is used to predict the likeliness of being used in the near future. The counter dictates when each flag will be checked.

The flag based scheme is a conservative approach to power gating. It is meant to be simple and save switching overhead. The size of the global counter ($log(N)$ bits) is configurable to be relatively small or large, depending on the requirements of the architecture or types of workload. Since each functional unit only adds a single flip flop to its hardware, and a single counter is added per core, the overall impact of adding this scheme hardware is minuscule compared to the power and performance impact of power and gating itself.

The global counter is incremented on every clock cycle. Once the counter rolls over (reaches



(a) On every clock cycle (b) On FU Access

Figure 4.3: Flag based scheduling flowchart: How a Flag Based power gating scheme decides when to shut down a functional unit.

the predetermined N), the flags of all power gating enabled functional units are checked. If the unit is asleep, the algorithm skips it since there is nothing more that can be done. Otherwise, if the flag has been set, that means that at some point in the last N clock cycles, there had been an access to the FPU and it might be a poor decision to shut it down. The flag is reset to 0 and the hardware continues checking the remaining functional units. If the flag was not set, it is assumed that it is reasonably safe to shut down the unit. The shutdown signal is sent to the sleep transistor. The hardware then processes the rest of the functional units.

Whenever a unit is accessed, the flag is set. If the unit is asleep, it is also woken up.

Because there is a single global counter, this scheme is decoupled from the functional unit itself. Specifically, a given FPU may spend anywhere between N and $2N - 1$ idle cycles before being put to sleep. This means there is a wide range for when an FPU might be shut down, causing a degree of uncertainty. Because of this, a flag-based scheme may not be desirable and the counter scheme might be better suited for many functional units.

4.1.2 Counter

The counter power gating scheme attempts to draw on temporal locality in a more localized way. The scheme presented in [12] is the basis for this design. Instead of placing a single global counter across all functional units, each unit is given a private counter. A functional unit can keep better track of its own execution pattern with its own counter. Shut down

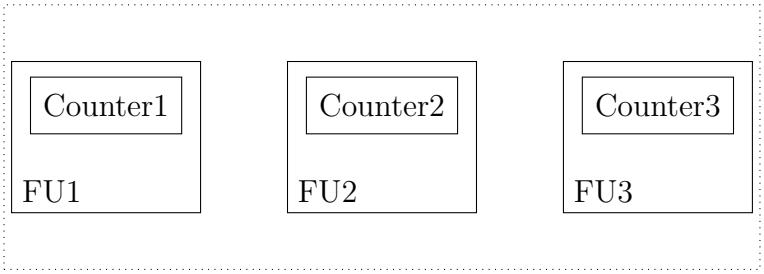
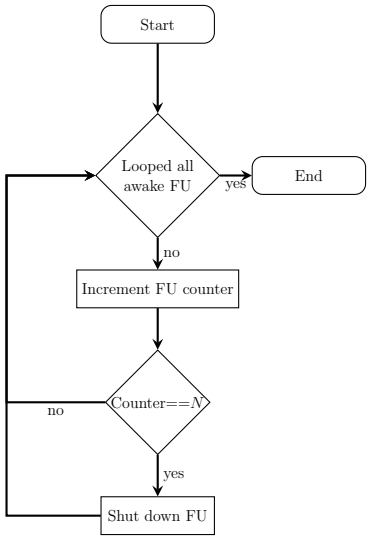
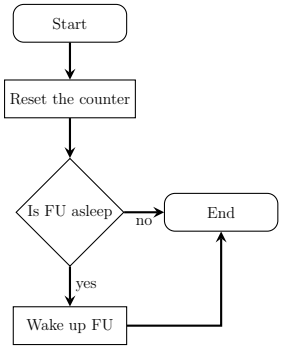


Figure 4.4: Counter layout: The high level block diagram for a Counter power gating scheme.



(a) On every clock cycle



(b) On FU Access

Figure 4.5: Counter scheduling flowchart: How a Counter power gating scheme decides when to shut down a functional unit.

decisions can be made with more precision.

There is a higher cost in general overhead due to each functional unit containing its own N-bit counter. The overhead is not much higher than the flag based scheme when considering a processor core with a low number of power gating enabled functional units. As the number of functional units grows, the number of necessary counters grows with it. While this adds slightly more complexity, the results in Chapter 5 show considerable potential in both power and performance.

Similarly to the flag based scheme, counters are updated every clock cycle. This happens for every unit that is still awake. If it is not awake, there is no need to continue counting for that unit. A check is performed to see if the counter has rolled over. If it has, the unit is powered down.

Once accessed, the unit's counter is reset and its sleep status is checked. If it is asleep, the unit is woken up.

This kind of counting gives a single unit the ability to make more precise decisions. Each macro's decision is tightly based on its own history and the macro knows exactly how long is has been idle. Additionally, though it is not modified in our testing, N can be changed between each separate functional unit. This might encourage one unit to be used more heavily so others can make better use of sleep sessions. It also becomes trivial to gate other pieces like integer units alongside FPUs. Their counters can be set to independently configured values for N. This scheme lends itself well to characteristics of individual units.

4.1.3 Trend

The trend scheme is built on top of the counter scheme. It utilizes the same hardware counters embedded into the functional units but each unit is also equipped with predictor hardware. The predictor hardware predicts upcoming idle gaps based on the history of power gating successes and failures. The trend scheme was constructed to be expandable to swap in many types of predictors. This can be seen in the high level control flow details. Existing

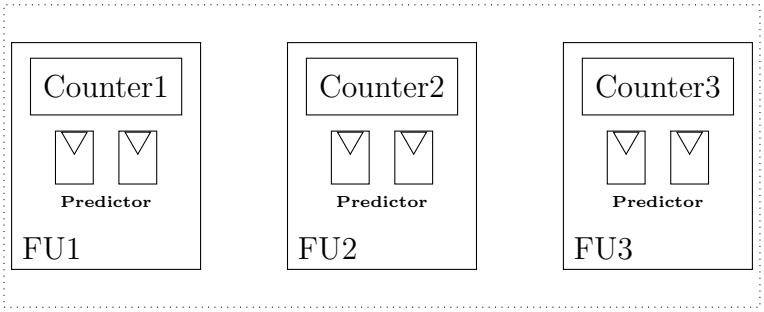


Figure 4.6: Trend layout: The high level block diagram for a Trend power gating scheme.

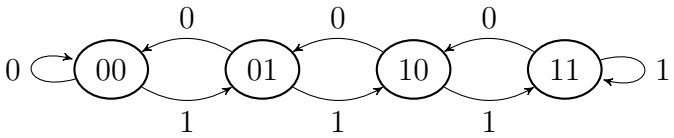
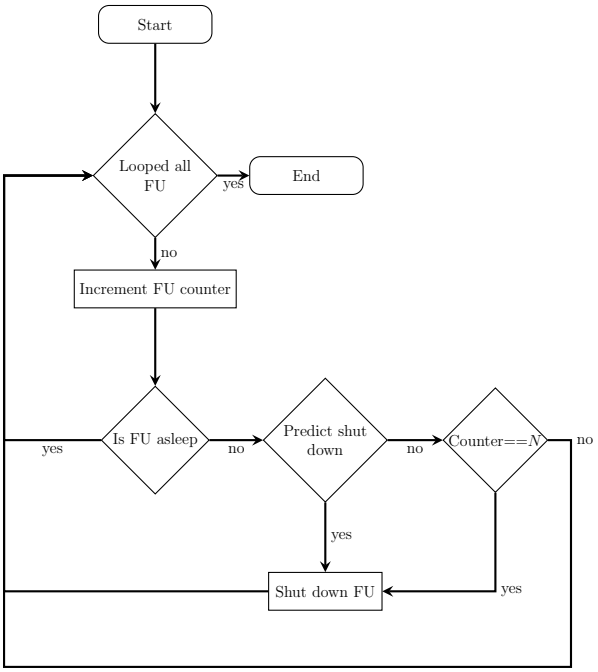


Figure 4.7: Trend 2-bit predictor: There are four states to be in at any given point. Every successful power gating attempt ($counter >= N$ and FU is asleep) is represented by a 1 on an edge. Every FU access that isn't successful power gating is represented by a 0 on an edge. The only state that is predicted to power gate is "11." All others predict no power gating.

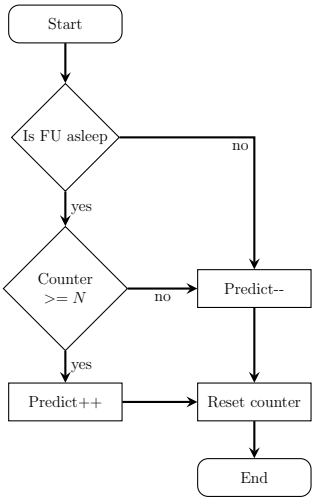
branch predictor models were the basis for the predictive hardware in this scheme. A simple conservative 2-bit predictor was modeled and placed in each FPU.

The 2-bit predictor logic, shown in Figure 4.7, follows one edge every time it is woken up. If there has been a successful power gating attempt, represented by a "1" edge, the current state moves closer to, or stays at, the highest state where power gating is predicted. All other states predict no power gating in an effort to minimize mispredictions and the energy and performance overhead that accompanies them. All FU accesses that are not classified as "successful" decrement the current state to a floor of "00."

With a trend power gating scheme, an attempt is made to capture history even further than the counter scheme. If a string of successful attempts are made to power gate a certain functional unit, temporal locality dictates it is likely to happen again. The trend scheme



(a) On every clock cycle



(b) On FU Access

Figure 4.8: Trend scheduling flowchart: How a Trend power gating scheme decides when to shut down a functional unit.

takes advantage of that knowledge.

Similar to the counter scheme, all the functional units are checked every clock cycle. However, their counters are updated regardless if it is asleep or not. This is done in order to have a notion of a successful power gating. A shut down is only successful if it had been idle for at least the last N cycles. If awake, the functional unit's predictor is checked to see if it is a candidate to be shut down early. If it is not, the counter is checked to see if enough clock cycles have passed since the last access. If enough cycles have passed or the predictor dictates an early shut down, the unit is powered off.

On an access, if a unit is either awake or if the counter had not yet reached N, it is marked as an unsuccessful period. Otherwise, it is marked successful. This is the information the predictor can use to judge the history. Before concluding the access, the counter is reset.

Since this particular trend scheme was built with extensibility in mind, Figure 4.8 accounts for that. The power gating scheme can implement any type of predictive hardware that gets checked in the "Predict shut down" decision in Figure 4.8a. The "$Predict++$" and "$Predict--$" blocks in Figure 4.8b are the other predictor access points. Any custom predictor can handle power gating successes or failures there.

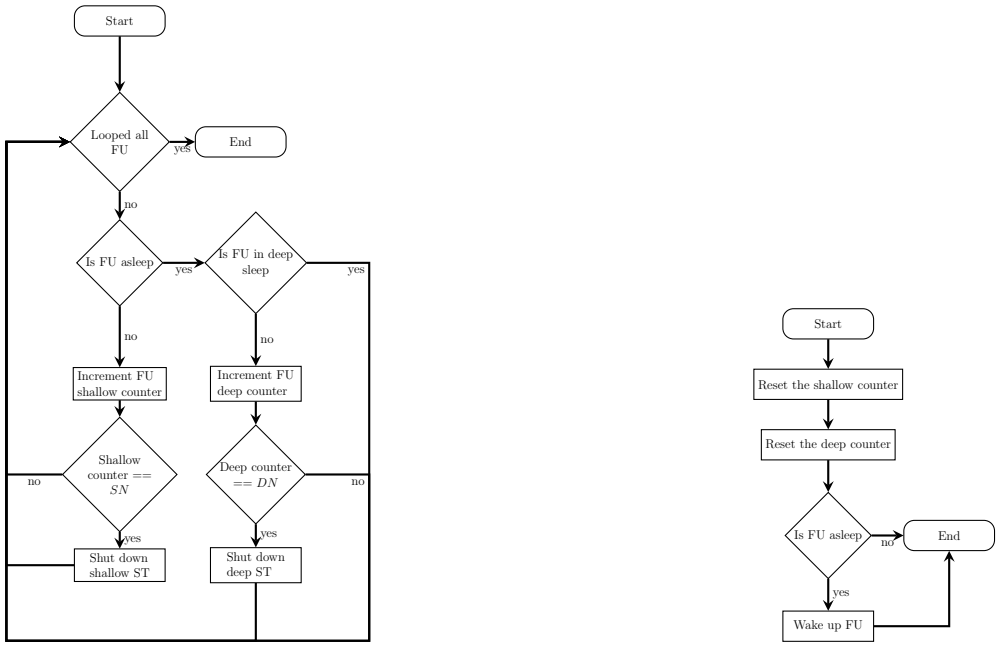
4.2 Hybrid Counter

All previously discussed power gating schemes were expanded into hybrid models. However, in this thesis, the only one analyzed is a hybrid counter scheme. This is the simplest scheme

of the three devised. It also provides a much larger energy savings compared to flag-based and is comparable to the trend analysis, making it the most efficient in terms of complexity and energy savings.

In a hybrid counter scheme, each functional unit contains two counters instead of a single one. One of them is paired with the shallow sleep transistor and the other is paired with the deep sleep transistor. Each one is incremented and checked depending on the functional unit state. They also have their own upper counting limits, SN and DN, for the shallow counter and the deep counter, respectively. With different length counters, the lifetime and effectiveness of each switch can be individually controlled. The shallow transistor can be effectively used to quickly shut down a functional unit. This saves as much energy as possible. However, it is only expected to be in a shallow sleep for shorter periods of time. The unit will either be woken up or put into a deep sleep. If a long enough idle period is seen, the scheme places the functional unit into a deep sleep mode with the secondary transistor. This protects the secondary transistor from being shut down prematurely and wasting time, energy, and its limited number of lifetime switches.

There are two nearly identical paths for a functional unit to take, one checking for shallow sleep and the other for deep sleep. If not asleep, the counter will be incremented and checked to see if it has reached its threshold SN. If that limit has been reached, the unit is placed into a shallow sleep and the clock cycle is completed. If it is already asleep, a deep sleep check is done. If the unit is in a deep sleep, there is nothing more to happen and this functional unit is done with this clock cycle. Otherwise, the deep sleep counter is incremented and checked



(a) On every clock cycle (b) On FU Access

Figure 4.9: Hybrid counter scheduling flowchart: How a Hybrid Counter power gating scheme decides when to shut down a functional unit.

against DN. If it is idle for long enough, the functional unit is placed into a deep sleep.

There is little logic necessary to expand the counter scheme to this hybrid model as seen by Figure 4.9. The checks on each clock cycle are identical to what is already done for a single transistor and the sole modification for what occurs on each access is resetting the deep sleep counter. The redundancy of the design eases the burden of implementation.

While the complexity of expanding a single transistor scheme into a hybrid one is almost trivial, the results shown in Chapter 5 indicate that there are considerable gains to harness.

Chapter 5

Results and Analysis

Power gating with different types and/or numbers of switches introduces many parameters to be addressed. In this work, focus was placed on types of applications, differences between switch types, power gating schemes from Chapter 4, and changing the N value within power gating schemes. Energy, performance, and NEMS lifetimes are analyzed with respect to all of the gathered results. Energy was chosen to be discussed instead of power because of its relationship with time. As discussed in Section 2.3 and as Figure 3.1 shows, when power savings increase, it does not mean that power gating saves more energy or provides a net benefit. Energy is more indicative of successful power gating.

Figure 5.1 provides an overview of the results. It shows four Pareto charts, one for each chosen benchmark. Each point in the graphs represents a different power gating scheme with different configurations and types of sleep transistors. They create a holistic view of

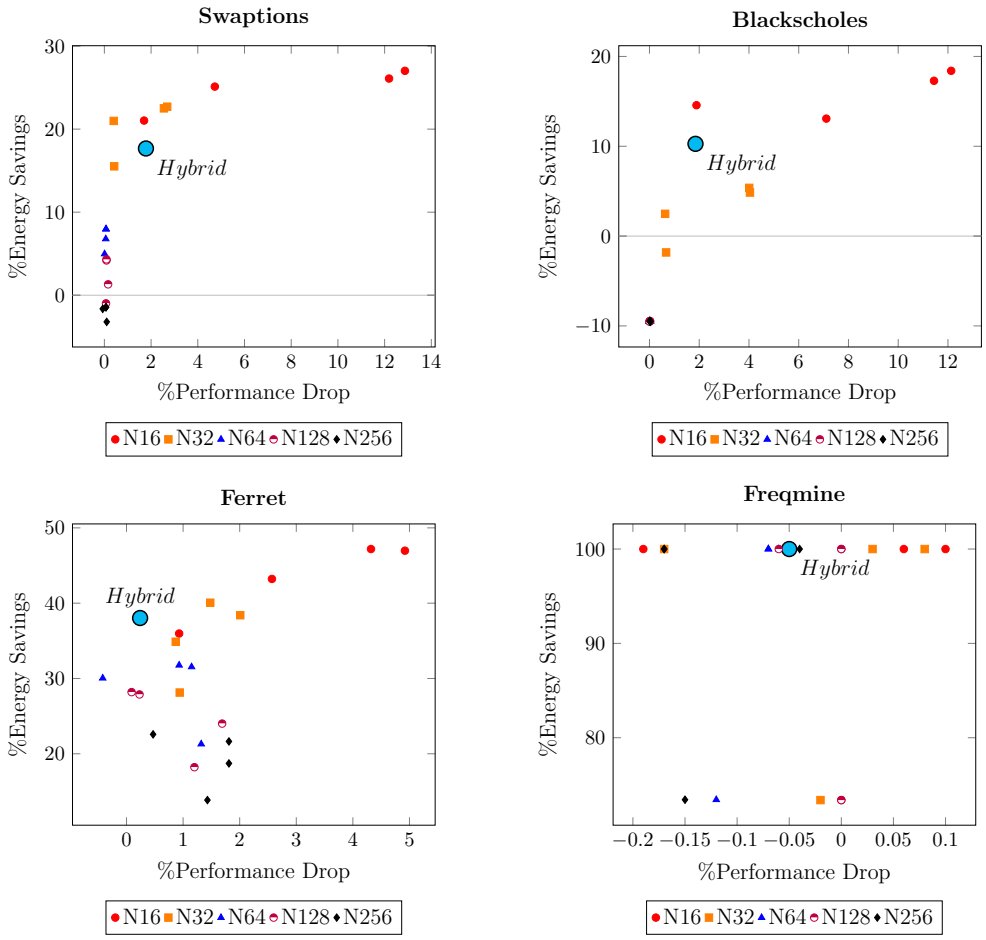


Figure 5.1: Overview of results: Power gating schemes with both NEMS and PMOS are shown in terms of their performance drop and energy savings, separated by their value N. A Hybrid Counter scheme where $SN = 16$ and $DN = 256$ is shown for each application.

how different types of power gating affect both performance and energy.

The hybrid model operated within the optimal region for each application. The optimal region was where performance degradation was minimal for a large amount of energy savings. Each of the high FPU activity benchmarks had this energy-performance trade-off managed by the schemes. The low activity benchmark, Freqmine, showed many results operating at 100% energy savings with virtually no impact on performance. Freqmine was the baseline benchmark used to show what can be reasonably done in each configuration for an application that would have the largest benefit from power gating.

Hybrid power gating exposed its flexibility by operating near the optimal point across every type of application when compared with PMOS and NEMS of various schemes. This flexibility is a desirable attribute in general purpose processors. In many cases, it is unknown whether a user would run a large amount of scientific applications, heavily utilizing the FPU, or applications closer to the Freqmine benchmark with little FPU activity, or any mix of the two. In testing applications with a large number of FLOPS, it is assured that even in worst case scenarios, the hybrid model operates efficiently.

Every application ran against the architecture set in Table 3.4. The results were averaged across all four cores and across both FPUs in a core. This ensured that if a single FPU inside a single core was more (or less) heavily utilized than any others, a clear trend can still be made based on the application and power gating configuration. In practice, core utilization varied for some applications more than others but each FPU within each core were used about equally.

All energy and performance numbers are with respect to a processor that has not been power gated. This gives all configurations for each application the same baseline to compare against. Energy and performance are compared in terms of effectiveness instead of pure numbers.

The linux thread scheduler was also used in the operating system running on the multicore processor. This means that there is some small expected variation in execution time. Even in the face of high latency NEMS power gating, it might seem there is a performance gain instead of performance loss if the FPU activity is low enough. Differences in execution patterns caused instructions to be ordered differently and the thread scheduler caused a degree of nondeterminism. This variation can be seen in the performance drop for the Freqmine benchmark in Figure 5.1.

5.1 Applications

It is important to note the role that applications play in effective power gating. This role must be considered in order to ensure that the more active applications will not perform so poorly that both energy and time are wasted. Likewise, designers also want to maximize the energy savings of low activity applications.

From the PARSEC benchmark suite, four applications were chosen to study power gating. Figure 5.2 shows the effect of each benchmark in terms of energy savings and NEMS switch lifetime. Three of the applications have a high percentage of cycles spent with the FPU busy

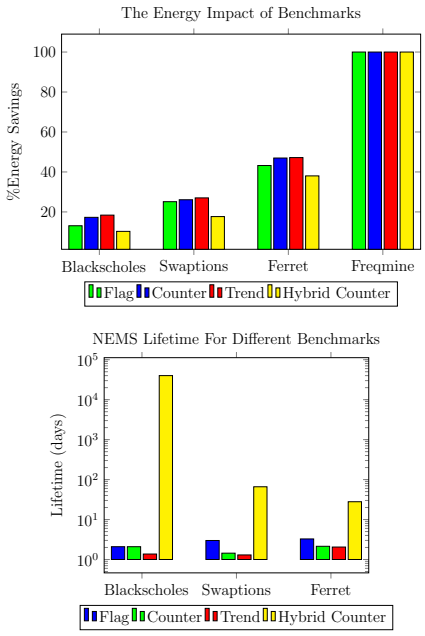


Figure 5.2: Energy savings and NEMS lifetime for various applications: The graphs compare each application with each NEMS scheme with regards to their energy savings and how long they will last (in number of days) provided that the benchmark continuously runs. Power gating was built with $N = 16$ for single transistor schemes and $SN = 16$ and $DN = 256$ for Hybrid.

and one has nearly no FPU activity. Most applications will operate more efficiently when compared to the three high FPU activity applications. Each benchmark performs distinct tasks. This means that the frequency between FPU accesses in an instruction stream and the percentages of different types of instructions are also distinct between them. The impact of the specifics in instruction accesses is compared through energy savings and performance. Furthermore, the differences between applications will be a continuous discussion throughout the rest of the results as well.

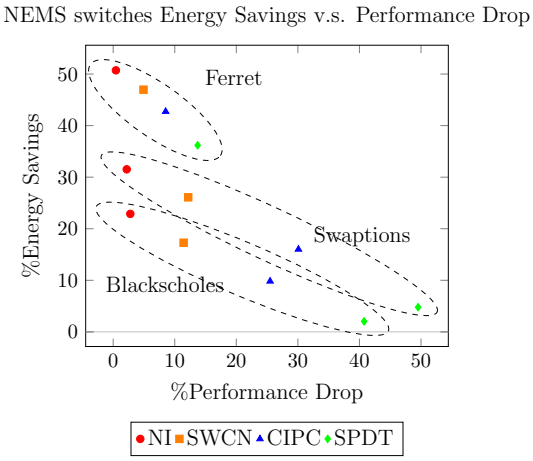


Figure 5.3: Energy and performance for various NEMS switches: Each application is executed with $N = 16$ and are executed with the Counter power gating scheme. NI is a near ideal NEMS switch, SWCN is Single-Walled Carbon Nanotube, CIPC is Curved In-Plane Cantilevered Thin Film, and SPDT is Single-Pole Double-Throw. Details of each switch can be found in Table 3.1.

5.2 Different Switches

Major differences between sleep transistors were discussed in Section 3.1. These differences have been analyzed to get an understanding of how PMOS transistors compare to various NEMS switches and the hybrid model.

The NEMS switches presented in Table 3.1 differ in ways that alter power gating results and that can be reasonably considered. The largest of these include latencies and lifetimes. Both energy and performance are highly dependent on the latency and can be seen in Figure 5.3. Any perceivable energy savings quickly diminish when presented with a longer latency. When paired with much longer execution times, NEMS are shown to be an unacceptable replacement to MOSFETs.

Switches with longer latencies consistently, across applications with heavy FPU usage, saw

a large drop in performance and energy savings. To prevent this decreased performance, a scheme has to be more cautious with when it shuts down the FPU. However, if a scheme is overly cautious, the increased V_{DD} will still cause a rise in consumed energy. In this case, the unit would not have been gated long enough to take advantage of any reduced power. A low switching latency is critical to efficient power gating operations.

Of the NEMS switches analyzed, the Single-Walled Carbon Nanotube type appeared to have the best lifetime and lowest latency and was used in all tests where a NEMS switch type is not specified. Other NEMS switch types might be used in fabrication due to familiarity with how they are built, process yield consistencies, lower actuation voltages, material types, contact resistances, and other factors. These factors cannot be taken into account due to having negligible or no direct relationship to performance, energy, or lifetimes. Regardless, as the latency of the NEMS switch increases, the effectiveness of the switch decreases. The power gating schemes need to be more cautious not to introduce dramatic reductions in performance and the degree of cautiousness will depend on the chosen switch.

Differences between PMOS, NEMS, and the hybrid model have also been tested. Each have their own advantages and disadvantages across all types of applications. Figure 5.4 shows the effect of the different switches.

A NEMS only design was consistently much more efficient in energy savings yet resulted in a significant increase in the number of cycles taken to execute a program. The PMOS switch only saved slightly less energy than a NEMS switch for the higher FPU activity applications while not causing a large performance degradation. In looking at the Freqmine benchmark,

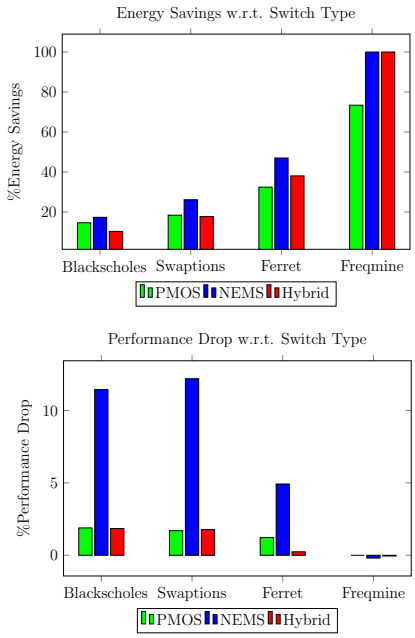


Figure 5.4: Energy and performance for switch types: Comparing differences between PMOS, NEMS, and hybrid in terms of both energy and performance is done across all 4 benchmarks. The Counter scheme is used where $N = 16$ for single transistors and $SN = 16$ and $DN = 256$ for hybrid.

however, the PMOS saved less than 75% of the energy that a NEMS or a hybrid model saved. If a circuit is idle for a large amount of time, there is a vast amount of energy that is potentially saved beyond what a PMOS can do. This is why a NEMS switch is introduced in power gating.

Performance and energy savings were similar between a PMOS and the hybrid model for the high FPU activity applications. As the FPU utilization decreased, the energy savings of the hybrid model more closely matched the NEMS switch. In this region, the weakness of the PMOS became clearer. The large idle periods were captured by the deep sleep NEMS switch while the FPU was still gated by the PMOS during shorter idle periods.

Also shown are applications where the hybrid model performed worse than both PMOS and

NEMS. This is largely due to the V_{DS} differences across the PMOS transistor. Section 3.2 discussed differences between various values for V_{DS}. For this hybrid model, the comparative power consumption is larger in the shallow sleep regions. In the Blackscholes and Swaptions executions, the FPU did not spend enough time in deep sleep to account for the increased shallow sleep transistor V_{DS} and the increased dynamic power. Most sleep cycles were spent in a shallow sleep. When the power gating scheme had spent more clock cycles in deep sleep, it operated better than a layout with a single PMOS.

5.3 Power Gating Schemes

Each scheme has a level of aggressiveness which is indicative of how often the power gating logic will shut down the FPU. For the same N, the trend scheme is the most aggressive, followed by the counter scheme, then the flag scheme. The impact of each scheme is shown in Figure 5.5.

For a more aggressive policy, more energy is saved as the FPU is put to sleep for more cycles. With the larger energy savings, there is also a worse performance drop. Every time an FPU is woken up, the processor stalls until the FPU is fully awake and ready to execute instructions. This has a direct negative impact on performance. This relationship between aggressiveness, energy, and performance is true for both NEMS and PMOS, though the degree of each varies. The hybrid counter scheme was also compared to each single transistor scheme type. Hybrid power gating offered performance and energy savings rivaling PMOS transistors for

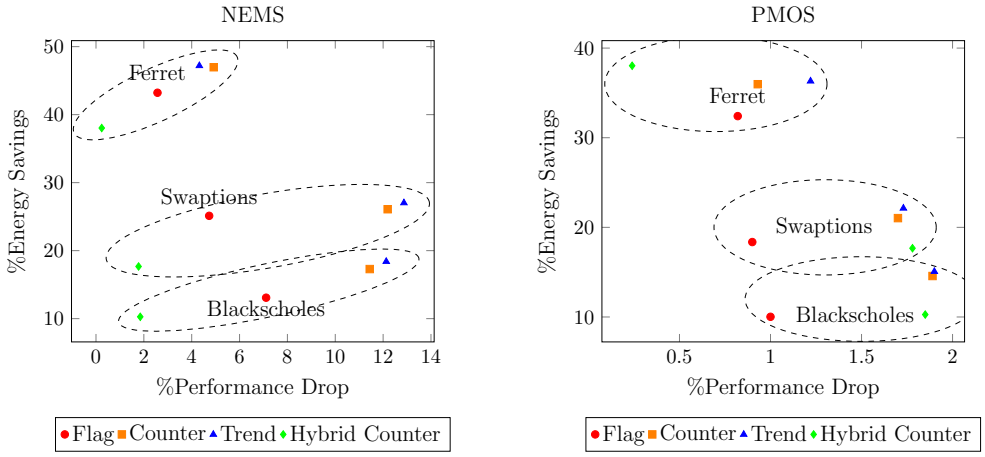


Figure 5.5: Energy and performance for different schemes with NEMS and PMOS: Compares the most power efficient designs without regard to device lifetime. The NEMS switch schemes are built where $N = 16$ and the Hybrid scheme is built where $SN = 16$ and $DN = 256$.

each high activity applications and offered very low performance loss when compared with NEMS switches.

The more aggressive schemes come with an additional consequence besides decreased performance. They may also negatively affect the lifetime of a NEMS switch. Since NEMS only have a certain number of switches before they break, the number of sleep sessions must be carefully controlled. To achieve the rates shown in Figure 5.5 they must switch at a very high rate. If running the Blackscholes or Swaptions applications constantly, the NEMS sleep transistor would break in less than two days. Because of this, a NEMS-only power gating scheme with a low N is not a realistic option.

While the hybrid counter scheme did not achieve the same energy savings of a pure NEMS design in some benchmarks, it did drastically increase the lifetime of the NEMS switch. This was done while offering the flexibility for 100% power savings in low activity applications

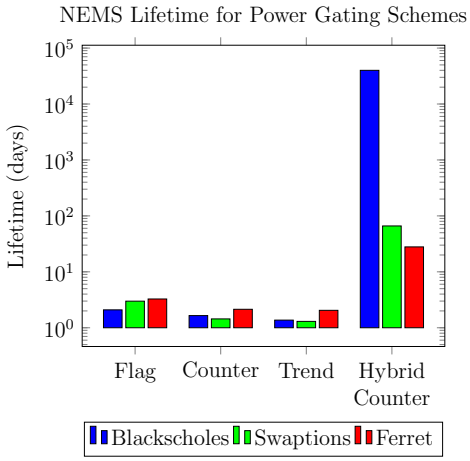


Figure 5.6: NEMS lifetime for different schemes: Benchmarks (minus Freqmine) with respect to power gating schemes where $N = 16$ and for Hybrid, $SN = 16$ and $DN = 256$.

and energy savings comparable to PMOS in high activity applications.

5.4 Changing N

N reflects the level of cautiousness a particular scheme has. With a smaller N, the FPU will be gated, or checked for gating eligibility, earlier compared to one with a larger N. Shutting down early attempts to save more energy, taking advantage of as many idle cycles as possible. This increases the rate of sleep transistor switching as more sleep sessions are attained. Increased transistor switching causes a higher dynamic power, worse performance, and, if using a NEMS switch, the transistor will break sooner. A larger N mitigates some of the negative effects of shutting down early but would not take advantage of as many idle gaps. If enough idle periods are not captured, the increased V_{DD} will have a detrimental effect on energy. In this section, the effects of different sizes of N are quantified.

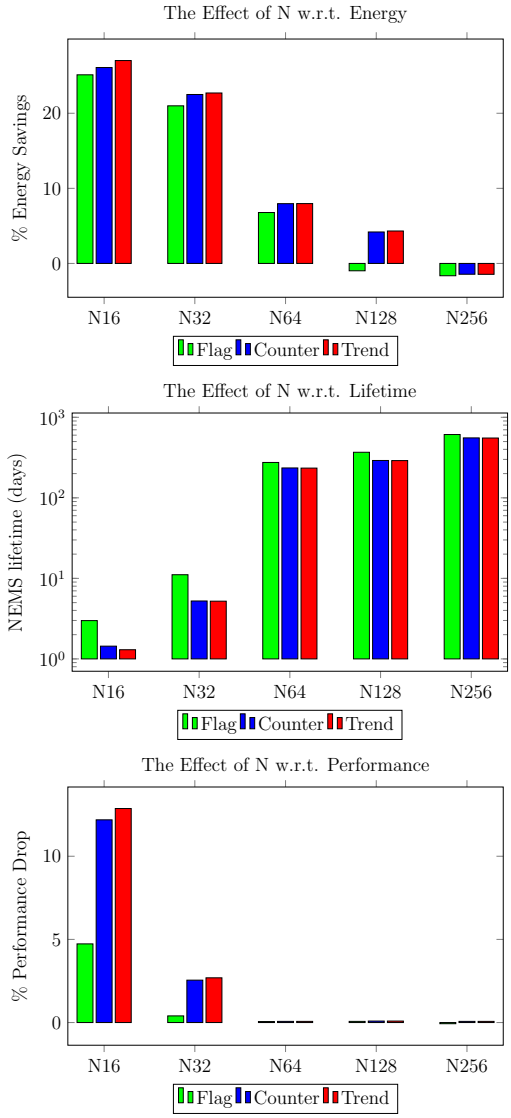


Figure 5.7: The effect of changing N on schemes: Analysis is performed for the Swaptions benchmark on a various schemes with various values for N and a NEMS sleep transistor.

As N increased, Figure 5.7 shows energy savings decreased and performance got better. For NEMS switches, a larger N was necessary regardless of energy savings. A small N caused a NEMS switch to break in a matter of days while a larger N increased the lifetime in even the most active applications. The lifetime improved to about one year of constant use for these applications while still providing energy savings. Performance degradation also quickly diminished once N was increased.

Each variation of N caused the same effect across every scheme. When N was too large, energy savings began to degrade to the point where power gating was harmful. When N was too small, the NEMS switch quickly broke and the processor performed much slower.

Every benchmarks resulted in the same trend for a changing N. An increase in N increased the NEMS switch lifetime but failed to capture many possible idle gaps and even performed worse at certain points. For the Blackscholes application, when $N = 64$, the larger V_{DD} over the system caused a much larger overall power consumption. The scheme could not put the FPU to sleep for the number of cycles required to break even and account for the increased voltage. The decrease in energy savings is seen for Swaptions and Ferret as well. Only Freqmine is unaffected due to its infrequent FPU utilization.

While the issue still existed in hybrid power gating, the effects of a bad N were reduced. MOSFETs also have a nearly infinite lifetime so the only concern in suffering additional MOSFET switches is a drop in performance and increased dynamic power. Performance is not as large of a concern in MOSFETs as they are in NEMS switches as shown in Figure 5.4. This is due to their fast switching speeds. Because of this, the N for the PMOS shallow

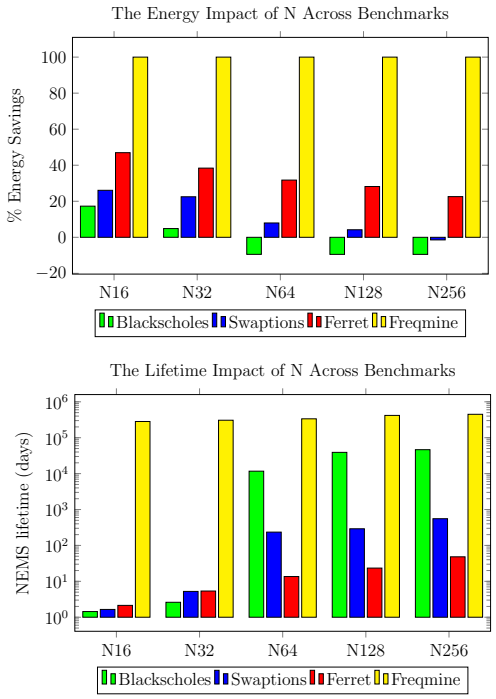


Figure 5.8: The effect of changing N on benchmarks: Each benchmark was run with a Counter power gating scheme and single NEMS switch.

sleep transistor is kept low. To preserve the NEMS lifetime, it is desired that NEMS only be turned off for periods that are assuredly long. This yields the nearly 100% energy savings seen in the Freqmine application with energy savings performing close to PMOS for all other applications.

Analysis has been done on the proper value for the deep sleep transistor's N in an effort to maximize energy savings with respect to its lifetime. The performance impact of changing the deep sleep N is consistently low since the majority of switching is done on the MOSFET. Therefore, it is not accounted for.

Increasing values of DN (Deep N) were used for the deep sleep transistor's counter while SN (Shallow N) was held constant. Figure 5.9 shows that the effects of a changing DN

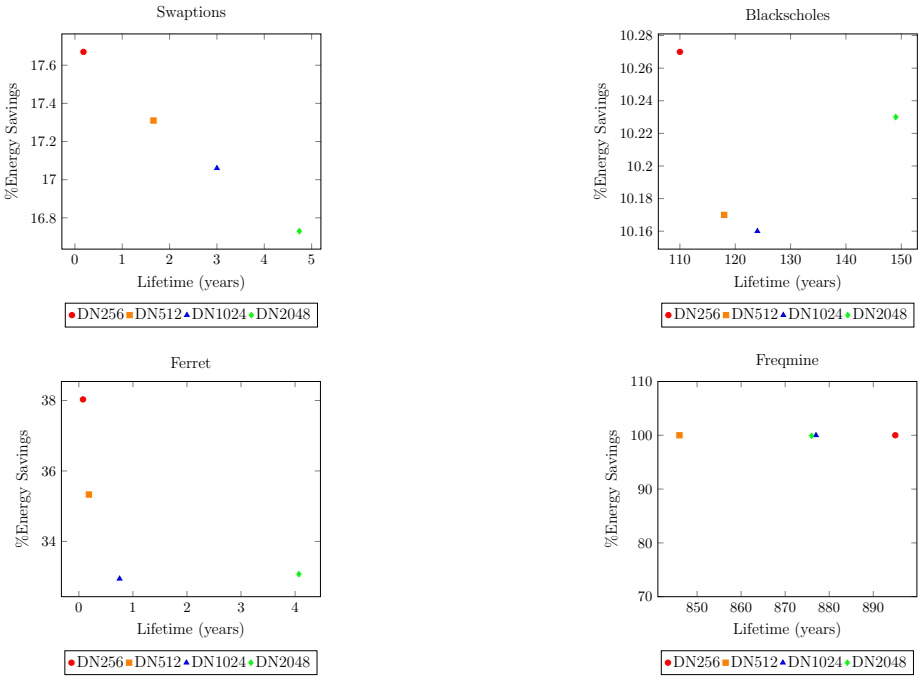


Figure 5.9: Energy and lifetime for various deep N: For each benchmark, $SN = 16$ in the Counter scheme and DN is given various values.

for each application vary in magnitude but all consistently encourage using a larger DN. The reduction in energy savings from $DN = 256$ to $DN = 2048$ was never more than 3%, as shown for the Ferret benchmark, yet the lifetime increased from days to multiple years of constant use. A lifetime increase was seen for each application, even Blackscholes whose lifetime already lasted over 100 years with $DN = 256$.

Regardless of the DN, the Freqmine benchmark reported a 100% energy savings and the lifetime did not change by any noticeable degree. This was to be expected. For an application that made light use of the FPU, the FPU would be put to sleep nearly the same number of time for any DN. If there was any difference, it would only differ by a comparatively small amount and these differences would be majorly dictated by instruction ordering and

thread scheduling. During the majority of the long deep sleep waiting period, the FPU was in shallow sleep as well, still saving nearly 75% of the idle energy. In that regard, waiting 2048 cycles instead of 256 to put the FPU to deep sleep failed to make a noticeable impact on total energy savings.

With a large DN, the FPU can be gated by a PMOS transistor for the majority of time when there is a high level of FPU activity. During times of low activity, the NEMS switch will be turned off for 100% power savings. Each transistor is used in a manner that reflects their best properties and provides a greater overall reduction in energy than is possible for any single transistor across a breadth of applications.

5.5 Embedded System Applications

The MiBench applications, Basicmath, Lame, Dijkstra, Rsynth, SHA, and FFT, were all analyzed in terms of their energy savings and performance degradation for PMOS, NEMS, and hybrid power gating. The lifetime of a NEMS switch is also analyzed for the NEMS-only and hybrid power gating. The specific scheme that was simulated was a counter scheme where $N = 16$ for PMOS-only and NEMS-only and the hybrid counter scheme where $SN = 16$ and $DN = 256$ for hybrid power gating.

As noted in Section 3.4.1, both SHA and Dijkstra have very little FPU activity. As expected, Figure 5.10 shows that their energy savings for NEMS and Hybrid are ideal, at 100%. The PMOS switch saves about 75% of energy. The lifetime of the NEMS switches for these

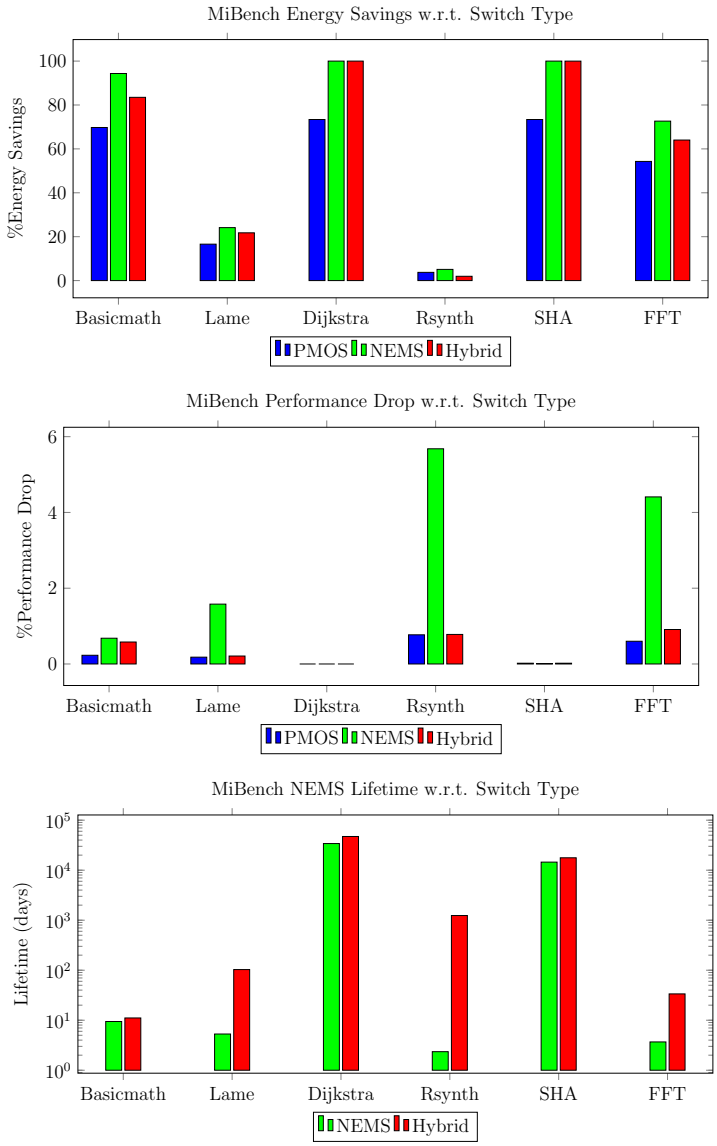


Figure 5.10: MiBench effectiveness for switch types: These graphs compare differences between PMOS, NEMS, and hybrid in terms of energy, performance, and lifetime across the 6 chosen MiBench benchmarks. The Counter scheme is used where $N = 16$ for single transistors and $SN = 16$ and $DN = 256$ for hybrid.

applications are also significantly long. These results match what was seen for the Freqmine application in the PARSEC benchmark suite.

All of the other applications show results similar to those analyzed for the PARSEC benchmarks as well. Longer idle cycles for applications, as seen in Figure 3.7, yield larger energy savings. NEMS-only power gating also consistently has a much larger performance drop when compared to PMOS-only and hybrid power gating. The magnitude of the performance drop is still relatively lower than all of the PARSEC benchmarks, though, with a maximum reduction of less than 6%.

In monitoring the lifetime of NEMS switches, hybrid power gating always reduces the number of times the NEMS switch will toggle. This trend is consistent across all applications, though the magnitude of the difference is not consistent. With the increased lifetime, energy savings nearing that of NEMS-only power gating, and minimal performance drop near that of MOSFETs, hybrid power gating is optimal for these embedded systems applications.

5.6 Coarse-Grained Power Gating

The coarse-grained comparisons between switches were performed by gathering average synthesized loads on a processor core in terms of its active time, idle time, and number of cycles between sleeping over an assumed period of time. For the purposes of this study, it was assumed that the core of a processor had a periodic behavior. This provided regular intervals in which power gating occurred.

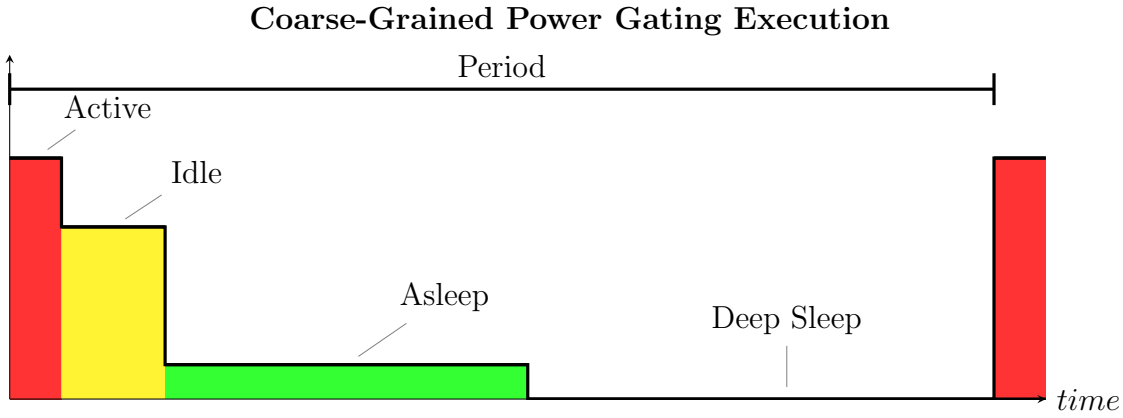


Figure 5.11: Coarse-grained power gating execution: In the coarse-grained power gating approach, each period is assumed to consist of 4 different sections. The active mode lasts for various percentages of the entire period. The idle time lasts for 50 clock cycles. The asleep time lasts for 500 clock cycles if using hybrid power gating or the remainder of the time if not. The deep sleep time lasts the remainder of the cycles of each period in hybrid power gating.

Figure 5.11 shows a period of execution of a power gated core. Each period length consists of active time, idle time, sleep time, and deep sleep time for hybrid power gating. Single transistor power gating only consists of active, idle, and sleep time. The active period lasts for a certain percentage of the total period length. The length of this period affects energy savings in a core. The idle time lasts 50 cycles before putting the core to sleep. For hybrid power gating, the shallow sleep time lasts 500 cycles before putting the core into deep sleep. The 50 cycle idle time is equivalent to SN and the 500 cycle sleep time is equivalent to DN from the hybrid counter power gating scheme from Section 4.2.

Figure 5.12 shows how PMOS, NEMS, and hybrid power gating performed for gating the core of a processor in specific intervals at 40% active periods. The performance was affected by how often the sleep transistor switched and the switching latency of each transistor type. The performance degradation for each switch began to level off around 500 cycles per period.

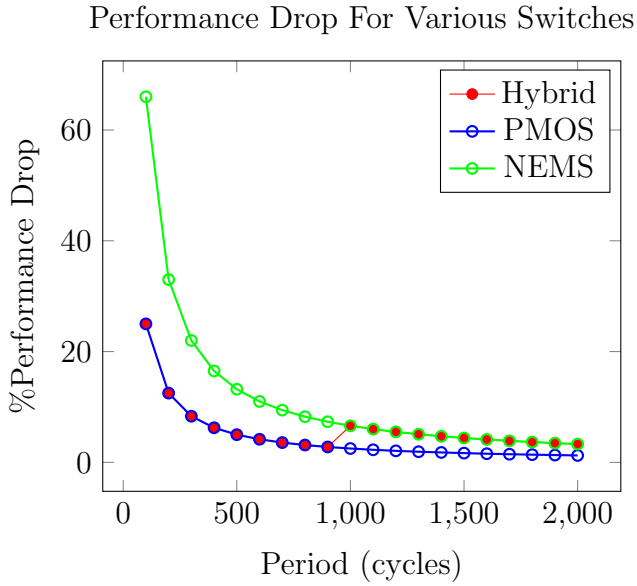


Figure 5.12: Coarse-grained performance drop: The performance drop for PMOS, NEMS, and the hybrid MOSFET-NEMS sleep transistors across different periods where the period is the number of cycles between sleep sessions. 40% of each period is active. 50 cycles are spent idle before shutting down and 500 cycles are spent before going into deep sleep.

At 40% active periods, hybrid power gating matched MOSFETs at time periods of fewer than 900 cycles and began to match NEMS at periods of 1000 cycles. This was because in regular intervals, the deep sleep sessions would not occur unless the idle periods were of a sufficient length. Once they did match, the core was always in deep sleep when being woken up.

In Figure 5.13, periods of 500 cycles to 2000 cycles are shown for various energy savings and percentages of core activity. The efficiency of power gating quickly diminished as cores grew more active. This occurred in all switch types.

Figure 5.14 shows the cross sections of both the most active and most idle periods from Figure 5.13. When the core was active at 40%, the energy savings of hybrid power gating

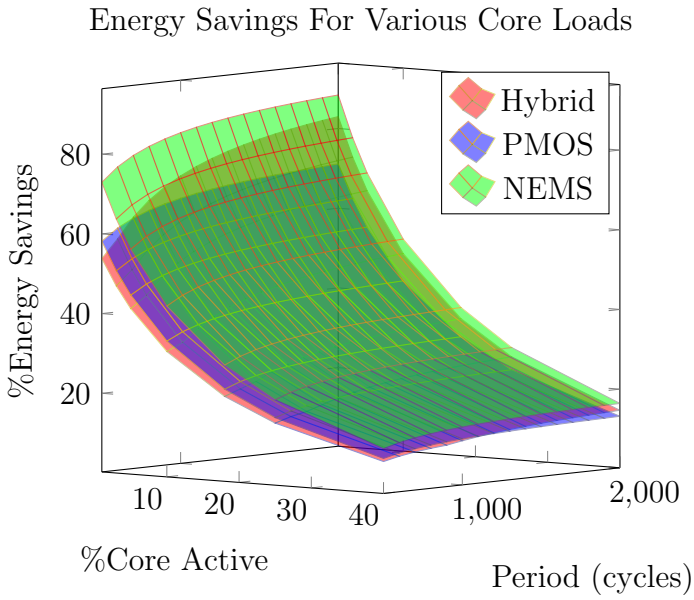


Figure 5.13: Coarse-grained overall energy savings: This chart displays the percentage of energy saved for various period lengths and percentages of time the core is active. The period is the number of cycles in between sleep sessions. %Core active is the percentage of time that the core is in the active state within each period.

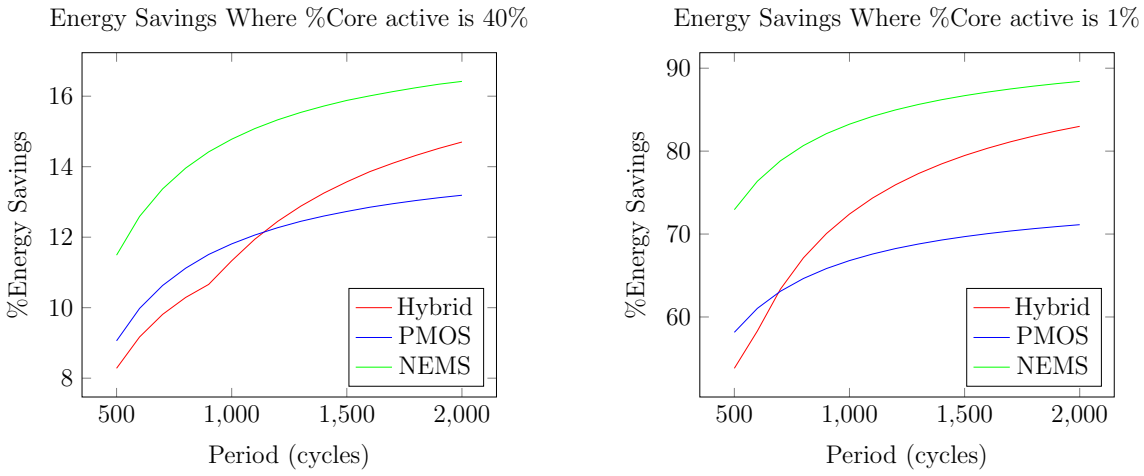


Figure 5.14: Coarse-grained energy savings under different loads

surpassed the energy savings of the PMOS-only savings at large periods. This was because it took many cycles to put the core into a deep sleep and each period had fewer idle cycles with 40% active periods.

When the number of active cycles was 1% of the total period, the cross section in Figure 5.14 shows the energy savings of the hybrid model surpassing PMOS-only when the period length was much lower. There was a greater opportunity to place the core into deep sleep for lower activity. The NEMS switch was consistently better at saving energy due to its 100% leakage savings despite its larger performance impact. As activity decreased, the hybrid model more quickly approached the energy efficiency of NEMS switches.

Chapter 6

Conclusion

This study sought to analyze the utility of hybrid MOSFET-NEMS power gating. Devices can spend the majority of their time in the idle state but when on, devices should operate with a minimal negative impact. Making efficient use across a range of execution patterns is difficult as current sleep transistors are limited in their abilities to power gate at a fine-grained level. Hybrid power gating is a proposed solution to capture the best elements of each type of both MOSFETs and NEMS.

To see how hybrid power gating compares to its single-transistor competition, different applications were run against various power gating hardware. MOSFETs operated well for devices with a large amount of transistor switching, where applications more heavily used the macro being gated. NEMS switches worked well when hardware operated with extended idle states. The effectiveness of different sleep transistors were determined through measured

performance, energy, and lifetime.

Results showed that hybrid power gating was unmatched in its flexibility. It consistently showed energy savings near the optimal points for all applications while performance remained low. Despite this, there were points where hybrid power gating performed worse than a single sleep transistor in terms of energy savings or performance but in those areas, the difference is minimal. Particularly because the applications tested focused on worst case scenarios, it is expected that hybrid power gating will perform better than presented in the average case. This is especially true due to the trend of increasing idle periods of low power devices. The results seen from applications in the MiBench benchmark suite and the coarse-grained analyses reinforced this conclusion. The following list provides a summary of the results with respect to performance, energy savings, and sleep transistor lifetime.

1. **Performance:** MOSFETs consistently offered minimal performance losses while NEMS switches performed significantly worse across applications. Hybrid operated with a similar performance to MOSFETs.

2. **Energy Savings:** MOSFETS saved significantly less energy than NEMS switches across all benchmarks and power gating schemes. The hybrid model saved slightly less energy than both MOSFETS and NEMS alone for high activity applications due to its increased V_{DS}. As application FPU activity decreased, the hybrid model more closely matched a NEMS switch, savings up to 100% of energy.

3. **Sleep Transistor Lifetime:** MOSFETs last indefinitely. A NEMS switch lasted

hours in the worst case situation and days for the best case while still providing energy savings for applications with significant FPU activity. For the same applications, the lifetime of the NEMS switch in a hybrid model is extended by over 10 times that of a NEMS switch by itself. It was also shown that the NEMS lifetime can be further extended to many years with minimal reductions in energy savings.

Incorporating hybrid power gating into existing and emerging architectures provides increased energy efficiency for minimal performance loss regardless of the types of applications a user might run. The device may also be configured to last for the lifetime of the device, no matter the workload. With a lower energy consumption, battery powered devices will last longer and electric costs will be reduced. Furthermore, there is less stress on a system in terms of heat generated and current drawn through power wires.

6.1 Limitations

Despite successfully analyzing the impact of hybrid power gating hardware, there are certain limitations present throughout this work. The largest limitation is that a small subset of benchmarks are run against these systems. Running different types of applications in different suites may offer more variety and a deeper analysis of average case performance. The benchmarks that were executed were chosen specifically for their use of the FPU and successfully showed how different workloads affect the results. They do not, however, provide a comprehensive review of all the types of applications device might encounter.

Another limitation of the work is that the attributes for each type of NEMS switch were taken from various reports and do not necessarily reflect wholly realistic values. As mentioned in Section 3.1.2, NEMS processes may vary by wide margins. As such, it is difficult to provide an accurate realization of a general NEMS switch. Instead, looking at very specific processes for a single type of NEMS switches might be more appropriate to gauge its viability in power gating.

6.2 Future Work

Research regarding hybrid power gating can be expanded by focusing on additions to power gating hardware, leveraging software developer knowledge, and putting theoretical work into production.

The proposed power gating schemes can be modified and built up to be made more efficient. They are currently simplistic models and only operate on a basic history of temporal locality to judge when a circuit should be switched off. Schemes also only monitor individual functional units and have no knowledge of other units or instruction execution patterns as a whole. Smarter schemes potentially save more energy while reducing the number of times a sleep transistor will switch. They can be further tied to the properties of the switches themselves to guide when to provide shallow sleep or deep sleep.

Software can also assist power gating decision within a scheme. With special power gating instructions, developers can inject commands to shut down units if it is clear that they will

be unused for large amounts of time. These commands may be injected explicitly from developers themselves or implicitly by compilers. If a scheme receives a special instruction, it would shut the macro off instead of waiting for more time to elapse. This saves energy that may be otherwise impossible to capture. The developer or compiler can analyze the entire instruction stream and make decisions that may not be done dynamically. The compiler can be further utilized by reordering instructions in a manner that lends itself well to power gating. When considering a power gated FPU, if floating point instructions may be reordered next to each other, the compiler may make the decision to do so. This reordering aims to free up more sleep cycles and has the potential to eliminate an FPU wake up.

Lastly, building a functional unit with the proposed power gating hardware would provide the most accurate test as to the feasibility of hybrid power gating. Despite using McPAT and gem5, some of the most accurate available architecture simulators, they are poor substitutes when compared to real measurements. The inputs to the simulators, the properties of the switches themselves, may also vary with real production. The true properties of MOSFETs and NEMS cannot be completely known until they are put into construction and tested.

Every piece of future work attempts to further reduce subthreshold leakage power by having smarter systems and more accurate analyses. Hybrid power gating reduces leakage power beyond what a MOSFET can provide and makes the inclusion of a NEMS switch realistic. For low power devices, saving energy is of the utmost importance and hybrid power gating lends the flexibility to save large amounts of energy in any environment.

Bibliography

[1] Christopher L. Ayala, Daniel Grogg, Antonios Bazigos, Simon J. Bleiker, Montserrat Fernandez-Bolaos, Frank Niklaus, and Christoph Hagleitner. Nanoelectromechanical digital logic circuits using curved cantilever switches with amorphous-carbon-coated contacts. *Solid-State Electronics*, 113:157 – 166, 2015. Selected papers from {ESSDERC} 2014.

[2] Christian Bienia. *Benchmarking Modern Multiprocessors*. PhD thesis, Princeton University, January 2011.

[3] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, and Kai Li. The parsec benchmark suite: Characterization and architectural implications. Technical Report TR-811-08, Princeton University, January 2008.

[4] Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K. Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R. Hower, Tushar Krishna, Somayeh Sardashti, Rathijit Sen, Korey Sewell, Muhammad Shoaib, Nilay Vaish, Mark D. Hill, and David A. Wood. The gem5 simulator. *SIGARCH Comput. Archit. News*, 39(2):1–7, August 2011.

[5] David A Czaplewski, Gary A Patrizi, Garth M Kraus, Joel R Wendt, Christopher D Nordquist, Steven L Wolfley, Michael S Baker, and Maarten P de Boer. A nanomechanical switch for integration with cmos logic. *Journal of Micromechanics and Microengineering*, 19(8):085003, 2009.

[6] H. F. Dadgour and K. Banerjee. Hybrid nems-cmos integrated circuits: a novel strategy for energy-efficient designs. *IET Computers & Digital Techniques*, 3(6):593–593, 2009.

[7] X. L. Feng, M. H. Matheny, C. A. Zorman, M. Mehregany, and M. L. Roukes. Low voltage nanoelectromechanical switches based on silicon carbide nanowires. *Nano Letters*, 10(8):2891–2896, 2010. PMID: 20698601.

[8] Daniel Grogg, Ute Drechsler, Armin Knoll, Urs Duerig, Yu Pu, Christoph Hagleitner, and Michel Despont. Curved in-plane electromechanical relay for low power logic applications. *Journal of Micromechanics and Microengineering*, 23(2):025024, 2013.

[9] Matthew R Guthaus, Jeffrey S Ringenberg, Dan Ernst, Todd M Austin, Trevor Mudge, and Richard B Brown. Mibench: A free, commercially representative embedded benchmark suite. In *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pages 3–14. IEEE, 2001.

[10] Yuhei Hayamizu, Takeo Yamada, Kohei Mizuno, Robert C. Davis, Don N. Futaba, Motoo Yumura, and Kenji Hata. Integrated three-dimensional microelectromechanical devices from processable carbon nanotube wafers. *Nature Nanotechnology*, 3(5):289–94, 05 2008. Copyright - Copyright Nature Publishing Group May 2008; Last updated - 2014-03-19.

[11] Michael Henry, Meeta Srivastav, and Leyla Nazhandali. A case for nems-based functional-unit power gating of low-power embedded microprocessors. In *Proceedings of the 48th Design Automation Conference*, DAC '11, pages 872–877, New York, NY, USA, 2011. ACM.

[12] Zhigang Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose. Microarchitectural techniques for power gating of execution units. In *Low Power Electronics and Design, 2004. ISLPED '04. Proceedings of the 2004 International Symposium on*, pages 32–37, Aug 2004.

[13] V. Khandelwal and A. Srivastava. Leakage control through fine-grained placement and sizing of sleep transistors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(7):1246–1255, July 2007.

[14] Sang Wook Lee, Dong Su Lee, Raluca E. Morjan, Sung Ho Jhang, Martin Sveningsson, O. A. Nerushev, Yung Woo Park, , and Eleanor E. B. Campbell. A three-terminal carbon nanorelay. *NANO LETTERS*, 4(10):2027–2030, 2004.

[15] Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen, and Norman P. Jouppi. The mcpat framework for multicore and manycore architectures: Simultaneously modeling power, area, and timing. *ACM Trans. Archit. Code Optim.*, 10(1):5:1–5:29, April 2013.

[16] Owen Y Loh and Horacio D Espinosa. Nanoelectromechanical contact switches. *Nature nanotechnology*, 7(5):283–295, 2012.

[17] Changbo Long and Lei He. Distributed sleep transistor network for power reduction. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 12(9):937–946, Sept 2004.

[18] M. N. Lovellette, A. B. Campbell, H. L. Hughes, R. K. Lawerence, J. W. Ward, M. Meinhold, T. R. Bengtson, G. F. Carleton, B. M. Segal, and T. Rueckes. Nanotube memories for space applications. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 4, pages 2300–2305 Vol.4, March 2004.

[19] S. Ning, T. O. Iwasaki, K. Shimomura, K. Johguchi, E. Yanagizawa, G. Rosendale, M. Manning, D. Viviani, T. Rueckes, and K. Takeuchi. Investigation and improvement of verify-program in carbon nanotube-based nonvolatile memory. *IEEE Transactions on Electron Devices*, 62(9):2837–2844, Sept 2015.

[20] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand. Leakage current mechanisms and leakage reduction techniques in deep-submicrometer cmos circuits. *Proceedings of the IEEE*, 91(2):305–327, Feb 2003.

[21] Kaijian Shi and David Howard. Challenges in sleep transistor design and implementation in low-power designs. In *Proceedings of the 43rd Annual Design Automation Conference*, DAC '06, pages 113–116, New York, NY, USA, 2006. ACM.

[22] R. F. Smith, T. Rueckes, S. Konsek, J. W. Ward, D. K. Brock, and B. M. Segal. Carbon nanotube based memory development and testing. In *Aerospace Conference, 2007 IEEE*, pages 1–5, March 2007.

[23] K. Usami, T. Hashida, S. Koyama, T. Yamamoto, D. Ikebuchi, H. Amano, M. Namiki, M. Kondo, and H. Nakamura. Adaptive power gating for function units in a microprocessor. In *Quality Electronic Design (ISQED), 2010 11th International Symposium on*, pages 29–37, March 2010.

[24] Kai-Chiang Wu, Ing-Chao Lin, Yao-Te Wang, and Shuen-Shiang Yang. Bti-aware sleep transistor sizing algorithm for reliable power gating designs. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 33(10):1591–1595, Oct 2014.

[25] Bin Xue. Formal approaches to globally asynchronous and locally synchronous design, 2011.

[26] Zhao Yapu. Stiction and anti-stiction in mems and nems. *Acta Mechanica Sinica*, 19(1):1–10, 2003.