Opportunities for Fine-Grained Adaptive Voltage Scaling to Improve System-Level Energy Efficiency

Ben Keller
Borivoje Nikolic, Ed.
Krste Asanović, Ed.

Acknowledgement

The author would like to thank the many students, faculty, and staff of the ASPIRE Lab and the Berkeley Wireless Research Center for their hard work and assistance. This work was funded by the National Science Foundation Graduate Research Fellowship, the NVIDIA Graduate Fellowship, and Intel ARO.

Opportunities for Fine-Grained Adaptive Voltage Scaling to Improve System-Level Energy Efficiency

by Benjamin Keller

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Borivoje Nikolić
Research Co-Advisor

Date

* * * * * *

Professor Krste Asanović
Research Co-Advisor

Date

Contents

List of Figures

Abstract

Dynamic voltage and frequency scaling (DVFS) can yield significant energy savings in processor SoCs. New integrated voltage regulator technologies enable fine-grained DVFS, in which many independent voltage domains switch voltage levels at nanosecond timescales to save energy. This work presents an overview of DVFS techniques and an energy model by which to evaluate them. Applying the energy model to traces of a cycle-accurate processor system simulation predicts energy savings of up to 53% from the application of fine-grained DVFS techniques. Further exploration and implementation of these technologies has the potential to dramatically reduce energy consumption in future systems.

Chapter 1

Introduction

Energy efficiency is the most important figure of merit for modern digital designs. Thermally limited systems, such as servers and warehouse-scale computers, already consume the maximum amount of power permitted by their enclosures, so the only way that their computational throughput can be increased is by decreasing the amount of energy required for each unit of computation. Energy-limited systems, such as laptops or smartphones, are constrained in the amount of computational work that they can complete by a fixed amount of energy available to the system, typically supplied by a battery. As the energy density of batteries is improving only slowly, the only way to increase the amount of work available to these systems is to decrease the amount of energy required for the same amount of work. Thus, improvements in energy efficiency will be the key driver of performance improvements in nearly all computing devices.

Voltage scaling is a simple and powerful technique to save energy in digital circuits. Decreasing the voltage of digital logic slows the circuit, but it decreases its power consumption even more, leading to a dramatic overall decrease in energy consumption. However, many systems require fast circuits for high performance in some situations, so these systems tend to operate under increased voltage to speed computation, even at the expense of energy efficiency. Dynamic voltage and frequency scaling (DVFS) speeds logic when high performance is required, but reduces the voltage when possible to improve energy efficiency. Most modern processors employ some form of DVFS to improve their system energy efficiency.

System designers face many design decisions when implementing DVFS. Foremost among these are the *granularity* of the available scaling. Spatial granularity is determined by the size of each independent voltage domain. Smaller voltage domains increase the number that can be individually subjected to DVFS. Temporal granularity is determined by the minimum time required to switch between voltage modes; finer temporal granularity implies faster switching between levels. Decreasing the spatial and temporal granularity of DVFS increases the effectiveness of DVFS at improving system energy efficiency, but each is associated with various drawbacks as well. Understanding these tradeoffs is the key to designing an energy-efficient system.

This report examines the implementation options for DVFS. Chapter 2 provides a general overview on the theory of voltage scaling in digital circuits. This is followed by a broad survey in Chapter 3 of the design space for DVFS, particularly in light of recent improvements in the performance of integrated voltage regulators in modern systems. Chapter 4 uses these design options as a starting point to examine possible implementations of fine-grained DVFS, both in space and in time. Finally, Chapter 5 presents the results of a limit study that uses both theoretical modeling and FPGA-based simulation to estimate the potential energy savings of DVFS in a target system. Taken together, the report serves as a starting point for the goal of determining the best use of DVFS on a modern system on chip (SoC).

Chapter 2

Fundamentals of Voltage Scaling

Voltage scaling to save energy is a well-understood technique that was proposed more than two decades ago [1]. This section reviews the fundamentals of voltage scaling as a means to save energy in modern SoCs.

2.1 Digital Circuit Power

The power consumed by a digital gate in operation can be broken into two components: dynamic power and static power. Dynamic power is the power consumed by circuit switching that charges a load capacitance (typically the gate or drain capacitance of one or more transistors) and then discharges it to ground. If a digital block is operating at some frequency f, and a gate within that block has a probability α of switching from low to high each cycle, then its average dynamic power consumption P_{dyn} of that gate is given by

$$P_{dyn} = \alpha C V^2 f, \tag{2.1}$$

where V is the supply voltage and C is the total capacitance of the load. (Note that this analysis neglects short-circuit power incurred in the brief period in which the pull-up and pull-down networks of the gate are both partially ON. This power is typically a small percentage of total dynamic power consumption in modern processes.) This equation pertains to a single gate; activity-capacitance products can be summed across an entire digital block to determine the average dynamic power consumption of that block. Finding the total load capacitance of the block, both internally and externally, and multiplying by the average activity of each node in the block serves as a reasonable approximation for this sum.

Static power is power consumed by a digital gate regardless of whether it is switching. The primary component of static power is typically subthreshold leakage current that flows from the drain of a transistor to its source even when its gate is OFF. Although current decreases exponentially when the gate voltage is below the threshold voltage V_t, it does not decrease to zero. If V_{gs} and V_{sb} are 0, then subthreshold current is exponentially dependent on supply voltage V_{DD} due to the drain-induced barrier lowering caused by the large drain-source voltage across a short channel:

$$I_{sub} \propto e^{kV_{DD}} \tag{2.2}$$

Decreasing supply voltage therefore causes leakage to decrease dramatically. Decreasing supply voltage to zero eliminates static power entirely, as there are no longer any voltage differentials to induce leakage currents. This is the basis of power gating.

While static power depends strongly on V_{DD}, the simple exponential relation above is complicated by a variety of factors and is not useful as an analytical model. At low voltages, junction leakage cannot be ignored, and the magnitude is exacerbated by the onset of gate-induced drain leakage (GIDL) in heavily doped drains. Gate leakage may be a concern as well, although this effect has been mitigated by the widespread adoption of high-K metal gate transistors. Leakage is also exponentially dependent on temperature, and can change by an order of magnitude over the operating range of a chip. Furthermore, design decisions can change the amount of static power consumption in different parts of the design. For example, stacking two transistors reduces leakage through the pair by an order of magnitude, and non-critical gates can use high-threshold devices to reduce leakage selectively. Since leakage is also state-dependent, it is even difficult to make assumptions about the leakage through a gate at any particular time. Analyzing the leakage reported from a processor in Intel's 32nm process [2] reveals a strong dependence of static power on voltage, but the data provided fits a cubic function as well as an exponential one. Static power decreases more-than-linearly with supply voltage, but a more explicit model depends on the details of process, implementation,

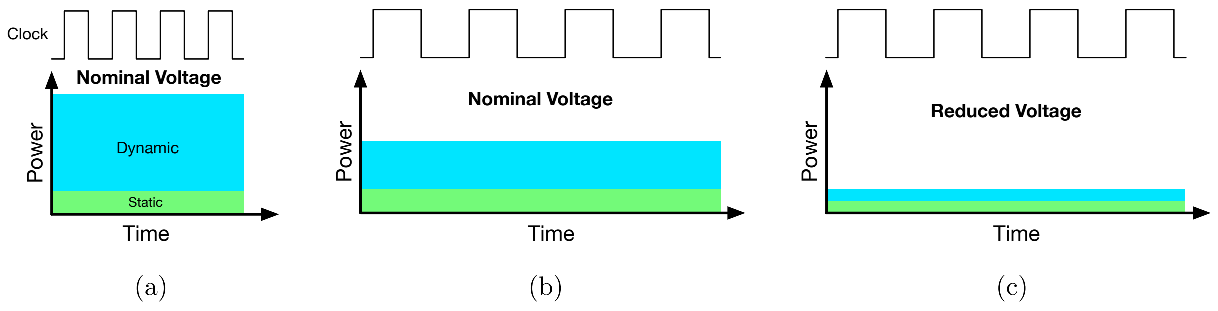


Figure 2.1: An illustration of the efficacy of frequency scaling compared to voltage scaling. Frequency scaling (b) reduces power compared to the nominal (a), but increases total energy, while scaling frequency and voltage together (c) reduces power and energy.

and operating conditions.

2.2 Power and Energy

It is important to distinguish the power consumption of digital circuits, which is measured per unit time, from the energy consumption, typically measured per unit of useful computation completed. Reducing the frequency of operation of a circuit while holding all other parameters constant reduces its dynamic power (it does not substantially affect its static power). However, reducing frequency also proportionally reduces the amount of work completed per unit time. Therefore, the dynamic energy expended to complete the same amount of work does not change with a reduction in frequency. Because the circuit leaks for a longer time while completing the same amount of work, the total energy consumed by the system actually increases, even as power decreases (see Figure 2.2). Frequency scaling is not a useful technique to increase energy efficiency, although some systems employ dynamic frequency scaling (DFS) to temporarily reduce power in a thermally-limited environment.

The other factors of power consumption affect both power and energy. Reducing activity reduces dynamic power and energy, provided that no additional cycles are needed to complete the same amount of work. This is the basis of clock gating. Reducing load capacitance also reduces dynamic power and energy, although this must be done at design time. Most importantly for this work, reducing the supply voltage dramatically reduces both dynamic power and energy. Static power is reduced as well, although the situation for static energy is somewhat more complicated.

In order to determine the dynamic power and energy savings from voltage scaling, we determine the dependence of operating frequency on supply voltage. Digital logic switches more quickly at high voltages and more slowly at lower voltages because the amount of current that a transistor can source is a function of V_{gs} and V_{ds}. Unfortunately, long-channel analytical models of transistors become much more complicated when short-channel effects such as velocity saturation are taken into account. One common simplification uses an "alpha-power" law [3] that is a compromise between the square-law dependence of current on gate voltage predicted by long-channel transistor models and the linear relationship observed in a fully velocity-saturated device:

$$I_{ds} \approx k(V_{gs} - V_t)^{\alpha}, \tag{2.3}$$

where α is between 1 and 2. The delay τ of a digital gate is proportional to its supply voltage and the current through its ON transistors. Using the alpha-power law above, we find

$$\tau = \frac{CV_{DD}}{I_{ds,\mathrm{ON}}} = k\frac{CV_{DD}}{(V_{DD} - V_t)^{\alpha}} \tag{2.4}$$

with V_{gs} set to V_{DD} because the transistors are ON. Inverting delay gives a relationship between maximum frequency and voltage for a single gate:

$$f = k\frac{(V_{DD} - V_t)^{\alpha}}{CV_{DD}} \tag{2.5}$$

This relationship also holds for the frequency of a digital block composed of many such gates. Note that we assume that the transistors are operating above the threshold region.

Although the alpha-power model is itself a simplified approximation, it still does not lend itself to simple hand analysis. Fortunately, if $\alpha \approx 1.5$ then the frequency-voltage curve is very nearly linear, as can be seen in Figure 2.2. α tends to be between 1.1 and 1.4 for modern processes; nonetheless, this nearly-linear trend is observed in recently-published data for fully-realized processors (see Figure 2.2). This discrepancy can be reconciled by allowing V_t to vary as another parameter in the model, which also results in a better overall fit. Accordingly, we use the linear model

$$f = k(V_{DD} - V_t) \tag{2.6}$$

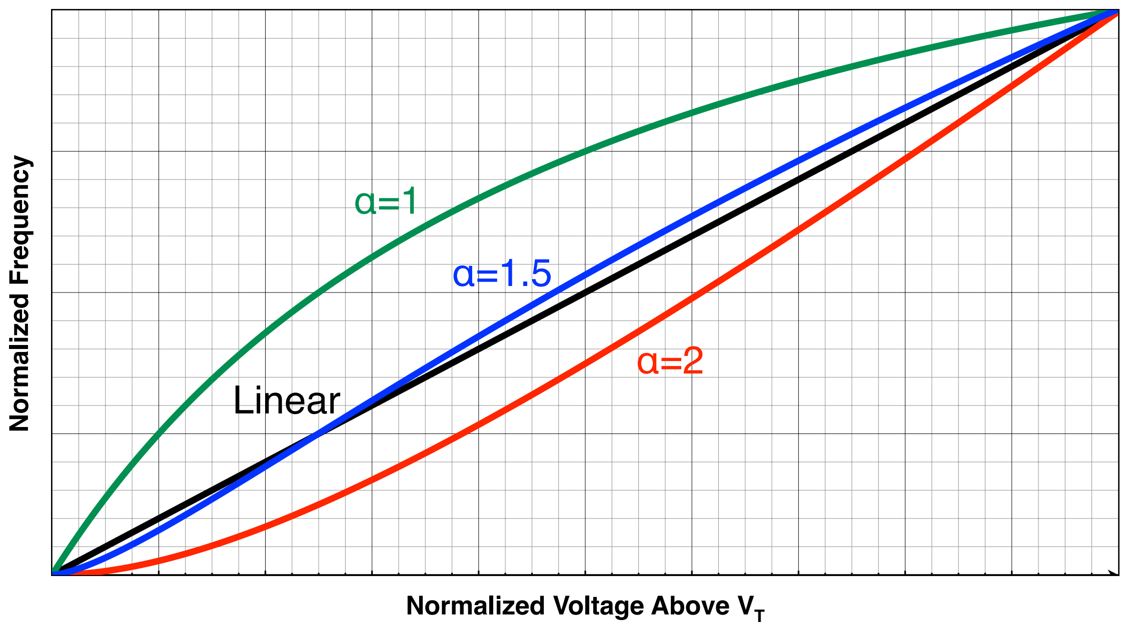


Figure 2.2: Various alpha-power law frequency-voltage curves compared to the linear approximation. When α is near 1.5, the frequency-voltage relationship is well-approximated by a straight line.

to determine the necessary change in operating frequency when voltage is scaled. This approximation is only valid when V_{DD} is substantially greater than V_t; as V_{DD} approaches V_t, the operating frequency slows dramatically.

Combining Equations 2.1 and 2.5 results in the fundamental relationship governing dynamic power as a function of voltage:

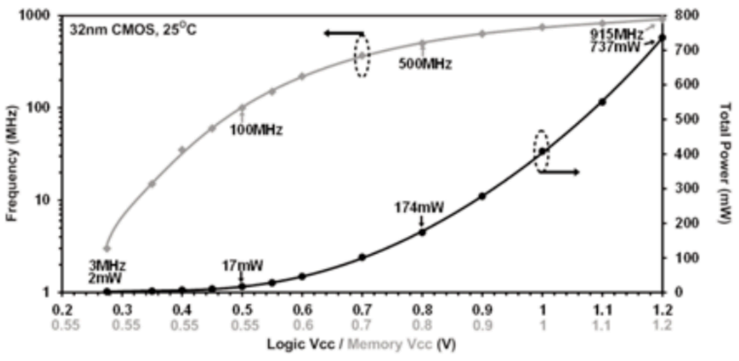
$$P_{dyn} = \alpha C V^2 f \approx \alpha C V^2 \cdot k(V - V_t) \propto V^3 \tag{2.7}$$

Energy is power consumed over some time period; this period is determined by the frequency. Slower frequencies require longer time of operation to complete the same work, so a factor of frequency must be divided out of the power relation to find the dynamic energy as a function of voltage:

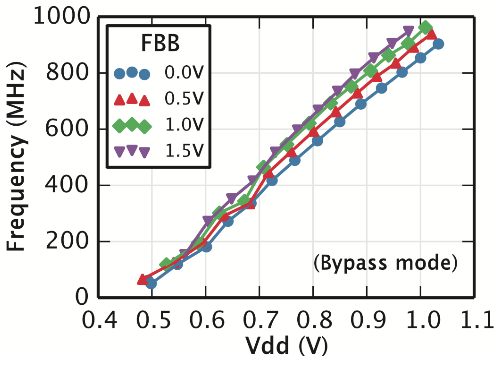
$$E_{dyn} = P_{dyn}/f = \alpha C V^2 \propto V^2 \tag{2.8}$$

Both instantaneous power and the energy consumed for a given unit of work are reduced super-linearly with supply voltage reductions, although the power savings are more dramatic.

As we have seen, it is difficult to analytically relate static power to V_{DD}. We expect super-linear static power savings as supply voltage decreases, particularly at higher V_{DD}. Since frequency scales roughly linearly with voltage, this would imply that static energy also decreases as voltage is scaled down, since the static power savings should outweigh the extra time needed to complete the same amount of work. In practice, this may not always be the case, and the details depend heavily on process technology. In [2], a processor fabricated in 32nm bulk CMOS reduced static energy by a factor of three when scaling from 1.2V to 0.7V. However, supply voltages lower than 0.7V did not result in additional savings; the static power savings are not as significant, and the cycle time begins to increase significantly as the supply voltage approaches the threshold. Near or below the threshold voltage, static power continues to decrease slowly, but static energy increases dramatically as cycle time increases exponentially in the subthreshold region.



(a)



(b)

Figure 2.3: Measured voltage-frequency curves from modern processes; (a) is measured in a 32nm bulk process [2] and (b) is measured in a 28nm FD-SOI process [4]. Each shows a roughly linear relationship between voltage and frequency (note that the frequency scale in (a) is logarithmic).

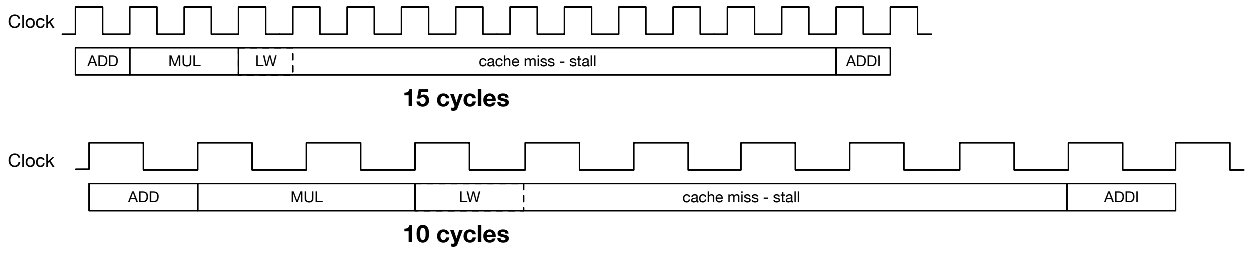


Figure 2.4: An example of the cycle count changing as voltage is scaled. A slower frequency lessens the cycle count of a fixed-latency cache miss.

2.3 Voltage Scaling and Cycle Count

In the above analysis, we have assumed that the number of cycles required to complete a given amount of work is fixed as the supply voltage changes. In practice, this may not be the case, because the system under voltage scaling is interacting with other systems that may remain at a fixed frequency, or undergo a different scaling regimen. For example, a processor interacting with a fixed-frequency memory system will see lower memory latency (in processor cycles) if it operates under a reduced supply voltage, because each of its cycles will take longer. This means that the processor will likely take fewer total cycles to complete the same amount of work, because it will spend fewer cycles waiting for the memory system (see Figure 2.4). Because processors frequently stall waiting for slower I/O or memory systems, this effect generally occurs to some degree.

The effect of frequency reduction on the cycle count means that the energy savings of voltage scaling will tend to be larger than an analysis using constant cycle counts would suggest. Because the reduction in cycle count is primarily a reduction in idle time, this effect tends to be especially pronounced when accounting for total static energy. In the previous section, we observed that static energy may not be reduced much as voltage scales if the cycle count remains constant. In practice, these savings are improved because the number of cycles required to complete the work is reduced. While some prior work focuses on energy per cycle as the energy metric of interest [5], the most useful energy metric is energy required to complete a unit of work, regardless of cycle count. This metric will be used whenever possible in analysis and measurements.

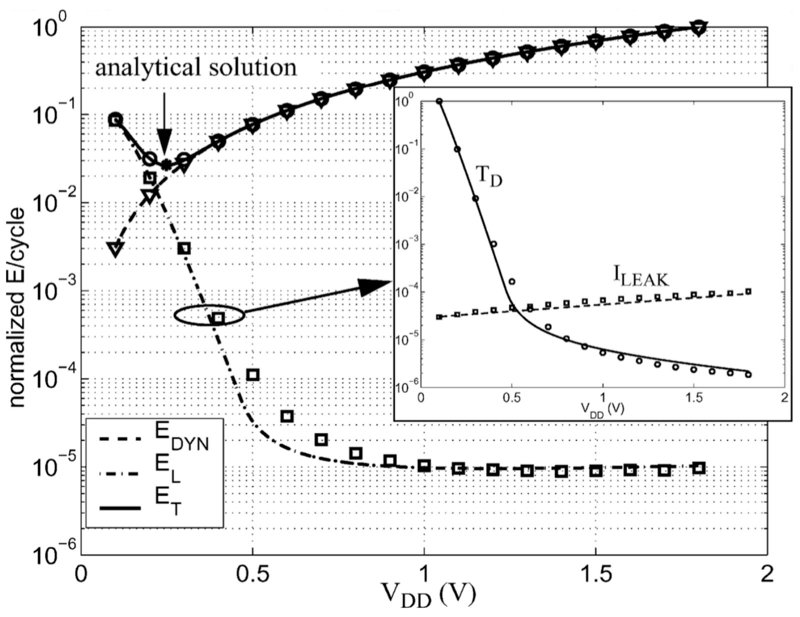


Figure 2.5: Analytical modeling and simulation of the leakage and dynamic energy consumption of an FIR filter (reprinted from [5]). The minimum energy point falls at 250mV, well below the threshold voltage for the modeled process.

2.4 Subthreshold Operation

Static CMOS gates continue to function correctly well below the threshold voltage of the transistors used to compose them. As noted in Section 2.2, frequency decreases dramatically as supply voltage approaches the threshold, which means that static energy increases even as dynamic energy decreases. Eventually, static energy dominates the total energy consumption of the circuit. At this point, further voltage reductions increase the total amount of energy because the circuit takes so much longer to complete the same amount of work that more static power is consumed over that period. Therefore, a minimum energy point must exist, the supply voltage that minimizes energy per unit work. A first-order analysis typically places this minimum energy point in the near-threshold or subthreshold region (see Figure 2.5).

Based on this analysis, it would seem that the most energy-efficient digital designs ought to operate only at this minimum energy point. However, there are many challenges to designing functional circuits in the subthreshold operating region. Typical SRAMs are not CMOS circuits, and rely on ratioed devices within each bitcell for correct operation. Most conventional SRAMs do not operate correctly near the threshold; special macros must be designed, often with considerable area overhead. On-chip variation between devices, already an issue at typical supply voltages, becomes much more severe as the supply voltage is reduced, requiring large margins or complicated adaptation techniques. Standard VLSI toolflows are also not well-suited to subthreshold design: standard-cell libraries are often not characterized at low voltages, and often must be pruned of cells that will not function well in this regime. Power gating and clock distribution pose their own challenges. The additional design effort required to implement subthreshold designs and the various overheads imposed by coping with these challenges are generally not worth the energy savings gained by subthreshold operation.

Furthermore, finding and operating at the minimum energy point can be quite difficult in real designs. For example, [6] found that common benchmarks and voltage regulator inefficiencies both increased the actual minimum energy point of operation. When taken together, these effects actually increased the minimum energy point well above the threshold voltage. While subthreshold operation may pay off for certain extremely low-activity workloads, there is little evidence that operating designs in the subthreshold region saves energy for most computing platforms. For these reasons, we do not consider subthreshold and near-threshold operation as viable options to save energy in this work.

Chapter 3

The DVFS Design Space

The energy savings that can be enabled by operating at lower voltages make voltage scaling an attractive option. Systems that require energy-efficient operation and have lax performance constraints can permanently operate at a reduced voltage to trade performance for energy savings. However, many systems must be able to operate with high performance at least some of the time. These systems can operate at a high nominal voltage and frequency when performance is critical, and then scale to a lower voltage and frequency when high performance is not needed. Furthermore, some systems can perform this dynamic voltage and frequency scaling at a block level, scaling the voltage down on less critical blocks while maintaining a high voltage on those blocks that are critical to performance at any given time.

For decades, many system designers were not concerned about power or energy consumption, and systems operated only at a high nominal voltage to achieve the best performance. Today, thermal challenges and mobile device limitations make energy efficiency a first-class constraint; some systems aggressively apply DVFS independently to myriad voltage areas according to their workload. Between these two extremes are many different design points that trade off the potential energy savings of DVFS with the complications and complexity introduced by aggressive DVFS schemes. This section examines several of these tradeoffs in turn. To frame exploration of the design space, each tradeoff is considered in the context of two cross-cutting examples: a **conservative** system that implements traditional design decisions common in industrial designs in recent years, and an **aggressive** system that implements cutting-edge ideas to better enable DVFS (see Figure 3.1). These examples make concrete the tradeoffs implicit in energy-saving design in modern SoCs.

3.1 Asynchronous Interfaces

Converting one large voltage area into several smaller voltage areas introduces asynchronous boundary crossings into a system because voltage scaling must be coupled with frequency scaling. If each voltage area can be supplied with its own voltage independent of its neighbors, then the clock periods of neighboring blocks cannot be assumed to have any relationship in frequency or phase. This asynchrony means that the interface must be decoupled, since data cannot necessarily be sent across at every clock edge.

Care must be taken when designing these asynchronous boundary crossings to avoid metastability, which can occur in a flip-flop when its input changes near the clock edge. This timing issue is normally avoided in synchronous design by ensuring that every path between two flip-flops meets some known timing constraint, but because no assumptions can be made about the arrival time of data across the asynchronous boundary, extra circuits must be included to synchronize signals crossing this boundary. Without these circuits, metastability can cause unpredictable and incorrect circuit operation.

Several approaches to protect against metastability are described below. Each approach imposes at least some latency penalty compared to the synchronous interface that would be used if no clock-domain crossing were present. Beyond this latency penalty, clock-domain crossings also incur tradeoffs in complexity and have implications for the design of the larger system. Accordingly, splitting a design into multiple voltage and clock domains must be done thoughtfully. The decoupled interfaces often benefit from first-in, first-out (FIFO) queues that can tolerate mismatch between the data rates of the sender and the receiver. If the design already has synchronous queues that decouple different sub-blocks, defining voltage area boundaries at these existing queues reduces the overhead of this division.

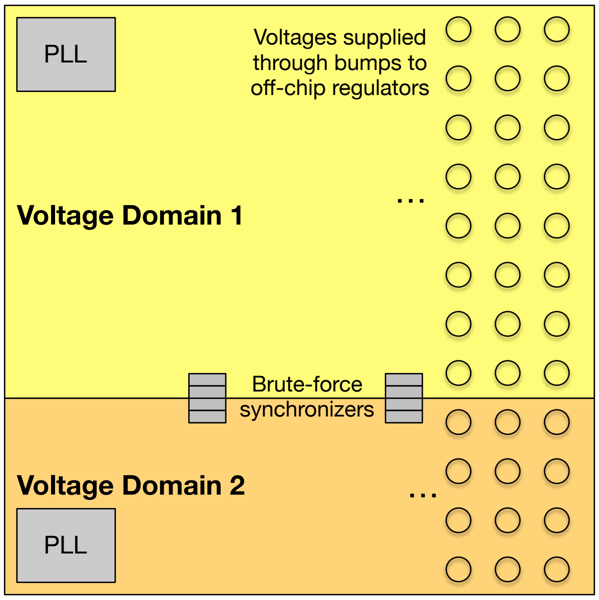
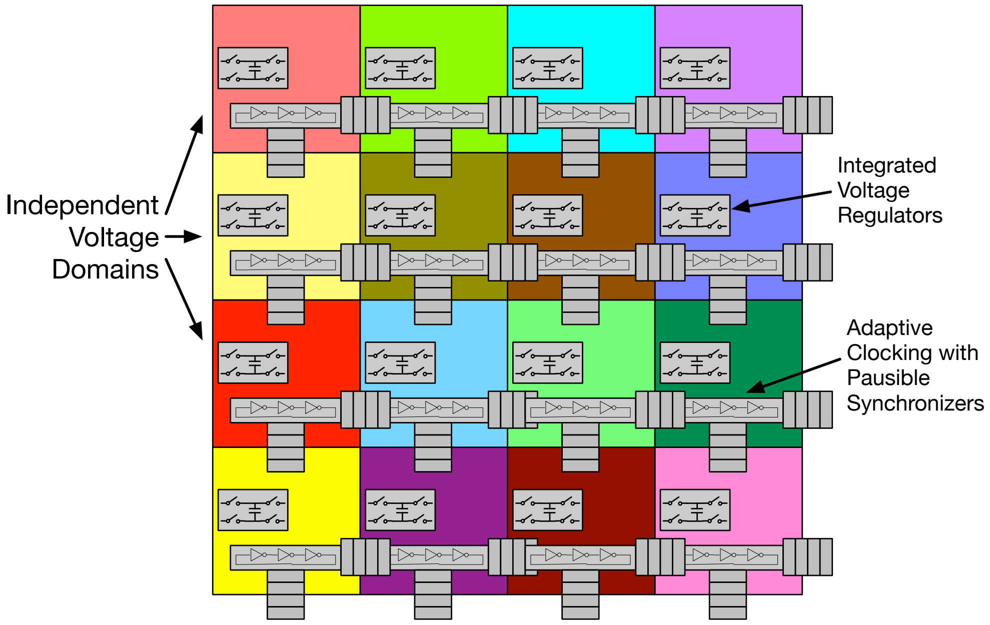

PLL
Voltages supplied through bumps to off-chip regulators
Voltage Domain 1
...
Brute-force synchronizers
Voltage Domain 2
...
PLL

Independent Voltage Domains
Integrated Voltage Regulators
Adaptive Clocking with Pausible Synchronizers

Figure 3.1: The key features of the conservative and aggressive reference systems. The conservative reference system has only two voltage domains, each supplied by an off-chip regulator and clocked from PLLs. The aggressive reference system has many independent voltage domains supplied by integrated regulators and clocked by oscillators that track the local critical path.

3.1.1 Brute-Force Synchronizers

The most common way to cope with the danger of metastability at asynchronous boundary crossings is to place two or more flip-flops in series clocked by the receiving domain for any signal that crosses the boundary. These *brute-force synchronizers* delay the sampling of the signal in the receiving clock domain, so that if the first flip-flop samples a changing value and becomes metastable, it will have one or more extra cycles for this metastability to resolve, preventing its propagation into the rest of the circuit. Metastability is an unstable operating condition, and theoretical and practical studies have shown that metastability duration in flip-flops becomes exponentially less likely over time [7]. Inserting these extra cycles of latency can therefore dramatically reduce the probability of metastability, often to the point where it is vanishingly unlikely over the operational lifetime of the circuit.

Brute-force synchronizers can be combined with a FIFO queue to create a brute-force bisynchronous FIFO as shown in Figure 3.2. (We use the term *bisynchronous* to refer to a circuit that operates with two independent clocks. An *asynchronous* circuit, in contrast, may have no clock at all.) This FIFO can efficiently move entire words across the asynchronous interface, and only the read and write pointers are synchronized via brute-force synchronizers. This approach remains the most commonly used method of moving data between asynchronous clock domains [8], and our conservative reference system uses brute-force synchronizers between its clock domains (see Figure 3.1). The main drawback of brute-force synchronizers is that their effectiveness is based on explicitly inserting latency at the asynchronous interface. The latency of a data word through a brute-force bisynchronous FIFO will be at least the number of series flip-flops in the brute-force synchronizer, typically two in academic contexts [9] but often three or more in industry designs, which must guarantee failure-free operation for tens of millions of flip-flops over the lifetime of hundreds of millions of chips. This high interface latency is a major impediment to splitting a design into many independent voltage areas.

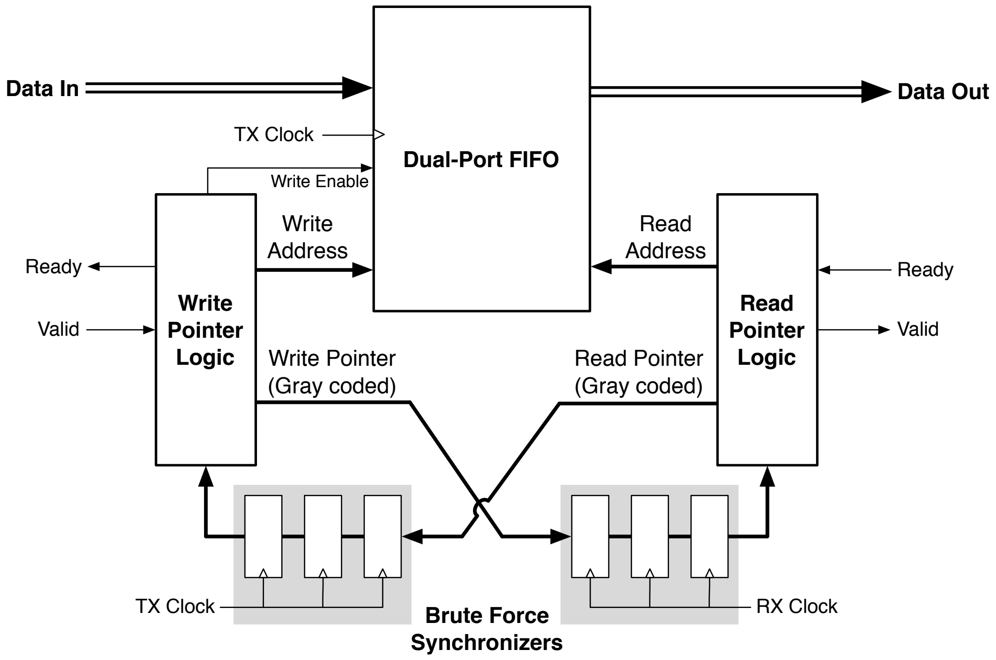


Figure 3.2: A standard brute-force bisynchronous FIFO.

3.1.2 Loosely Synchronous Clocks

One way that designers have dealt with the high latency of asynchronous boundary crossings is to challenge the assumption that the two clocks are fully asynchronous. The authors of [10] describe several different classes of "loosely synchronous" clocks, in which some information about the relative frequency or phase of the two clocks is known. *Mesochronous* clocks have the same frequency, and differ only by a fixed, unknown phase difference. *Ratiochronous* clocks differ in frequency by some fixed integer ratio, and have a predictable phase relationship. *Pleisiochronous* clocks have very slightly different frequencies, causing the relative phase of the two clocks to drift slowly over time. If these relationships between the two clocks are known at design time, brute-force synchronizers may not be necessary: the authors of [11] demonstrate a synchronizer that can pass signals across any of these loosely synchronous clock boundaries with sub-cycle latency. However, while these fast synchronizers are useful for certain applications, they are generally not applicable to the fully asynchronous clock domains associated with fine-grained DVFS, as it is rarely possible to define known relationships between the frequency and phase of these clock domains.

3.1.3 Pausible Clocks

Pausible[1] clocking is a different method to reduce the latency of asynchronous boundary crossings. This approach tightly couples the synchronization circuitry with the circuit used to generate the clock for each clock domain. If an asynchronous signal switches at a time that would cause metastability, the clock is paused and the next clock edge delayed until the signal has finished switching. As shown in Figure 3.3, these pausible synchronizers can be incorporated into a *pausible bisynchronous FIFO* that can synchronize data across a fully asynchronous interface with low latency [12]. Unlike brute-force synchronizers, these circuits completely eliminate the possibility of metastability, rather than reducing the probability until it is negligible. However, because of their integration with the clock generator, pausible clock circuits must be designed carefully to meet several timing constraints, including constraints on the minimum clock period and maximum insertion delay of the clock domain. Only if these constraints are reasonable for the application can pausible clocks be used to reduce interface latency. Our aggressive reference system implements fast synchronization between voltage domains via pausible clocks (see Figure 3.1).

3.2 Clocking

Fully synchronous designs that operate at a fixed voltage only need to generate a single clock at a fixed frequency. This clock is traditionally generated by a phase-locked loop (PLL) before being distributed via a clock tree to the entire system. PLLs are the subject of a diverse array of research to improve their accuracy and resiliency and reduce their system footprint. For instance, IBM augmented the PLL in its POWER7 microprocessor with a skitter circuit that measures the phase of the clock and gives warning when it approaches some timing boundary [13]. Digital control can then adjust the clock to avoid the margin hazard. Other designers have focused on the implementation of all-digital PLLs [14] or phase-picking DLLs [15]. Nonetheless, these reference-based clock generators tend to require considerable area and power, and so it is generally impractical to scale such typical clock-generation techniques

[1]The correct spelling is "pausable", but we use the spelling consistent with the literature here.

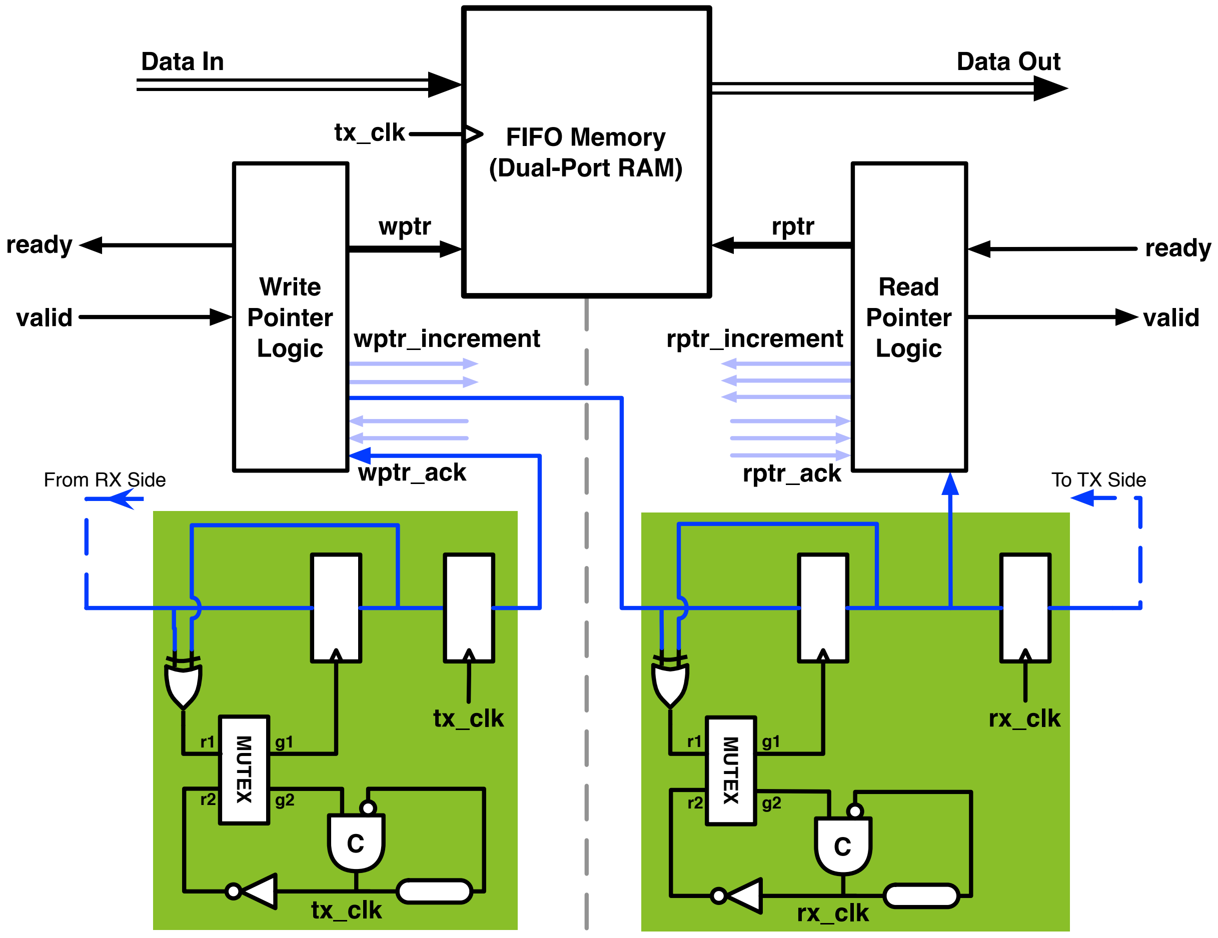


Figure 3.3: A pausible bisynchronous FIFO as proposed in [12].

to systems with many independent clock domains.

The globally asynchronous, locally synchronous (GALS) approach to system clocking requires independent clocks for each voltage domain, and these clocks must be able to change frequency as needed as the voltage of domain area changes. One approach to generating the many clocks needed for fine-grained DVFS is to generate a fast global reference clock and then divide the clock as necessary within each voltage area. However, distributing and manipulating a single global clock is not the most efficient way to generate clocks for many voltage areas. Simple local oscillator circuits such as ring oscillators are better suited for distributed clock generation. These circuits can generate the appropriate clock frequency via a lookup table based on the operating voltage. Such lookup tables can be calibrated per chip or even per voltage area to improve their accuracy under process variation. More aggressive schemes use some form of adaptive clocking. Instead of a lookup table, adaptive clocking circuits operate on the same supply voltage as the logic they are clocking, replicating some subset of the critical paths in the circuit and generating clock edges based on the instantaneous operating conditions of the circuit. These adaptive clocking circuits are ideally suited for the many different voltage areas used in fine-grained DVFS, although they do not lock to a fixed frequency and are less predictable than their traditional PLL counterparts.

Clock generators for fine-grained DVFS must also be able to quickly switch between different frequency settings as voltage changes. Adaptive clocks are able to instantaneously react to changes in voltage, as they immediately “sense” the voltage change as a change in the delay through critical-path replicas. Lookup-table-based schemes may take longer to switch between different frequency settings, and may require the clock to be temporarily halted during the transition, as is the case in [16]. PLL clock generators in particular may require many cycles to lock onto a new reference frequency, and so are not well-suited for fast frequency transitions.

While the elimination of a global clock imposes more complexity in clock generation, it also eliminates some of the overheads of clock distribution. The clock tree can be simplified as global clock distribution is no longer necessary, saving power and freeing routing resources. Expensive global clock meshes that may be necessary to distribute a clock globally [17] can be removed. Finally, if a PLL is no longer needed, this may result in savings even if many smaller local clock generators are included instead. These benefits of local clock generation and distribution may offset some of the complexities of its implementation.

The aggressive reference system implements a free-running ring-oscillator-based clock generator that generates a local clock for each voltage area (see Figure 3.1). Each clock generator is tuned to replicate the timing of the local critical paths and automatically adjusts to changes in supply voltage. These clock generators interact with the interface synchronization as described in Section 3.1.3 to implement pausible clocking. By contrast, the conservative reference system uses a PLL to generate a clock for each of its voltage domains; these PLLs can change frequencies but must re-lock to do so, so transitions are slow.

3.3 Power Supply Generation

Generating the required voltage supplies is the key design challenge in implementing fine-grained DVFS. There are two main difficulties in generating these supplies. First, fine-grained DVFS requires that many different voltage supplies be made available at the same time, so that different parts of the chip can be independently supplied. Second, fine-grained DVFS requires that portions of the chip can switch quickly between different voltage levels with minimal overhead. These goals must each be achieved while maintaining high power conversion efficiency, the ratio of power delivered by the regulator compared to the power supplied to it. If the conversion efficiency is too low, the energy losses from the inefficient power delivery circuits may outweigh the savings achieved through DVFS. Fine-grained DVFS is only possible if a high-efficiency power delivery system can supply many different voltages and switch quickly between them.

Traditionally, processor dies are supplied a small number of distinct voltages by off-chip voltage regulators mounted nearby on the circuit board. These regulators can achieve high conversion efficiencies and can source a large amount of current. However, they are only able to switch between modes relatively slowly because of the large time constants of the passives employed in their power conversion. Furthermore, each regulator can typically supply only one voltage, meaning that every additional independent voltage area on-chip would require an additional off-chip regulator to implement fine-grained DVFS. Apart from the overhead in cost and board area, this approach does not scale well to large numbers of on-chip voltage areas because of the limited numbers of bumps available for supply inputs on the processor die. Adding more supplies would decrease the number of bumps available per supply, quickly leading to degradation of the quality of the on-chip supplies. Off-chip regulators are not practical for power delivery for fine-grained DVFS; two alternate approaches, on-chip regulation and multi-rail switching, are described below.

3.3.1 Integrated Regulators

Since off-chip regulators are impractical for fine-grained DVFS, one alternative is to integrate the regulator circuitry on the processor die itself. In these schemes, only a small number of voltages need be supplied to the die, and on-die regulators generate additional voltages to supply different parts of the chip as needed. In addition, on-die regulators tend to be able to switch more quickly than their off-chip counterparts, making them well-suited for DVFS applications.

The simplest on-die voltage regulators are linear regulators (see Figure 3.3.1). These regulators do not require any large passive components or decoupling capacitors and so have little area overhead. They can also switch very quickly between voltage modes. However, the efficiency of these regulators is proportional to the conversion ratio of voltage supplied to voltage delivered. This means that linear regulators are efficient for small steps in voltage but inefficient at delivering voltages much lower than their supply [18].

Switching regulators tend to be more efficient than linear regulators because a series transistor is switched alternately on and off to deliver power, so the switch itself has less overhead than that of a linear regulator. One class of switching regulators is inductor-based and are known as step-down buck converters (see Figure 3.3.1). Buck converters can achieve high conversion efficiencies, but require high-quality inductors that are often not available in standard CMOS processes [20]. Accordingly, these circuits require off-chip inductors to achieve high efficiency. The inductors are typically either mounted nearby on the board or integrated into the package, but either approach mitigates the benefits of integrating the

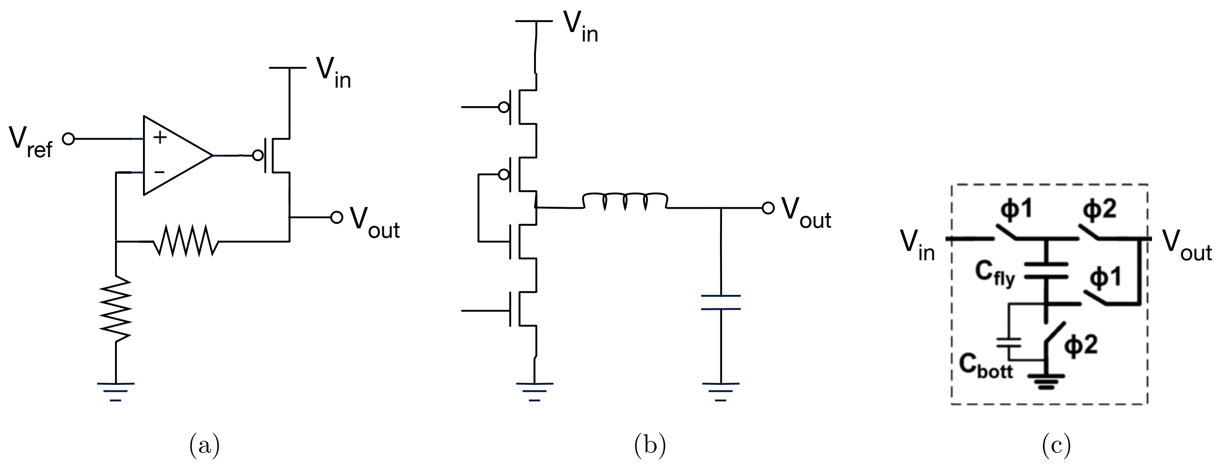


Figure 3.4: Three common types of voltage regulators: (a) linear regulators, (b) buck converters, and (c) switched-capacitor converters [19].

regulators in the first place. A second class of switching regulators are *switched capacitor* (SC) circuits, which do not use inductors (see Figure 3.3.1). SC voltage regulators are usually statically configured to achieve the highest efficiencies at particular conversion ratios [19]. These voltage regulators cannot achieve as high efficiencies as buck converters, but they require only high-density capacitors, which can be fabricated entirely on-die.

Integrated voltage regulators make it easy to supply many different on-chip voltages. However, different designs may suffer from worse efficiency than off-die regulators. Additionally, these designs take up area on the die, particularly those that require large passive components. The power density of regulators is an important metric independent of their efficiency; typically measured in watts per square millimeter, power density is the amount of power that can be delivered by regulators taking up a given amount of die area. One impediment to integrating voltage regulators is that the power spent in conversion, which would otherwise be dissipated by discrete components, now must be fit into the power constraints of the processor die. Integrated regulators also require careful careful floorplanning and implementation of the on-die power grid, increasing the design complexity of the processor die.

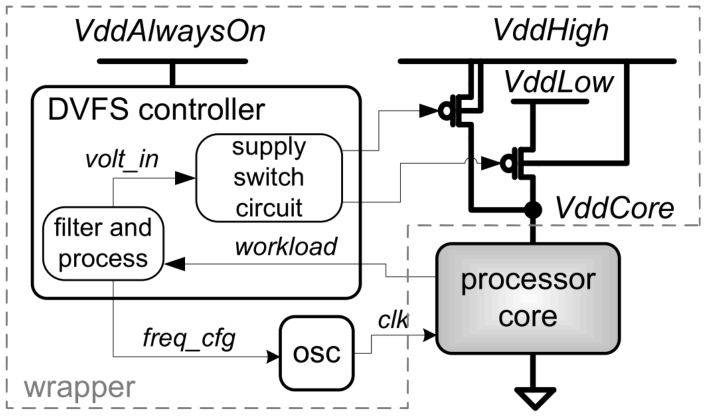


Figure 3.5: The controller for dual-rail supplies implemented in [21]. The logic ensures that the core is supplied by only one voltage at a time.

3.3.2 Multi-rail Switching

Multi-rail switching is a simpler alternative to on-die voltage regulation that still allows for different areas of the chip to switch quickly between voltage modes. In this approach, a small number of different supplies are provided from off-chip and distributed throughout the die. Many implementations use only two supplies, one high voltage and one low voltage [21]. Each voltage area is connected to each rail via a series of power switches in parallel; only the switches connected to a single rail are on at any time. In this way, each voltage domain can independently switch between voltages simply by switching off its connection to one rail and then connecting to the second. This simple scheme requires only header or footer transistors to implement (see Figure 3.5), and can achieve very fast switching between voltage modes.

Despite the simplicity of this approach, it does suffer from several drawbacks that may require additional circuit techniques to address. Two or more supplies must be distributed throughout the chip, which may require additional routing resources to ensure supply integrity. Also, unless care is taken, areas switching between the two supplies may temporarily short them together. This can be mitigated by ensuring that the original supply is completely switched off before connecting to the new one, but this approach may lead to temporary voltage droops while the voltage domain is not connected to any supply. Additionally, these switching events can result in large changes in current loading on the supplies, leading to

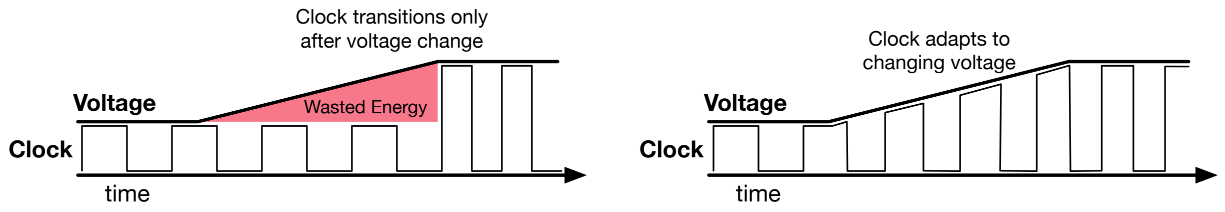

Clock transitions only after voltage change
Voltage
Wasted Energy
Clock
time
Clock adapts to changing voltage
Voltage
Clock
time

Figure 3.6: The energy overhead of slow-transitioning clocks during a voltage transition. If the clock cannot change to a higher frequency until the voltage has settled at the new level, energy losses are incurred. By contrast, adaptive clocks avoid such losses by tracking the voltage as it changes.

large amounts of di/dt noise on the voltage rails. Finally, this system is limited to supplying only a handful of fixed voltages, although this disadvantage can be mitigated by the voltage dithering technique described below.

3.3.3 Mode Switching Overhead

A voltage-mode transition between two different voltage levels inevitably incurs some energy overhead. The amount of overhead is heavily dependent on the clocking and voltage regulation scheme used, but it is important to account for this overhead, as it will offset some of the potential energy savings of DVFS.

One source of overhead is associated with the clocking of the block under DVFS. If the clock must be halted during the DVFS transition (for example, so that a PLL can re-lock), this represents an obvious overhead, but other clocking schemes have overheads as well. If the clock is set to the slowest frequency necessary for correct operation at the lower voltage over the entire duration of the voltage transition, then for much of the voltage transition the circuit will be operating at a slower clock rate than necessary for the instantaneous voltage. This is energy-inefficient because work is being done more slowly than is possible during the transition while consuming power consummate with a higher voltage (see Figure 3.6). Adaptive clocking schemes that actively track the changing voltage during the transition and do not require halting the clock eliminate most of this clocking overhead.

Another source of energy overhead is caused by the power consumption of the circuits that effect the supply voltage change. This overhead is different for different voltage regulation schemes, but switching regulators generally incur some losses when switching between different voltage modes. This energy cost can be caused by non-ideal switches, charge sharing, and additional inductor IR losses. The authors of [22] provide a detailed model of the losses associated with typical modern inductive switching regulators. Other schemes, such as switched-capacitor converters or multi-rail switching, may incur less overhead in switching between voltages. Nonetheless, any analysis of DVFS energy savings must account for the energy overhead of switching between different voltage modes.

3.3.4 Number of Voltage Levels and Voltage Dithering

Many voltage regulation schemes are limited in the number of discrete voltage modes that they can supply. Ideal DVFS would allow continuous voltage scaling so that any operating frequency is matched exactly to the minimum voltage required to operate at that frequency; operating at any higher voltage than necessary incurs energy overhead. Regulation schemes with only a small number of discrete voltage levels, such as integrated switched capacitor voltage regulators or multi-rail switching, can approximate continuous voltage scaling by dithering between two voltages with different duty cycles [23]. Figure 3.7 shows the theoretical efficiencies that can be achieved by dithering between three discrete voltage modes in a multi-rail switching scheme. As shown in the figure, dithering results in only minor energy losses when compared to ideal voltage scaling if enough evenly-spaced discrete voltage modes are available.

Voltage dithering allows circuits to operate at a "virtual" voltage and frequency by operating part of the time at a higher voltage and faster frequency and part of the time at a lower voltage and slower frequency. One design challenge when voltage dithering is deciding how frequently to switch between the two modes in order to maintain the desired duty cycle. As noted above, switching between modes may incur overheads, which would suggest that mode switches should occur as infrequently as possible. (In the limiting case, a voltage area could switch only once between the fast and slow operating points over the duration

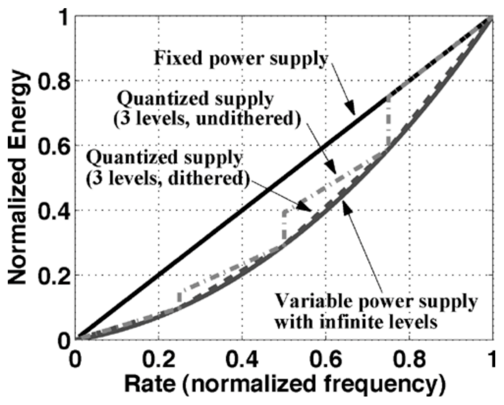


Figure 3.7: The relative efficiency of voltage dithering and continuous voltage scaling (reprinted from [23]). Dithering between three discrete voltage levels is nearly as energy efficient as continuous scaling because it linearizes only small segments of the energy-frequency curve.

of the virtual voltage and frequency operation.) However, switching too infrequently may result in flow-control issues that would not occur if the block were actually running at the virtual operating point. For example, if a producer block runs at the fast operating phase of the dither for too long, it might fill a queue to a consumer block that is operating more slowly, forcing it to stall, while more frequent dithering would maintain the balance of the queue. Some tradeoff is necessary between fealty to the virtual operating condition and the mode-switching inefficiencies of the system; the result of this tradeoff is an increase in the overhead of the dithering approach. Accordingly, a system that can supply a greater number of discrete voltages may be able to achieve higher energy efficiency.

3.3.5 Regulation in the Reference Systems

As shown in Figure 3.1, the conservative reference system supplies its two voltage domains from off-chip regulators, with one regulator needed for each supply. A separate set of IO bumps are reserved for each voltage supply. The off-chip regulators can switch between voltages to implement DVFS, but only at relatively slow timescales.

In contract, the aggressive reference system is supplied with a single, fixed off-chip voltage. Each of its myriad voltage domains converts that voltage into a desired level using integrated switched-capacitor voltage regulators. The integrated regulators can only generate a small number of voltages, so dithering is used to meet intermediate performance-power requirements.

3.4 Additional DVFS Tradeoffs

Design decisions concerning clocking, voltage supply, and synchronization can impact other aspects of system operation and performance. This section explores additional costs and benefits of DVFS implementation in the context of power gating, variability, and design complexity.

3.4.1 Power Gating

Power gating is another strategy for saving energy in SoCs by completely powering off blocks when they are not in use. This technique is generally implemented via a set of header or footer transistors that can isolate a block from the power supply (see Figure 3.8). Various circuit techniques, such as body biasing and MTCMOS, are used to ensure that the power gate transistors have high OFF resistance to minimize leakage but low ON resistance so that supply integrity of the block is minimally impacted during normal operation. In contrast to DVFS, power gating allows individual blocks to be completely powered off when they are not in use, reducing their power consumption to negligible levels.

Like DVFS, power gating introduces additional design complexity when it is implemented in an SoC. Because blocks adjacent to an unpowered block may still be powered, isolation cells must be placed at the voltage area boundary to ensure that invalid logic values do not propagate into operating blocks. Additionally, state elements of blocks that are power gated will not retain their state because flip-flops and SRAMs depend on active feedback provided by powered cross-coupled inverters to retain information. Power-gated blocks can deal with this loss of state by writing important state to a memory outside the block before gating the supply. This information can then be restored when the block is powered on.

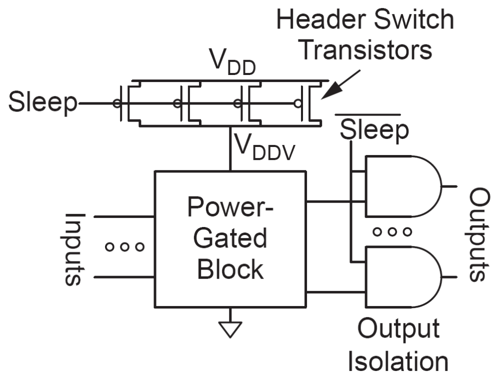


Figure 3.8: Header transistors used as power gating switches for a digital block (reprinted from [24]).

Alternatively, key state elements can be replaced with special retention flip-flops that can retain state in a low-power mode from a separate supply during power gating. Otherwise, state can simply be lost upon power gating. These options either increase the area overhead of power gating or the transition time into and out of a gated state. Finally, waking blocks from their gated state can impose dramatic changes in current loading on the supply, leading to voltage droops. If many systems are to be re-powered, their start-up must be staggered so the supply is not overly impacted.

DVFS and power gating are complementary techniques that can be applied under different circumstances to save energy. Indeed, some of the on-die voltage regulation techniques that allow for fine-grained DVFS can be easily modified to support power gating as well. However, the implementation challenges of power gating are generally considered more manageable than those of DVFS. Notably, power gating avoids the complexities of adaptive clocking and synchronization required for DVFS implementations. Accordingly, power gating is more widely implemented on commercial SoCs than DVFS, and it is generally implemented at a finer spatial granularity. It is important to consider whether the additional design complexity of DVFS is merited by improvements in energy savings compared to simpler power gating.

3.4.2 Variation and Noise Tolerance

Splitting a large design into multiple independent voltage areas increases the resilience of the design to noise and variation. This resilience can translate directly into smaller timing or voltage margins, which can improve performance or increase energy efficiency. The total variation across any clock domain will be less if each domain is smaller, so margins can be reduced to account for only this smaller amount of systematic on-die variation. This effect can be exploited further if the speed of each distinct voltage area is measured; each area can then be assigned a slightly different voltage to mitigate the variation between them, as was demonstrated in [25]. Alternatively, different voltage areas can be assigned different voltage settings and workloads based on their relative speed and power consumption, exploiting the variability instead of adapting to it. Finally, if the fine-grained DVFS system uses some form of adaptive clocking to generate clocks based on the varying local voltage (as in the aggressive reference system), these adaptive clocks also respond to process and temperature variations, as well as noise on the supply, improving the resilience of the system and reducing the required margins.

Increasing the number of independent voltage areas does not necessarily improve resilience to power supply noise. The most harmful form of power supply noise is di/dt noise, which is generally one or more voltage droops caused by sudden increases in current demand from the load on the supply. Having more independent voltage areas may decrease the likelihood that many parts of the system will suddenly turn on, reducing the probability of a current spike. On the other hand, voltage areas may switch between modes frequently in a fine-grained DVFS scheme, and these mode switches can place additional demands on the power supply network. Depending on how the independent voltages are generated, the power supply network may be more or less resilient to these fast changes, leaving the net effect of fine-grained DVFS uncertain. Existing approaches to mitigating dramatic current swings, such as taking care to wake large digital blocks from sleep one subsection at a time, may need to be rethought in a system with fine-grained DVFS.

3.4.3 Design Complexity

All of the tradeoffs above generally demonstrate that an increase in the spatial and temporal granularity of DVFS is associated with an increase in design complexity. This manifests itself as a design and verification challenge that ultimately requires more designer effort and expertise than a simpler system. In the research space, this can impact chip functionality (or even the ability to get a chip out the door); in the commercial space, it results in increased engineering costs and time to market. It is difficult to quantify this increase in design complexity, but it is nonetheless important to recognize this drawback of fine-grained DVFS. In the cross-cutting examples presented here, the aggressive references system would undoubtedly be a more complicated design to implement because of its many independent voltage domains with independent clocks and integrated regulators.

3.5 State of the Art

Many modern SoCs implement some form of dynamic voltage and frequency scaling. However, the design challenges of fine-grained DVFS have resulted in slow adoption of many of the techniques described in the previous section. A typical modern microprocessor SoC resembles the conservative reference design. The die is divided into a handful of independent clock domains, each of which is clocked via a PLL or DLL. Communication between each domain occurs through standard, high-latency brute-force synchronizers. Each voltage domain that can dynamically adjust its voltage is supplied by its own off-chip voltage regulator. These off-chip regulators are generally very efficient, but switch between modes relatively slowly (on the order of microseconds or even longer). The off-chip regulators, along with other on-board power management circuitry, may be governed by a microcontroller integrated with the main processor system, or the processor may communicate with a microcontroller elsewhere on the board that serves as a power management unit (PMU). This high-latency control loop, along with the time required to re-lock the clock generators for a domain undergoing DVFS, further limits the speed at which DVFS can be applied. In many SoCs, more areas of the chip are subject to power gating than to full-fledged DVFS,

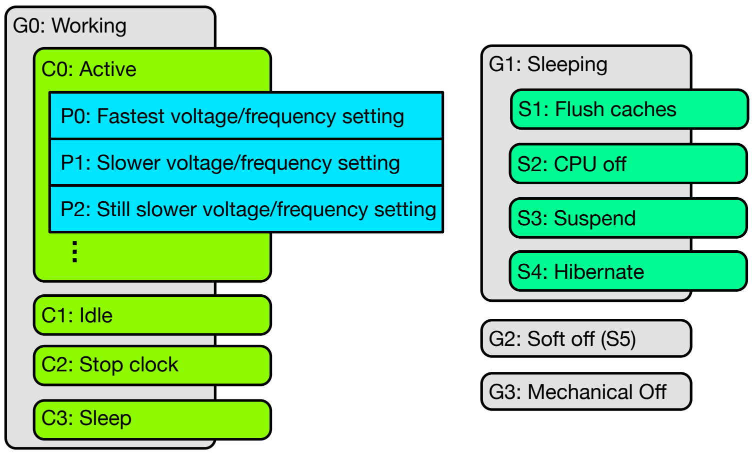


Figure 3.9: P-states and low-power sleep states in a typical microprocessor.

as the number of bumps required to provide high-quality independent supply rails limits the number of domains that can be supplied independently.

Switching between different voltage and frequency operating modes is controlled primarily by the operating system. A set of standards known as the Advanced Configuration and Power Interface (ACPI) govern the communication of power management information between hardware devices and the software running on them. Of most relevance to systems that implement dynamic frequency and voltage scaling, ACPI defines a series of implementation-dependent P-states that determine the power-performance setting of the system. Intel processors implement P-states as controlling the voltage and frequency settings of the system [26], with P0 representing the highest allowable voltage and frequency, P1 representing the next highest setting, and lower P-states representing additional voltage-frequency pairs (see Figure 3.9). Different parts of the system can be set to different P-states if they can operate at independent voltage and frequency settings. While P-states are typically set by the operating system based on its understanding of the need for performance or energy efficiency, the hardware PMU may have some freedom to adjust on-chip voltage and frequency independently of the software P-state setting. For example, a sudden increase in chip temperature might cause the PMU to scale the voltage of the chip, even if the operating system did not request such a change. In aggregate, however, on-chip power management tends to be limited to very slow timescales of milliseconds or longer.

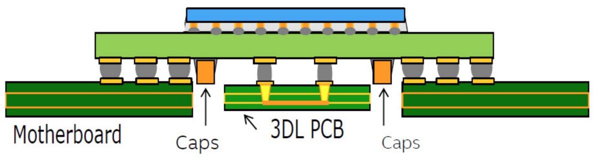


Figure 3.10: Inductors integrated into the PCB in Intel's Broadwell line [27].

3.5.1 Fast Integrated Regulators in Industry

Some recent microprocessors have improved upon industry-standard off-chip voltage regulation to enable more aggressive DVFS schemes by integrating voltage regulators more closely with the microprocessor SoC. While these chips do not take full advantage of fine-grained DVFS, they demonstrate that different approaches are feasible in industrial designs.

IBM has integrated voltage regulators into its POWER8 processors so that each core can operate at an independent frequency and voltage [18]. Each core is supplied by its own LDO regulators, allowing for fast per-core DVFS. However, as discussed in Section 3.3.1, the poor conversion efficiency of LDOs means that this technique is only useful for reducing instantaneous power, not improving overall energy efficiency. This is useful for IBM's server-class chips, which are typically power-limited, but does not realize the potential energy savings of fine-grained DVFS.

Intel has pursued a different approach with its latest Haswell and Broadwell processors. Haswell processors feature "fully integrated voltage regulators," or FIVRs, which are buck converters with air-core inductors integrated onto the package [28]. These efficient regulators can achieve 90% efficiency across a wide range of voltages and allow fast transitions between different voltage modes [28]. Each Haswell core can therefore operate at an independent P-state [29]. Broadwell processors revise the FIVR concept by moving the inductors onto the PCB, which must be specially designed to accommodate the Broadwell package (see Figure 3.10). These techniques realize much of the benefits of fine-grained DVFS, but incur considerable cost due to the complex package and board design required to realized the inductor-based regulation. Furthermore, it appears that processor voltage and frequency settings are still controlled primarily through P-states set by the operating system. Even

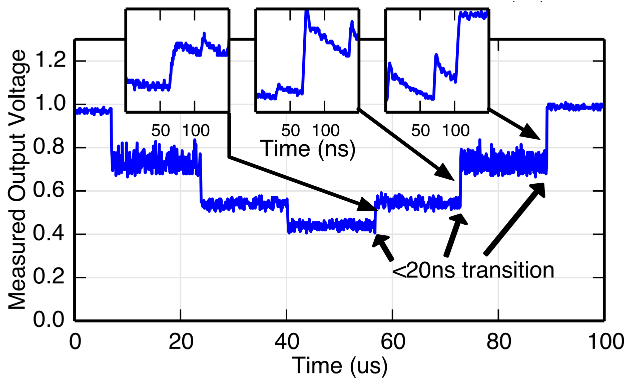


Figure 3.11: Silicon measurements of fast switching for DVFS implemented in the Raven project [4].

though the hardware supports fast transitions, the limited ability of the operating system to respond quickly to changes in operating conditions limits the energy savings of this approach.

3.5.2 The Raven Project

The Raven project implemented many of the design decisions of the aggressive reference system to enable fine-grained DVFS. As noted in Section 3.3.1, switched capacitor regulators are relatively easy to integrate on-die, but tend towards relatively low conversion efficiencies and power densities. The primary source of inefficiency in switched-capacitor regulators is due to charge sharing losses from the traditional approach of interleaving multiple converters to mitigate ripple on the supplied voltage [30]. Instead of interleaving, the Raven architecture switches all of its regulators simultaneously, avoiding such losses. The large ripple is tolerated by the system with an adaptive clock that adjusts the frequency of the chip on a per-cycle basis, ensuring high system efficiency. Resilient custom 8T SRAMs adapt their internal timing paths to the rippling voltage and employ assist techniques to operate at very low voltages.

Raven silicon, demonstrated in a 28nm FD-SOI process, achieved 26.2 GFlops/W operating at 550mV [4]. More importantly for this study, the full integration of the voltage regulators demonstrated very fast switching of 20ns or less between distinct voltage modes (see Figure 3.11) while maintaining a system energy efficiency above 80%. These results demonstrate the feasibility of fine-grained DVFS realized with efficient, fully integrated switched capacitor regulators.

Chapter 4

Options for Fine-Grained DVFS

The capabilities of integrated voltage regulators enable a wide array of possible schemes for fine-grained dynamic voltage and frequency scaling. In this chapter, we discuss some possibilities for implementing fine-grained DVFS in a processor system. We distinguish between two primary types of granularity. Fine-grained DVFS in time refers to mode switching that occurs quickly, ideally on a nanosecond timescale. Voltage scaling in time can be controlled either by software or by hardware and can be adjusted dynamically. Fine-grained DVFS in space refers to the implementation of many different independent voltage areas on a chip, each no more than a few square millimeters. The availability of DVFS in space is fixed in hardware at design time.

4.1 Fine-Grained DVFS in Time

There are numerous possible implementations of fine-grained DVFS in time. We consider both the nature of the control mechanism for setting voltage levels and the particular feedback to which the controller is responsive.

4.1.1 Mechanisms for DVFS

The control of voltage modes can be accomplished at many levels of the system hierarchy. As noted in Section 3.5, modern systems typically control voltage levels via the operating system. The OS determines a setting for each thread under its control, and communicates this setting to a power controller during a context switch. This control mechanism has limited ability to take advantage of fine-grained scaling, as thread switching occurs infrequently (on the order of milliseconds). Different control mechanisms both in software and hardware are better suited to fine-grained DVFS schemes.

Software control can be effected at a finer granularity than the operating-system level. The authors of [31] proposed the use of static code analysis to analyze programs at compile time and insert special instructions that indicate when the system voltage should change during the execution of the thread. This method could allow finer granularity than OS control, but it is an open-loop approach and cannot respond dynamically to the conditions of actual program execution. The authors of [32] instead call for dynamic recompilation of program segments based on the measured operating conditions of an initial execution pass. This dynamic recompilation technique would allow for closed-loop adaptive voltage scaling based on operating conditions. However, such an approach imposes overhead when code is profiled and recompiled, and it still cannot achieve scaling granularity in the microsecond or nanosecond range.

Other scaling mechanisms instead propose feedback control at the hardware level, typically by analyzing some set of microarchitectural counters to determine the appropriate operating voltage. One family of proposals averages the values of relevant counters over time and samples this average at some interval to predict the appropriate DVFS setting for the next interval [33] [34] [35] [36]. Alternatively, a particular microarchitectural event such as a cache miss could trigger a change in DVFS setting, with the setting remaining fixed between such events [37] [38]. These microarchitectural techniques have the greatest potential for accurate fine-grained DVFS algorithms because they respond directly to microarchitectural conditions and make no assumptions about the nature of the software being executed on the system.

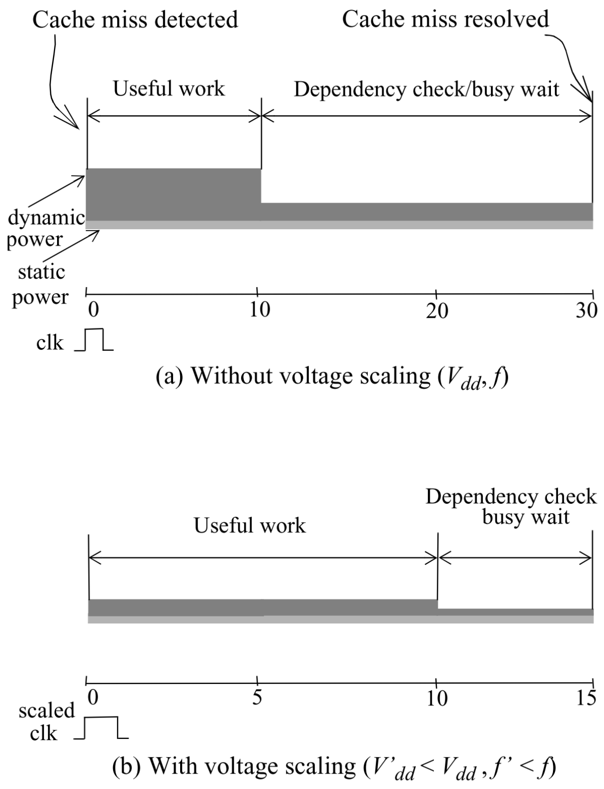
4.1.2 Algorithms for DVFS

Whether hardware or software control is employed, DVFS algorithms must respond to some particular system state. The algorithm must use the current state of the system to attempt to predict the lowest feasible voltage at which the system can safely operate without significantly impacting system performance. These algorithms face a fundamental tension between maintaining performance and saving energy. Conservative algorithms seek to ensure that performance is not unduly impacted by DVFS, but they may miss opportunities to reduce voltage and so incur energy overhead. Aggressive algorithms may improve energy savings at a performance cost. Systems that incur substantial energy or performance penalties when transitioning between DVFS modes have an additional incentive to switch voltages only when necessary.

The system state used to make decisions about DVFS settings can be predicted, observed, or some combination of the two. Common approaches track either a rolling average of instructions per cycle (IPC) [39] or cache miss rates [40] to attempt to determine whether a running program is in a CPU-bound or memory-bound phase. Low IPCs or high miss rates generally correspond to memory-bound program phases, during which the CPU is typically idle or underutilized. A DVFS algorithm could use these stimuli to scale down the voltage during these memory-bound program regions with minimal impact on overall performance.

A more general approach does not assume any particular system structure, instead generalizing CPUs and other systems to a group of functional units that communicate by queues. The authors of [34] propose monitoring the average state of these queues as a DVFS feedback mechanism. If the input queues to a block are mostly full and the output queues are mostly empty, the voltage is raised to increase the processing rate; if the input queues are empty and there is little work to do, the voltage is decreased in response. This approach easily extends to manycore architectures if each core can scale its voltage independently [16]. Such a feedback mechanism might require a global controller to avoid deadlock or local minima.

Relying on sampling or rolling averages can adapt voltage to reasonably fine timescales; as noted in [39], programs exhibit varying memory-boundedness under a wide range of time intervals. However, for extremely fine-grained DVFS, voltage scaling ought to be triggered



Cache miss detected
Cache miss resolved
Useful work
Dependency check/busy wait
dynamic power
static power
0
10
20
30
clk
(a) Without voltage scaling (V_{dd}, f)
Useful work
Dependency check busy wait
0
5
10
15
scaled clk
(b) With voltage scaling ($V'_{dd} < V_{dd}$, $f' < f$)

Figure 4.1: The savings of the fine-grained DVFS technique proposed in [37].

in response to a particular microarchitectural event. The authors of [37] proposed scaling the voltage of a core in response to individual cache misses, which can take tens or hundreds of cycles to resolve (see Figure 4.1). Alternatively, individual functional units that are used and idled on a fine timescale could be set to a higher operating voltage when in use, and to a lower voltage when idle. Techniques that scale voltage as an immediate hardware response to a microarchitectural event will be best equipped to implement DVFS in very short timescales.

4.2 Fine-Grained DVFS in Space

The choice of algorithm depends to a large extent on the manner in which the chip is partitioned into independently scaled voltage domains. Integrated voltage regulators free system designers from the bump limitations of traditional independent supplies, allowing for an effectively limitless number of voltage domains on a single die. As noted in Section 3.1, the

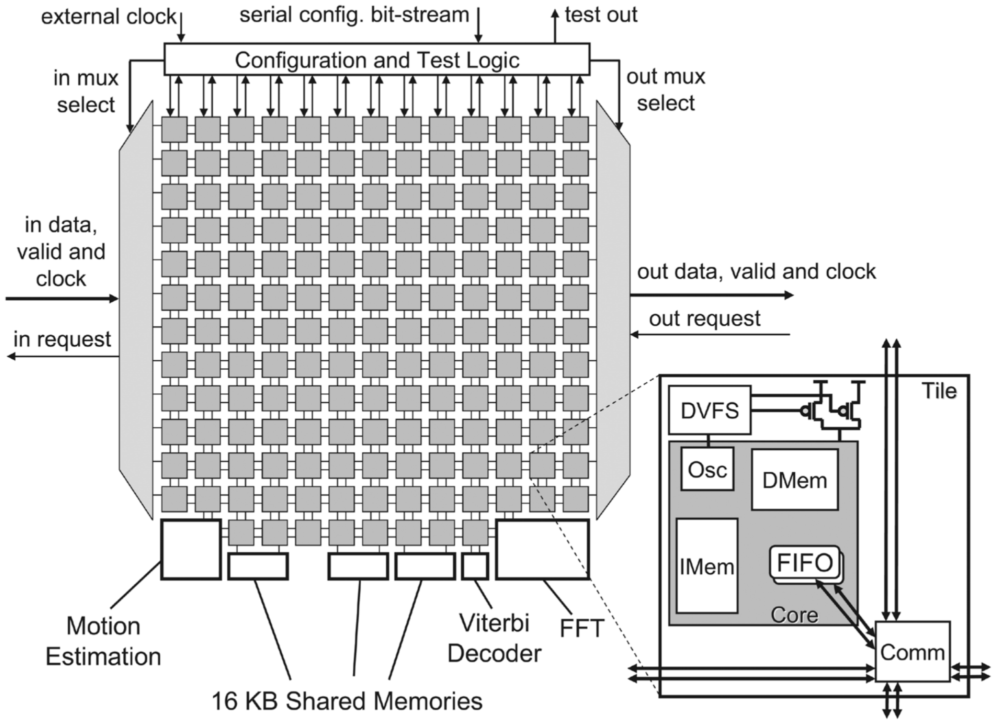


Figure 4.2: The manycore system presented in [16]. The voltage of each core can be set independently.

main disadvantage of increasing the number of voltage domains is that additional latency is introduced at the domain interfaces. Because of this limitation, logic cannot easily be partitioned arbitrarily into different domains, but instead must be split at ready-valid interfaces. This limits the vast number of possibilities for voltage area partitioning.

4.2.1 Manycore Systems

Manycore systems or designs with many independent processing elements lend themselves easily to fine-grained partitioning, as each core can be readily assigned to its own voltage area (see Figure 4.2). Cores can then independently scale voltage based on local workload estimation or a global controller. One example uses a dual-rail switching approach, in which each core adapts its voltage to locally measured workload [16]. If the manycore system is also used as a spatial networking fabric, care must be taken that cores that are slowed due to low workload do not inadvertently slow critical traffic on the network.

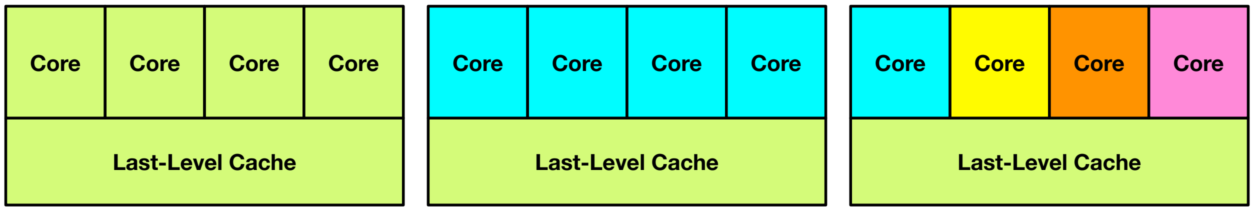


Figure 4.3: Different schemes for voltage area partitioning. Each color represents an independent voltage area.

4.2.2 Memory Interfaces

Memory subsystems represent another straightforward target for independent voltage areas. In a modern SoC with a shared last-level cache, the cache can act on a separate voltage domain from the cores, which can either be supplied with a single voltage or made independent as well (see Figure 4.3). This allows cores to be scaled while the memory system is active with a request. Alternatively, if all cores are busy with locally cached work and there is little traffic in the memory system, the shared cache could be scaled instead, although care must be taken to avoid slowing coherence traffic that might be critical for locally executing programs. Deeper memory hierarchies could be matched with further voltage domain partitioning, with separate voltage areas for each level of the hierarchy.

4.2.3 Out-of-Order Processors

While fine-grained DVFS schemes envision core boundaries as appropriate voltage area partitions, there have also been proposals to include multiple voltage areas within out-of-order cores. For example, the authors of [41] proposed splitting an out-of-order core into four independent voltage areas, separating the frontend, integer functional units, floating-point functional units, and memory system (see Figure 4.4. The authors of [42] chose a similar partitioning, except that the frontend is split into two domains, fetch and decode. The key decision in these designs is to separate the functional units from the processor frontend; since the out-of-order processor already decouples the fetching of instructions from their issue to the functional units, additional latency at this interface may be tolerable. As noted in [42], the main drawback of this approach occurs during branch mispredicts, in which additional

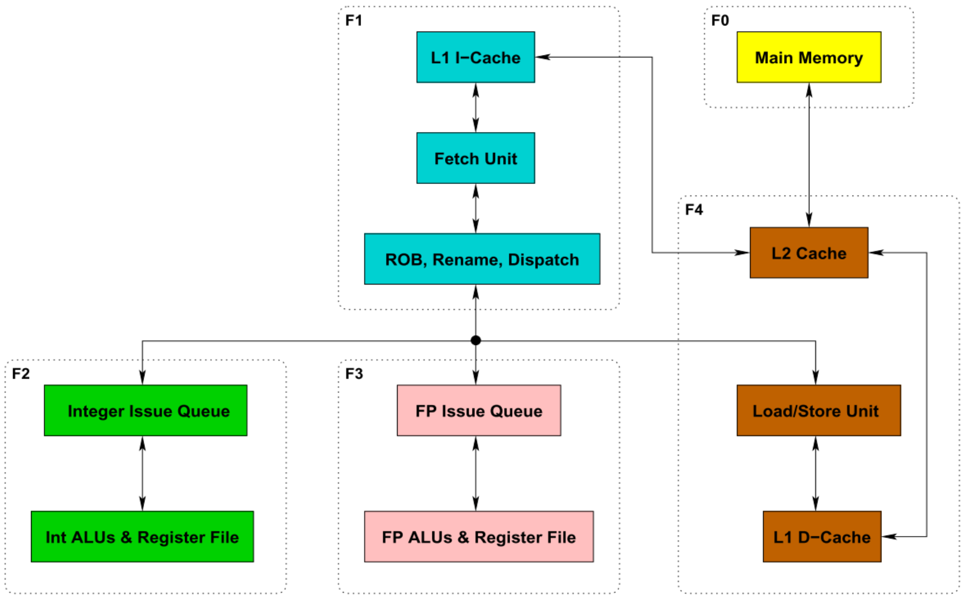


Figure 4.4: An out-of-order core partitioned into four voltage domains [41].

cycles are required after the pipeline flush before full throughput can again be achieved. However, the energy benefits of such extremely fine-grained partitioning may result in a worthwhile tradeoff against this performance penalty.

Chapter 5

Fine-Grained DVFS Limit Study

The analysis in previous chapters suggests a complex trade-off between the costs and benefits of fine-grained DVFS. In this chapter, data is presented that shows the best achievable benefits of a fine-grained DVFS approach. First, we derive a simple model to estimate the potential energy savings of fine-grained DVFS. Next, we present the results of a processor simulation to show the amount of idle time during program execution that could be leveraged to save energy using fine-grained DVFS in space. Finally, we simulate the impact on execution time of the additional latency imposed by fine-grained DVFS in time. Coupling the energy model with the simulated results finds that fine-grained DVFS provides typical energy savings of 18%, while the performance penalty from additional interface latency is limited to an average of 2.9%.

The data from Sections 5.2 and 5.3 were collected from simulation of the RISC-V Rocket processor [43]. The test inputs of selected benchmarks from the SPEC2006 integer suite were executed on the Rocket processor mapped onto a Xilinx Zynq FPGA. The processor was instrumented with counters to measure key microarchitectural events. This test infrastructure simulates a cycle-accurate model of processor RTL, and so reproduces the performance of real processor systems with high fidelity. A summary of key parameters of the simulated processor is presented in Table 5.1.

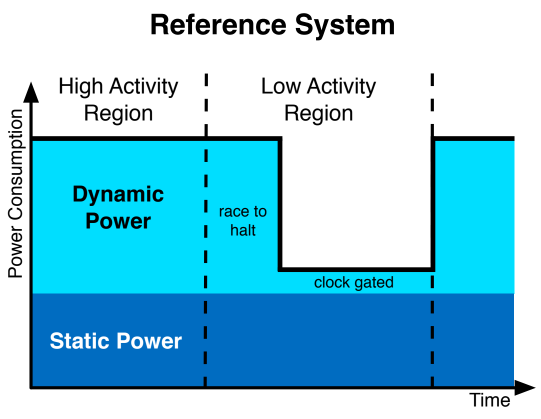
Table 5.1: Parameters of the simulated processor system.

Architecture	RISC-V G [44]
Pipeline	6-stage single-issue in-order
L1 Data Cache	16KB 4-way
L1 Instruction Cache	16KB 4-way
L2 Cache	2 banks, 128KB per bank, 8-way
Branch Predictor	64-entry BTB, 128-entry two-level predictor

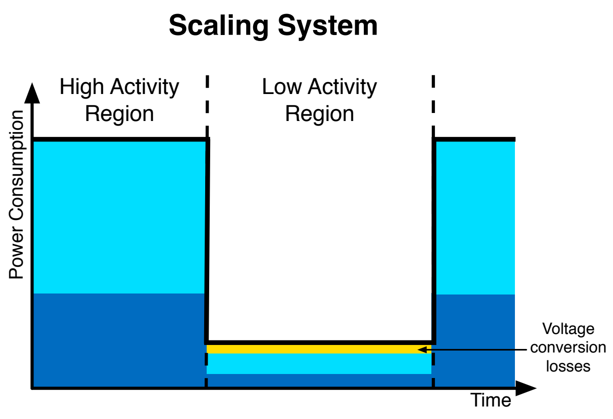
5.1 Energy Model

To estimate the potential energy savings achievable by fine-grained DVFS, we use a simple energy model based on the analysis in Chapter 2. We consider two systems for comparison: one that can scale between different voltage settings instantaneously with no overhead (the "scaling system"), and one that can only operate at a fixed voltage (the "reference system"). The reference system may implement coarse-grained DVFS, power gating, or other power-saving techniques, but we assume that these techniques are too slow be applied at the timescales of interest. For simplicity, we assume that the scaling system has only two operating modes, one at nominal V_{DD} and one at a reduced V_{DDL}, although the model can easily be extended to accommodate additional operating points. We assume that each system consists of a single voltage domain, neglecting any additional savings that could be realized through spatially fine-grained DVFS.

In order for the scaling system to save energy, its workload must be amenable to DVFS. For simplicity, we segment the workload into high-activity and low-activity segments, and without loss of generality we set the activity during high-activity segments to $\alpha = 1$ (though actual logic activity is lower even during high-activity periods). The reference system operates at the nominal voltage through the entire program, while the scaling system can reduce its voltage during low-activity segments to save energy without impacting overall runtime. We assume that the time scale of these changes in activity is too small for the reference system to use power gating or other slow power saving techniques. (For example, the low-activity segments could be caused by cache misses as discussed in Section 4.1.2.) The reference system can clock gate during idle periods of low-activity phases to reduce its



(a)



(b)

Figure 5.1: Energy consumption during high and low activity program segments for the reference system (a) and the scaling system (b) as modeled in this section.

dynamic power, but it will still consume static power (see Figure 5.1).

The relative energy consumption of each system can be found from the parameters of the system as discussed in Section 2.2. Equation 2.6 approximated the relationship between the the operating frequency f and the voltage of the scaling system as a simple function:

$$f = k(V_{DD} - V_t) \tag{5.1}$$

where V_t is the threshold voltage. Equation 2.1 then gives dynamic energy consumed by the system as

$$P_{dyn} = \alpha C V_{DD}^2 f = K\alpha V_{DD}^2 (V_{DD} - V_t), \tag{5.2}$$

where C is the capacitive load of the system, α is the activity factor, and K is a constant. As noted in Chapter 2, there is no simple analytical relationship between static power and V_{DD}. We assume that static power at V_{DD} and V_{DDL} are given by P_{stat} and P_{statL} respectively; these values can be determined via process characterization, simulation, or a leakage model fit to available data.

A given workload will contain some fraction t_{low} of its execution time in low-activity segments and the remaining time t_{hi} in high-activity segments. We will assume for simplicity that the ratio of the activity during high-activity segments to that of low-activity segments is the same as the ratio of clock frequencies at V_{DD} and V_{DDL}. This assumption implies that the scaling system always operates at full activity, and is never idle; its slower execution of low-activity segments just matches the lesser amount of work available during those segments. In reality, a wide variety of different activity modes would exist in most workloads. Accounting for such variation would complicate this analysis but would not dramatically affect its conclusions.

The reference system operates entirely at V_{DD}. The system energy consumption during high-activity segments is given by

$$\begin{aligned} E_{ref,H} &= t_{hi}(P_{dyn} + P_{stat}) && (5.3) \\ &= t_{hi}(KV_{DD}^2(V_{DD} - V_t) + P_{stat}) && (5.4) \end{aligned}$$

Note that $\alpha = 1$ because the system is at full activity. During low-activity segments, which have a lower activity factor $\alpha_L = \frac{V_{DDL}-V_t}{V_{DD}-V_t}$, the leakage energy of the reference system remains the same, but its dynamic power now consists of two components. The first component is the dynamic power P_{dyn} spent at full activity during t_{low}; this power is only spent over part of the t_{low} execution time (specifically, over a portion α_L of t_{low}). The second component is the dynamic power P_{cg} spent under clock gating. Ideally, no switching would occur during clock gating, and this energy would be zero. Real systems have some clock gating efficiency η_{cg} that represents the proportion of activity that is suppressed during clock gating, and therefore have some remaining switching activity $\alpha_{cg} = 1 - \eta_{cg}$. The total system energy consumption of the reference system during high-activity segments is therefore:

$$\begin{aligned} E_{ref,L} &= t_{lo}(\alpha_L P_{dyn} + (1 - \alpha_L)P_{cg} + P_{stat}) && (5.5) \\ &= t_{lo}(\alpha_L KV_{DD}^2(V_{DD} - V_t) + (1 - \alpha_L)(1 - \eta_{cg})KV_{DD}^2(V_{DD} - V_t) + P_{stat}) && (5.6) \end{aligned}$$

The scaling system operates at V_{DD} during t_{hi} and V_{DDL} during t_{lo}. The system energy consumption during high-activity segments is the same as that of the reference system:

$$\begin{aligned} E_{scale,H} &= t_{hi}(P_{dyn} + P_{stat}) && (5.7) \\ &= t_{hi}(KV_{DD}^2(V_{DD} - V_t) + P_{stat}) && (5.8) \end{aligned}$$

The system energy consumption during low-activity segments is lower in both static and dynamic energy because the supply voltage is reduced. While the system operates at full activity over the entire t_{lo} because it is completing the same amount of work at a lower frequency, it does so at a lower voltage, a net reduction in energy. Additionally, no dynamic energy is expended on imperfect clock gating. However, power lost due to the voltage regulators needed to supply the lower voltage must also be accounted for, because the regulators required to generate the lower voltage are not perfectly efficient. The total system energy consumption for the scaling system when supplied by a regulator with efficiency η is given by:

$$\begin{aligned} E_{scale,L} &= \tfrac{1}{\eta}(t_{lo}(P_{dyn} + P_{stat})) && (5.9) \\ &= \tfrac{1}{\eta}(t_{lo}(KV_{DDL}^2(V_{DDL} - V_t) + P_{statL})) && (5.10) \end{aligned}$$

Note that we pessimistically assume a fixed number of cycles for the computation, even though voltage scaling may decrease the number of cycles required to complete the same amount of work as discussed in Section 2.3.

The net energy savings achieved by the scaling system is by the ratio of the energy consumed by each system:

$$\text{Energy Savings} = 1 - \frac{E_{scale,H} + E_{scale,L}}{E_{ref,H} + E_{ref,L}}$$

To evaluate the potential savings of a real system, we use the parameters shown in Table 5.2:

Table 5.2: Parameters used for energy model.

V_{DD}	Nominal supply voltage	1V
V_{DDL}	Scaled supply voltage	0.5V
V_t	Process threshold voltage	450mV
$\frac{P_{statL}}{P_{stat}}$	Leakage ratio	0.2
$\frac{t_{hi}}{t_{lo}}$	Activity ratio	1
η_{cg}	Clock gating efficiency	0.9
$\eta = 0.85$	Voltage regulator efficiency	0.85

K is chosen such that the ratio of dynamic power to leakage power at V_{DD} is 2 to 1. These parameter values are realistic for modern processes and provide a general estimate of projected energy savings for a modern system.

Using these parameters, we find an overall energy savings of **24.6%** for the scaling system. This is a promising overall savings, even though some of the overheads of implementing DVFS are not accounted for in this simple energy model. This result is heavily dependent on the parameters chosen to complete the analysis. For example, a lower power conversion efficiency η reduces the savings of the scaling system, while a steeper scaling of leakage with V_{DD} increases the energy savings. The result is particularly sensitive to the workload characteristics encapsulated in the ratio $\frac{t_{hi}}{t_{lo}} = 1$. If more of the workload has low activity and is amenable to scaling, then the amount energy savings that can be realized increases dramatically. This motivates the study in the next section.

5.2 Idleness Survey

The utility of fine-grained DVFS in processors is predicated on the existence of periods of inactivity during normal program execution. We conducted an "idleness survey" of several SPEC2006 integer benchmarks. By instrumenting a Rocket processor with several counters, the aggregate idleness available during these representative approaches can be estimated. These results can be combined with the energy model presented in the previous section to more accurately estimate the potential energy savings of fine-grained DVFS.

The Rocket processor already includes built-in counters to measure the number of elapsed cycles (CYCLE) and the number of instructions retired (INSTRET) because these basic counters are mandated by the RISC-V specification [44]. We augmented the processor RTL with the following additional counters:

- L1DHIT, the number of L1 data cache hits
- L1DMISS, the number of L1 data cache misses
- L1IHIT, the number of L1 instruction cache hits
- L1IMISS, the number of L1 instruction cache misses
- L2MISS, the number of last-level cache misses

This augmented RTL is mapped to the FPGA so that long-running programs such as the

SPEC benchmarks can be run on the instrumented design. All of these counters are read out at the end of program execution to provide a performance summary of the execution.

An example of the counter values measured during program execution is shown in Table 5.3, which shows the results for the `astar` SPECINT2006 benchmark. These measurements can provide basic architectural information about the program execution. For example, dividing INSTRET by CYCLE determines the instructions per cycle (IPC) of the trace, which in this case is 0.37. Taking the ratio of L1DMISS to the total number of L1 data cache accesses gives the L1 data cache miss rate, which is 20.3% in this example. These statistics can be used to determine various estimates of the idle time during program execution.

Table 5.3: Counter values reported for the `astar` benchmark.

CYCLE	INSTRET	L1DHIT	L1DMISS	L1IHIT	L1IMISS	L2MISS
$6.33 \cdot 10^{10}$	$2.36 \cdot 10^{10}$	$9.15 \cdot 10^{9}$	$2.34 \cdot 10^{9}$	$2.86 \cdot 10^{10}$	$6.57 \cdot 10^{8}$	$4.40 \cdot 10^{8}$

The simplest estimate of system idleness during program execution can be found simply by comparing CYCLE and INSTRET. Because the processor is in order and can retire at most one instruction per cycle, any cycle in which an instruction is not retired must result in a pipeline stall. These stalls can occur for many reasons, such as cache misses, functional unit contention, or exceptions, and may last for a single cycle or for many. In the ideal case, however, any stall represents idleness that could be exploited by DVFS. We name this idleness metric IDLE0, defined as:

$$\text{IDLE0} = \tfrac{\text{CYCLE}-\text{INSTRET}}{\text{CYCLE}} \tag{5.11}$$

The average of IDLE0 for the data in Table 5.3 is 63%, suggesting that there are frequent opportunities for energy savings from fine-grained DVFS from this trace.

One important way in which the FPGA simulation environment differs from silicon implementations of the same RTL is the relative speed of the FPGA-mapped processor and the access time of off-chip DRAM. Because the FPGA logic is clocked much more slowly than equivalent silicon, the number of processor cycles required to resolve a DRAM request is much fewer than it would be in a real system. This effect mitigates the processor idleness incurred by an L2 miss; a real system would have to wait for more cycles for the memory system to respond. By estimating the magnitude of this effect we can attempt to correct for it to derive a more accurate estimate of system idleness.

We define two estimated parameters, LAT and LAT_FPGA. LAT is defined as the average number of CPU cycles required to resolve an L2 miss in a real system. LAT_FPGA is the number of CPU cycles required to resolve an L2 miss on the FPGA system. Since we have counted the number of L2 misses in the system, we can derive a cycle time estimate CYCLE_EST for the real system and use this to estimate the what the idle time would be if L2 misses required as many cycles to resolve as they would in a real system. We call this idleness metric IDLE1:

$$\mathrm{IDLE1} = \frac{\mathrm{CYCLE\_EST} - \mathrm{INSTRET}}{\mathrm{CYCLE\_EST}} = \frac{(\mathrm{CYCLE} + \mathrm{L2MISS} \cdot (\mathrm{LAT} - \mathrm{LAT\_FPGA})) - \mathrm{INSTRET}}{\mathrm{CYCLE} + \mathrm{L2MISS} \cdot (\mathrm{LAT} - \mathrm{LAT\_FPGA})} \tag{5.12}$$

If we estimate LAT=100 and LAT_FPGA=25, then IDLE1 for the data in Table 5.3 is 75%. Note that the idleness estimate increases by 12% when the latency-hiding nature of FPGA execution is accounted for.

A final estimate of available idleness can instead be based solely on L2 cache misses. This is representative of a DVFS scheme that scales down the voltage on an L2 cache miss, which may be a more realistic model. (It may not be possible for a DVFS algorithm to scale in response to stalls from other sources, both because those stalls may be of short duration and because they may be more difficult to detect or predict.) To determine this IDLE2 metric, we simply compare the number of L2 misses to the total execution time, using the CYCLE_EST metric described previously to account for the longer L2 miss latency in real systems:

$$\mathrm{IDLE2} = \frac{\mathrm{INSTRET} \cdot \mathrm{LAT}}{\mathrm{CYCLE\_EST}} = \frac{\mathrm{INSTRET} \cdot \mathrm{LAT}}{\mathrm{CYCLE} + \mathrm{L2MISS} \cdot (\mathrm{LAT} - \mathrm{LAT\_FPGA})} \tag{5.13}$$

Using the same estimates for LAT and LAT_FPGA yields an average IDLE2 of 46% for the data from Table 5.3. More than 60% of the idleness from the IDLE1 metric can be attributed to L2 miss latency, suggesting that a DVFS scheme that scales on L2 cache misses could take advantage of a signification portion of the total idleness available to the system.

Figure 5.2 shows the three idleness metrics calculated from the traces of several benchmarks from the SPECINT2006 suite. The simulation results show wide variation in estimated idle time both among different benchmarks and between the different idleness metrics. Some

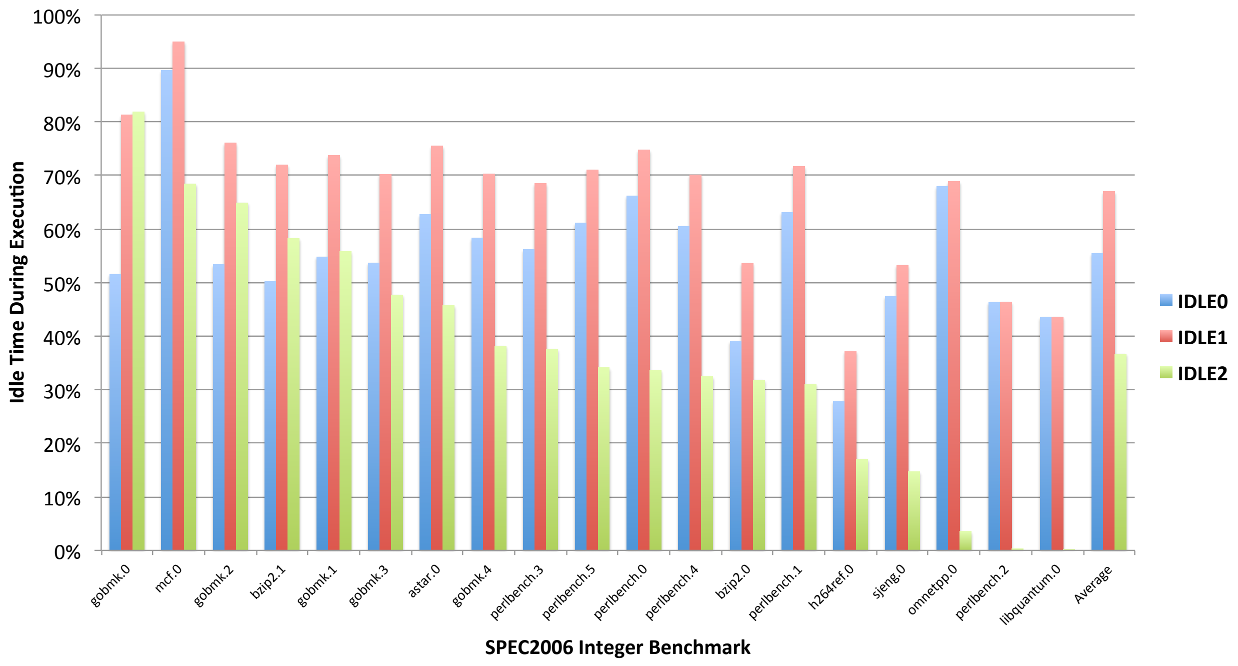


Figure 5.2: Idle time measured during benchmark execution on the simulated processor.

benchmarks, such as `gobmk`, `mcf`, and `bzip` spend a majority of execution time idle by any of the idleness metrics. In particular, `gobmk.0` measures 81% idle by the IDLE1 metric, but also reports a nearly identical result from the IDLE2 metric, suggesting that its idleness is almost entirely a result of L2 cache misses. In contrast, `mcf.0` measures nearly 95% idle by the IDLE1 metric, but only 68% idle by the IDLE2 metric, suggesting that a substantial portion of the idle time for this benchmark is not caused by L2 cache misses. Every benchmark measures IDLE1 as at least 30% of execution time, but several benchmarks, such as `perlbench.2` and `libquantum.0`, report very few L2 cache misses and negligible IDLE2 time. Averaging the reported results yields an IDLE1 time of 67% (a range of 38% to 95%) and an IDLE2 time of 37% (with a range from 0% to 82%), suggesting significant opportunity for fine-grained DVFS in these idle periods.

We can use the estimated idleness from this section in the energy model from the previous section by assuming that any idle time will be spent at the lower voltage and the remaining time will be spent at the higher voltage. We can therefore set t_{lo} and t_{hi} such that $\frac{t_{hi}}{t_{lo}}$ matches the derived idle proportion. Figure 5.3 shows the resulting energy savings for each benchmark using the IDLE2 metric. The model predicts significant energy savings for the benchmarks

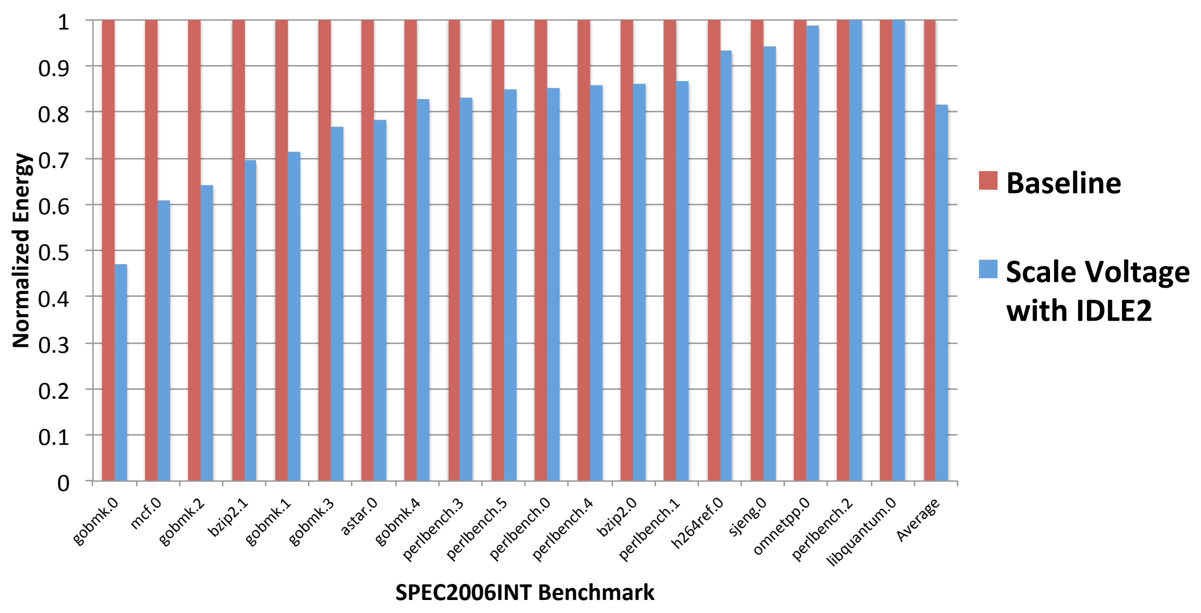


Figure 5.3: Energy savings predicted from the energy model for each benchmark.

with large IDLE2 time. On average, the benchmarks reduce their energy consumption by 18% according to the model, with results ranging from no savings to 53% savings depending on the specific benchmark.

5.3 Latency Impact

As noted in Section 3.1, one of the drawbacks of fine-grained DVFS in space is the additional latency associated with asynchronous interface crossings. This latency translates directly into a performance penalty by increasing the number of cycles required to finish a given amount of work. This performance penalty can in turn result in an energy penalty as the system must either operate for a longer time or increase the voltage to finish the work in the same amount of time. We use the FPGA infrastructure described in the previous section to estimate the penalties associated with this additional latency.

In a processor or multiprocessor system, independently scaling cores necessitate an asynchronous interface between each core and the shared higher-level memory system. In the Rocket processor, this interface separates the per-core L1 caches from the shared L2 cache.

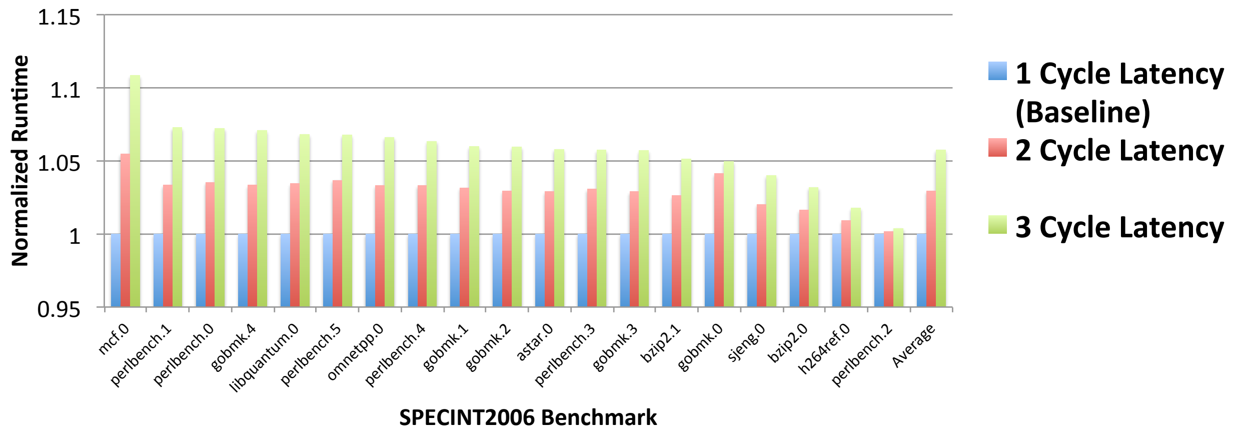


Figure 5.4: Simulated performance impact of added L1-L2 latency for several benchmarks.

Because queues are already present at this interface, and because L1 cache misses already incur several cycles of latency, this interface represents a sensible partition for DVFS. The synchronous queues present in the fully synchronous system already incur a minimum of one cycle of latency for data passing through them. From this baseline, the RTL of the Rocket processor was modified to increase this latency. One version of the RTL which we name 2C increased the latency to two cycles, representing a minimal brute-force asynchronous FIFO with two-stage synchronizers. This asynchronous FIFO design with two stages of latency is common in the literature. Another version of the RTL which we name 3C increased the latency at the interface to three cycles, representing a brute-force asynchronous FIFO with three-stage synchronizers. This more robust synchronizer would be more readily used in industrial designs. The same software benchmarks were then executed on each design mapped to the FPGA; total cycle counts capture the performance penalty incurred by the added latency at the L1-L2 interface.

Figure 5.4 shows the results of the experiment. The performance penalty incurred by the added latency varies by benchmark according to the number of L1 cache misses incurred during program execution. Some benchmarks, such as h264ref and perlbench.2, largely fit in the L1 cache, and so suffer very little penalty for increased latency. mcf incurs a large number of L1 cache misses, and so suffers a performance hit of 5.5% when run on 2C and of 10.8% when run on 3C. However, the average performance penalty across all of the benchmarks is relatively small; 2C incurs a 2.9% performance penalty on average (with a range from

0.2% to 5.5%), and 3C requires 5.7% more cycles to complete the benchmarks (with a range from 0.4% to 10.8%). These performance penalties are small compared the potential energy savings found in the previous section, demonstrating that fine-grained DVFS remains a viable approach to save energy even as this trade-off is considered.

Chapter 6

Conclusion

Fine-grained dynamic voltage and frequency scaling has the potential to dramatically increase energy efficiency in processor systems. The advent of fast, efficient, integrated switching regulators can realize fine-grained DVFS in both space and time. In this report, we have outlined the fundamentals and trade-offs of fine-grained DVFS approaches, as well as providing a brief overview of possible algorithms and techniques. By analyzing program traces executed on a simulated processor, we demonstrated the feasibility and potential energy savings of one DVFS approach.

This work could be expanded to further make the case for fine-grained DVFS in modern SoCs. In particular, the model of energy savings could be refined by extending the model to more than two voltage levels. A sensitivity analysis of key parameters in the model would give insight into important tradeoffs and design decisions that would maximize the energy savings of such an approach. Finally, the model and associated FPGA-based simulations could be applied to more complex hardware and software, including longer-running benchmarks, out-of-order processors, and multicore systems. Further exploration of design, algorithms, and silicon implementation will be required to design and implement the best schemes for fine-grained DVFS.

Acknowledgements

The author would like to thank the many students, faculty, and staff of the ASPIRE Lab and the Berkeley Wireless Research Center for their hard work and assistance. This work was funded by the National Science Foundation Graduate Research Fellowship, the NVIDIA Graduate Fellowship, and Intel ARO.

Bibliography

[1] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, Apr. 1992.

[2] S. Jain, S. Khare, S. Yada, V. Ambili, P. Salihundam, S. Ramani, S. Muthukumar, M. Srinivasan, A. Kumar, S. K. Gb, R. Ramanarayanan, V. Erraguntla, J. Howard, S. Vangal, S. Dighe, G. Ruhl, P. Aseron, H. Wilson, N. Borkar, V. De, and S. Borkar, "A 280mV-to-1.2V wide-operating-range IA-32 processor in 32nm CMOS," in *Proc. IEEE International Solid-State Circuits Conference*, vol. 55, 2012, pp. 66–67.

[3] T. Sakurai and A. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 2, pp. 584–594, 1990.

[4] B. Zimmer, Y. Lee, A. Puggelli, J. Kwak, R. Jevtic, B. Keller, S. Bailey, M. Blagojevic, P.-F. Chiu, H.-P. Le, P.-H. Chen, N. Sutardja, R. Avizienis, A. Waterman, B. Richards, P. Flatresse, E. Alon, K. Asanovic, and B. Nikolic, "A RISC-V vector processor with tightly-integrated switched-capacitor DC-DC converters in 28nm FD-SOI," in *Proc. IEEE Symposium on VLSI Circuits*, 2015.

[5] B. H. Calhoun, A. Wang, and A. Chandrakasan, "Modeling and sizing for minimum energy operation in subthreshold circuits," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 9, pp. 1778–1785, 2005.

[6] J. Myers, A. Savanth, D. Howard, R. Gaddh, P. Prabhat, and D. Flynn, "An 80nW retention 11.7pJ/cycle active subthreshold ARM Cortex-M0 subsystem in 65nm CMOS for WSN applications," in *Proc. IEEE International Solid-State Circuits Conference*, Feb 2015, pp. 1–3.

[7] C. Dike and E. Burton, “Miller and noise effects in a synchronizing flip-flop,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 6, pp. 849–855, Jun 1999.

[8] P. Li, J. Shin, G. Konstadinidis, F. Schumacher, V. Krishnaswamy, H. Cho, S. Dash, R. Masleid, C. Zheng, Y. Lin, P. Loewenstein, H. Park, V. Srinivasan, D. Huang, C. Hwang, W. Hsu, and C. McAllister, “A 20nm 32-Core 64MB L3 cache SPARC M7 processor,” in *Proc. IEEE International Solid-State Circuits Conference*, Feb 2015, pp. 1–3.

[9] R. Ginosar, “Fourteen ways to fool your synchronizer,” in *Proc. IEEE Symposium on Asynchronous Circuits and Systems*, 2003, pp. 89–96.

[10] P. Teehan, M. Greenstreet, and G. Lemieux, “A survey and taxonomy of GALS design styles,” in *Proc. IEEE Design & Test of Computers*, vol. 24, no. 5, 2007, pp. 418–428.

[11] A. Chakraborty and M. Greenstreet, “Efficient self-timed interfaces for crossing clock domains,” in *Proc. IEEE Symposium on Asynchronous Circuits and Systems*, 2003, pp. 78–88.

[12] B. Keller, M. Fojtik, and B. Khailany, “A pausible bisynchronous FIFO for GALS systems,” in *Proc. IEEE Symposium on Asynchronous Circuits and Systems*, 2015, pp. 1–8.

[13] C. R. Lefurgy, A. J. Drake, M. S. Floyd, M. S. Allen-Ware, B. Brock, J. A. Tierno, and J. B. Carter, “Active management of timing guardband to save energy in POWER7,” in *Proc. International Symposium on Microarchitecture*, 2011, pp. 1–11.

[14] M. Abdelfattah, M. Ghoneima, Y. Ismail, A. Lotfy, M. Abdelsalam, M. Abdelmoneum, N. Kurd, and G. Taylor, “A novel digital loop filter architecture for bang-bang ADPLL,” in *Proc. IEEE SOC Conference*, Sept 2012, pp. 45–50.

[15] B. Stackhouse, S. Bhimji, C. Bostak, D. Bradley, B. Cherkauer, J. Desai, E. Francom, M. Gowan, P. Gronowski, D. Krueger, C. Morganti, and S. Troyer, “A 65 nm 2-billion transistor quad-core Itanium processor,” *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 18–31, Jan 2009.

[16] D. N. Truong, W. H. Cheng, T. Mohsenin, Z. Yu, A. T. Jacobson, G. Landge, M. J. Meeuwsen, C. Watnik, A. T. Tran, Z. Xiao, E. W. Work, J. W. Webb, P. V. Mejia, and B. M. Baas, “A 167-processor computational platform in 65 nm CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, pp. 1130–1144, Apr. 2009.

[17] J. Warnock, B. Curran, J. Badar, G. Fredeman, D. Plass, Y. Chan, S. Carey, G. Salem, F. Schroeder, F. Malgioglio, G. Mayer, C. Berry, M. Wood, Y.-H. Chan, M. Mayo, J. Isakson, C. Nagarajan, T. Werner, L. Sigal, R. Nigaglioni, M. Cichanowski, J. Zitz, M. Ziegler, T. Bronson, G. Strevig, D. Dreps, R. Puri, D. Malone, D. Wendel, P.-K. Mak, and M. Blake, “22nm next-generation IBM System z microprocessor,” in *Proc. IEEE International Solid-State Circuits Conference*, Feb 2015, pp. 1–3.

[18] E. Fluhr, J. Friedrich, D. Dreps, V. Zyuban, G. Still, C. Gonzalez, A. Hall, D. Hogenmiller, F. Malgioglio, R. Nett, J. Paredes, J. Pille, D. Plass, R. Puri, P. Restle, D. Shan, K. Stawiasz, Z. Deniz, D. Wendel, and M. Ziegler, “POWER8: A 12-core server-class processor in 22nm SOI with 7.6Tb/s off-chip bandwidth,” in *Proc. IEEE International Solid-State Circuits Conference*, 2014, pp. 96–97.

[19] H.-P. Le, S. Sanders, and E. Alon, “Design techniques for fully integrated switched-capacitor DC-DC converters,” *IEEE Journal of Solid-State Circuits*, vol. 46, no. 9, pp. 2120–2131, Sept. 2011.

[20] M. Seeman, V. Ng, H.-P. Le, M. John, E. Alon, and S. Sanders, “A comparative analysis of switched-capacitor and inductor-based DC-DC conversion technologies,” in *Proc. IEEE Workshop on Control and Modeling for Power Electronics*, June 2010, pp. 1–7.

[21] W. H. Cheng and B. M. Baas, “Dynamic voltage and frequency scaling circuits with two supply voltages,” in *Proc. IEEE International Symposium on Circuits and Systems*, May 2008, pp. 1236–1239.

[22] J. Park, D. Shin, N. Chang, and M. Pedram, “Accurate modeling and calculation of delay and energy overheads of dynamic voltage scaling in modern high-performance microprocessors,” in *Proc. IEEE International Symposium on Low-Power Electronics and Design*, 2010.

[23] B. H. Calhoun and a. P. Chandrakasan, "Ultra-dynamic voltage scaling (UDVS) using sub-threshold operation and local voltage dithering," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 238–245, 2006.

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

[25] S. Dighe, S. Gupta, V. De, S. Vangal, N. Borkar, S. Borkar, and K. Roy, "A 45nm 48-core IA processor with variation-aware scheduling and optimal core mapping," in *Proc. IEEE Symposium on VLSI Circuits*, no. May 2009, 2011, pp. 278–279.

[26] A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann, "Power management architecture of the 2nd generation Intel® Core microarchitecture, formerly codenamed Sandy Bridge," in *Hot Chips*, vol. 23, 2011.

[27] A. Nalamalpu, N. Kurd, A. Deval, C. Mozak, J. Douglas, A. Khanna, F. Paillet, G. Schrom, and B. Phelps, "Broadwell: A family of IA 14nm processors," in *Proc. IEEE Symposium on VLSI Circuits*, 2015.

[28] E. Burton, G. Schrom, F. Paillet, J. Douglas, W. Lambert, K. Radhakrishnan, and M. Hill, "FIVR — Fully integrated voltage regulators on 4th generation Intel Core SoCs," in *Proc. IEEE Applied Power Electronics Conference*, March 2014, pp. 432–439.

[29] "Desktop 4th Generation Intel Core Processor Family, Desktop Intel Pentium Processor Family, and Desktop Intel Celeron Processor Family," Intel Corporation.

[30] R. Jevtic, H. Le, M. Blagojevic, and S. Bailey, "Per-core DVFS with switched-capacitor converters for energy efficiency in manycore processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 1–8, 2014.

[31] F. Xie, M. Martonosi, and S. Malik, "Compile-time dynamic voltage scaling settings: Opportunities and limits," in *Proc. ACM Conference on Programming Language Design and Implementation*, 2003, pp. 49–62.

[32] M. Martonosi, D. Clark, V. Reddi, D. Connors, and D. Brooks, "A dynamic compilation framework for controlling microprocessor energy and performance," in *Proc. IEEE/ACM International Symposium on Microarchitecture*, 2005, pp. 271–282.

[33] K. Choi, R. Soma, and M. Pedram, “Fine-grained dynamic voltage and frequency scaling for precise energy and performance tradeoff based on the ratio of off-chip access to on-chip computation times,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 1, pp. 18–28, 2005.

[34] Q. Wu, P. Juang, M. Martonosi, and D. W. Clark, “Formal online methods for voltage/frequency control in multiple clock domain microprocessors,” in *Proc. International Conference on Architectural Support for Programming Languages and Operating Systems*, vol. 32, no. 5, Dec. 2004, p. 248.

[35] M. Martonosi and D. Clark, “Voltage and frequency control with adaptive reaction time in multiple-clock-domain processors,” in *Proc. International Symposium on High-Performance Computer Architecture*, 2005, pp. 178–189.

[36] G. Magklis, P. Chaparro, J. González, and A. González, “Independent front-end and back-end dynamic voltage scaling for a GALS microarchitecture,” in *Proc. International Symposium on Low Power Electronics and Design*, New York, New York, USA, Oct. 2006, p. 49.

[37] D. Marculescu, “On the use of microarchitecture-driven dynamic voltage scaling,” in *Proc. Workshop on Complexity-Effective Design*, 2000.

[38] H. Li, C. Y. Cher, K. Roy, and T. N. Vijaykumar, “Combined circuit and architectural level variable supply-voltage scaling for low power,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 5, pp. 564–575, 2005.

[39] K. Rangan, G. Wei, and D. Brooks, “Thread motion: fine-grained power management for multi-core systems,” in *Proc. International Symposium on Computer Architecture*, 2009, pp. 302–313.

[40] A. Bhattacharjee and M. Martonosi, “Thread criticality predictors for dynamic performance, power, and resource management in chip multiprocessors,” in *Proc. International Symposium on Computer Architecture*, vol. 37, no. 3, Jun. 2009, p. 290.

[41] G. Semeraro, G. Magklis, R. Balasubramonian, D. H. Albonesi, S. Dwarkadas, and M. L. Scott, “Energy-efficient processor design using multiple clock domains with dynamic

voltage and frequency scaling," in *Proc. International Symposium on High Performance Computer Architecture*, 2002, pp. 29–40.

[42] A. Iyer and D. Marculescu, "Power efficiency of voltage scaling in multiple clock, multiple voltage cores," in *Proc. IEEE International Conference on Computer-Aided Design*, vol. 78741, no. 4901, 2002, pp. 379–386.

[43] RISC-V Rocket Chip. [Online]. Available: https://github.com/ucb-bar/rocket-chip

[44] The RISC-V ISA. [Online]. Available: http://riscv.org