UC Berkeley

UC Berkeley Electronic Theses and Dissertations

Title

Energy-Efficient System Design Through Adaptive Voltage Scaling

Permalink

https://escholarship.org/uc/item/0sn700n1

Author

Keller, Benjamin Andrew

Publication Date

2017

Peer reviewed|Thesis/dissertation

eScholarship.org

Powered by the California Digital Library
University of California

Energy-Efficient System Design Through Adaptive Voltage Scaling

by

Benjamin Andrew Keller

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Borivoje Nikolić, Co-chair
Professor Krste Asanović, Co-chair
Associate Professor Duncan Callaway

Fall 2017

Energy-Efficient System Design Through Adaptive Voltage Scaling

Copyright 2017
by
Benjamin Andrew Keller

Abstract

Energy-Efficient System Design Through Adaptive Voltage Scaling

by

Benjamin Andrew Keller

Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Borivoje Nikolić, Co-chair

Professor Krste Asanović, Co-chair

Improving the energy efficiency of processor systems-on-chip (SoCs) is key to improving the performance and utility of thermally-limited servers, battery-constrained mobile devices, and energy-harvesting Internet-of-Things nodes. Adaptive voltage scaling adjusts voltage levels at runtime in response to changes in workload, reducing voltage when additional performance is not required. The granularity of this voltage scaling greatly impacts its effectiveness. Fine-grained adaptive voltage scaling (FG-AVS) in time reduces wasted energy by tracking microsecond-scale changes in workload; FG-AVS in space independently adjusts voltage levels in many different portions of an SoC, saving additional energy. FG-AVS has the potential to save considerable energy, but it has not yet seen adoption in commercial systems due to several challenges that complicate its implementation. Integrated voltage regulators are required to supply the many voltages required and switch quickly between modes, but efficient implementations impose high area overhead. Clocking, synchronization, and power management have their own requirements and difficulties.

This work presents key components of fully-featured SoCs that enable demonstration of FG-AVS. Integrated simultaneous-switching switched capacitor voltage regulators are presented that achieve high conversion efficiency when coupled with an adaptive clock generator. Power management is accomplished with programs run on a dedicated power management unit (PMU), and a pausible bisynchronous FIFO circuit that can achieve low-latency communication between asynchronous voltage domains is described. These components are integrated into systems that demonstrate the potential energy savings of FG-AVS. The Raven-3 testchip demonstrates efficient voltage regulation supplying a complicated digital load; the system achieves 26.2 GFLOPS/W while operating its processor under the generated supply voltage and adaptive clock. The Raven-4 testchip includes integrated power management circuits that allow fully integrated feedback for FG-AVS. The programmable PMU can run a wide variety of power management algorithms, including an AVS algorithm that saves 39.6% energy in a synthetic microbenchmark with minimal performance penalty. The Hurricane-1 testchip implements multiple independent voltage domains and hardware counters that can be used for power management. The Hurricane-2 testchip features finer spatial partitioning and more effective instrumentation for power management; simulation results show up to 13.3% energy savings for an algorithm exercising FG-AVS in time and 46.0% energy savings for an algorithm using FG-AVS in space. Together, these testchip implementations show the potential of FG-AVS to save energy in production SoCs.

To Helen.

Contents

List of Figures

List of Tables

Acknowledgments

I owe a tremendous debt of gratitude to those who made this dissertation possible. While all research stands on prior work, I think it's fair to say that to an extraordinary extent, my research would not have been feasible without all of those who contributed to the big ideas and massive projects that made this work happen.

I would be remiss if I did not begin by thanking my advisors, Bora Nikolić and Krste Asanović. The first aspects of this project began well before I arrived at Berkeley, and it is their research vision and tireless persistence that allowed my research to progress. Bora has never been afraid to think big, in terms of our research goals, our potential achievements, and the extent of our broad project collaborations. Without his leadership, this project would have surely foundered. Krste oversaw the development of an entire new architecture research infrastructure over just a few years, all while taking pains to convince every computer architect in earshot that tapeouts matter. The unique confluence of these two visions was the foundation for my work. Thanks for all of this, and for the million things I haven't space to mention.

Mixed-signal tapeouts are fantastically challenging; I am extremely fortunate to have worked with such a talented group of colleagues throughout my graduate career. Most importantly for my own work, Alberto Puggelli, Ruzica Jevtic, and Hanh-Phuc Le invented and implemented the voltage regulation that was the lynchpin of our chips and an absolute requirement for my own research. I cannot overstate the extent to which my own experimentation relies on their outstanding work. Jaehwa Kwak designed the clock generators that were critical to the high efficiency of our systems; his layout prowess remains unmatched. Brian Zimmer designed the custom memories that allowed our chips to operate at low voltages. He also served as a tapeout guru, circuit expert, and indefatigable mentor throughout his time at Berkeley. Yunsup Lee was responsible for many of the digital blocks that made up our tapeouts, including the vector processors and the first power management unit, but his most important contribution was his unhesitating commitment to dreaming big and achieving that vision. Martin Cochet designed the power monitoring hardware and temperature sensors, and Milovan Blagojević designed the integrated body bias generator; these two visiting students also provided invaluable tapeout help in understanding the STMicroelectronics process and working with our French collaborators. Pi-Feng Chiu and Stevo Bailey were my partners in crime through every tapeout. System integration is a thankless task, but their tapeout expertise remains invaluable, and I am in their debt for the many, many hours they devoted to this project. John Wright designed the serial links in the Hurricane tapeouts and integrated them into the memory system. Jarno Salonaa made improvements to the integrated voltage regulators in Hurricane-2. To these students, as well as Alon Amid, Keertana Settaluri, Jessica Iwamoto, and everyone else who made it possible to teach sand to think, thank you.

In addition to circuits, our tapeouts relied heavily on RISC-V, Chisel, and the Rocket Chip Generator, all technologies for which I have Krste Asanović's research group to thank. Andrew Waterman and Yunsup Lee, along with Krste and Dave Patterson, invented and

promoted the RISC-V ISA, and contributed a tremendous amount of software infrastructure that has proved invaluable to our efforts. The Chisel hardware description language made it possible to achieve the design productivity that we did, and I'm grateful to Jonathan Bachrach for conceiving and promoting the effort. The Rocket Chip Generator made it possible for us to tape out real processors; thanks to Yunsup, Andrew, and Henry Cook for remaining committed to open-source hardware generators. Palmer Dabbelt is a systems wizard; he made many an impossible problem disappear without breaking a sweat. Colin Schmidt made building and programming vector machines possible. Howie Mao has written countless hardware blocks that just happened to do exactly what we needed. Stephen Twigg spent many hours helping me understand the capabilities of Chisel. And my other architecture groupmates, Eric Love, Martin Maas, David Biancolin, Jack Koenig, Adam Israelevitz, Albert Magyar, Scott Beamer, Donggyu Kim, Albert Ou, Sagar Karandikar, and Rimas Avizienis, have contributed in numerous ways to our research ecosystem, as well as teaching me to reason about computer architecture. Thanks to the entire team, as well as everyone who has contributed to the open-source RISC-V, Rocket, and Chisel ecosystems.

In addition to those with whom I worked directly, many students at the Berkeley Wireless Research Center have made my research possible. My ignorance of analog design runs deep, and so I'm indebted to Sameet Ramakrishnan, Nathan Narevsky, Lucas Calderin, and Greg Lacaille for their patient explanations and for showing me around the lab. Thanks to all of the students in Bora's research group, including Angie Wang, Rachel Hochman, Antonio Puglielli, Paul Rigge, Amy Whitcombe, Nick Sutardja, Mira Videnović-Mišić, Katerina Papadopoulou, Milos Jorgovanovic, Sharon Xiao, Matt Weiner, Amanda Pratt, and Charles Wu, who provided useful research feedback and copious grad school advice while generally keeping things sane. Undergraduate students Gary Choi and Vighnesh Iyer respectively contributed a power virus and FPGA expertise to the project. The BWRC is a great place to work and learn, and it's my colleagues that make it so.

Faculty and staff have greatly contributed to my Berkeley experience. Thanks to Vladimir Stojanovic and Duncan Callaway for serving on my qualifying exam committee and providing crucial feedback on my research. In addition to overseeing the initial work on voltage regulation, Elad Alon has never hesitated to answer questions and provide much-needed guidance and perspective. Brian Richards has the unglamorous role of negotiating myriad CAD tools, foundries, license agreements, and everything in between; his tireless work makes it possible for us to do our research. James Dunn contributed many of the board designs so we could test our chips. Many thanks to Ubirata Coelho and Kostadin Ilov for keeping the servers running, and to Fred Burghardt for maintaining the lab. Thanks also to the front-office staff at BWRC and the ASPIRE Lab. Olivia Nolan, Candy Corpus, Leslie Nishiyama, Erin Hancock, Yessica Bravo, Roxana Infante, and Tami Chouteau kept the lights on and handled so much behind the scenes so that we could do our work.

Many people outside of Berkeley also contributed to this research. I'm particularly indebted to Brucek Khailany and Matt Fojtlik, who served as my mentors at NVIDIA Research. Their support and direction provided the foundation for Chapter 5 of this dissertation. Thanks also to Bill Dally, Yan Zhang, John Poulton, Tezaswi Raja, Stephen Tell, and John Edmonson for their invaluable discussions and feedback at NVIDIA. I would also like to thank Alon Naveh and Mark Reynolds of Intel, Tom Burd of AMD, Luke Shin and Frankie Liu of Oracle, and Dave Ditzel of Esperanto for taking an early interest in my work and providing valuable feedback as I continued it. My experiences and conversations with industry members yielded valuable guidance and grounding for the project.

My research was funded by the National Science Foundation Graduate Research Fellowship Program, the NVIDIA Graduate Fellowship, Intel Research, and GRC under Grant SRC/TxACE 1836.136. Aspects of the project were also funded by DARPA PERFECT Award Number HR0011-12-2-0016, AMD, and the Marie Curie FP7. Fabrication of all four testchips was donated by STMicroelectronics; I'm particularly grateful to Andreia Cathelin and Phillippe Flatresse at ST, whose tireless advocacy of our projects and our collaboration made our chip development possible. Thanks to all of these organizations, as well as all of the sponsors of BWRC and the ASPIRE Lab.

Finally, I would like to thank my family, without whom I certainly would not be where I am today. My parents, to whom I am eternally grateful, supported my decision to pursue twenty-four straight years of schooling without batting an eye. My brother has stuck with me through it all, a dependable, insightful sounding board for all of my frustrations and triumphs. And my wife Helen, who made it through problem sets, prelims, evening meetings, paper deadlines, tapeouts, qualifying exams, all-nighters, conference travel, and the inevitable existential uncertainty, somehow still wanted to marry me afterwards. Thank you.

Chapter 1

Introduction

Fine-grained adaptive voltage scaling (FG-AVS) is a powerful energy-saving technique that can be used to improve the efficiency of a wide range of system-on-chip (SoC) designs. By adjusting the operating voltage and frequency of each part of an SoC to track workload requirements, energy consumption can be greatly reduced during periods of low activity. The implementation of FG-AVS requires extensive changes in design methodology, presenting many challenges to successful adoption. However, key technology trends in deeply scaled process technologies make the use of FG-AVS increasingly important. This dissertation presents innovation in voltage regulation, clock generation, synchronization, power management, and system integration that enable these challenges to be overcome, paving the way for the widespread use of FG-AVS in modern digital design.

This chapter explains the fundamentals of FG-AVS, and motivates its use in SoC design. First, the critical importance of energy efficiency in all modern computing applications is reviewed. Next, the basic concept of FG-AVS is described, with examples of how reducing spatial and temporal granularity can save energy compared to the coarse-grained baseline. Several technology trends are examined to further motivate FG-AVS design by describing several co-benefits of the design technique. Finally, the chapter concludes with an outline of the remainder of the dissertation.

1.1 Motivation

Improving energy efficiency both motivates and enables considerable innovation in the design of semiconductor circuits. The cost savings enabled by Moore's Law have led to the production and deployment of a huge number of SoCs, each containing millions or billions of transistors. The cloud services provided by datacenters have quickly become essential to businesses and consumers, while embedded devices are ubiquitous in every sector of the economy. The energy required to power these devices therefore has considerable impact on the total energy consumption of the entire economy. As shown in Figure 1.1, one estimate finds that a halt to energy efficiency improvements would result in a near-doubling of U.S.

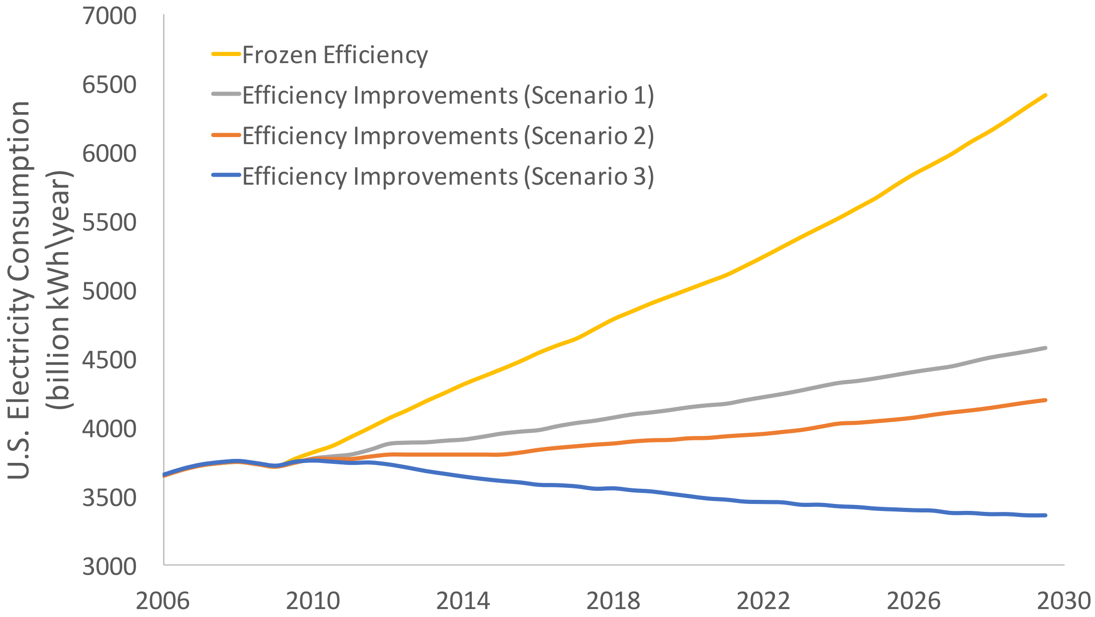


Figure 1.1: Results of a model estimating total U.S. electricity consumption by 2030 [1]. The frozen efficiency curve assumes no efficiency improvements past 2010, causing electricity consumption to grow with the economy. Scenario 1 shows the effect of implementing existing policies to improve energy efficiency. Scenario 2 shows the further reductions possible by the full deployment of existing technological efficiency advances. Scenario 3 shows the more aggressive reductions possible when all possible semiconductor-enabled efficiency improvements are included.

electricity demand from 2010 to 2030, while aggressive improvements to efficiency enabled by semiconductor innovation and deployment will instead decrease overall demand. Efficiency improvements therefore stand to dramatically reduce the need for additional electrical generating capacity and to mitigate the related impacts of greenhouse gas emissions on global climate change.

Notwithstanding the global impacts, improvements in the energy efficiency of future SoCs is required for continued improvements to their performance and utility. Datacenters are constrained in both cost and performance by the energy efficiency of the deployed silicon, while battery-powered devices must limit their energy consumption to retain utility. The following sections detail the impact of energy efficiency on each of these key use cases in more detail.

1.1.1 Energy Efficiency in Datacenters

Datacenters are the key driver of continued high-end processor demand. As more services are migrated to networks, and Internet services are increasingly consolidated into a handful

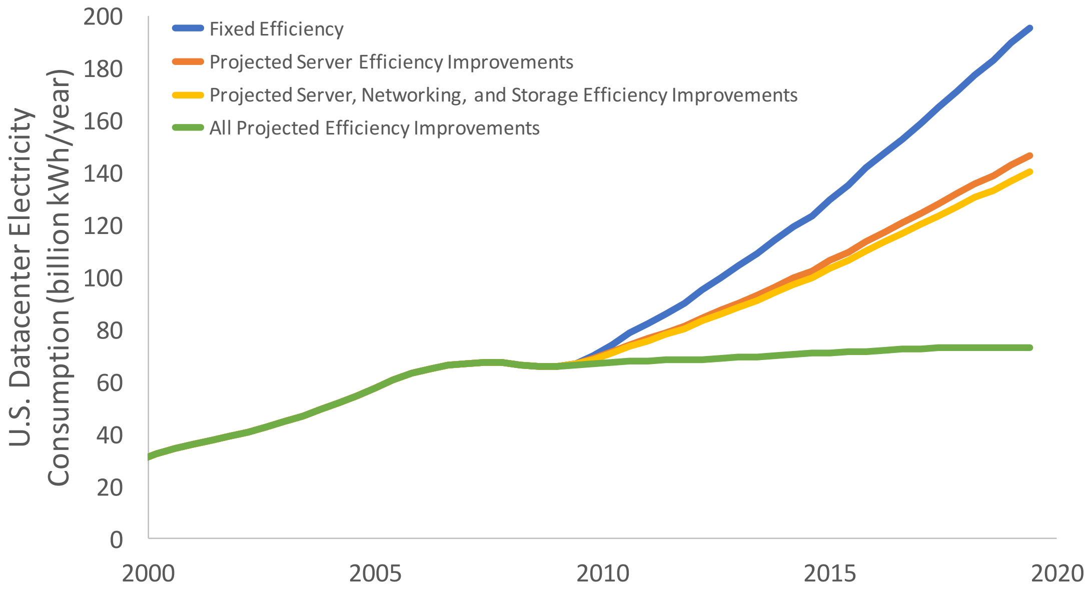


Figure 1.2: Estimated energy savings from current trends in US datacenter efficiency improvements [2]. Energy efficiency improvements are predicted to save a total of 620 billion kW h over a decade.

of massive cloud providers, the need for energy-efficient compute is driven by increased demand and competitive cost-cutting. In 2014, datacenters in the US consumed electricity equivalent to 6.4 million American homes [2]. Improvements in energy efficiency are the key to keeping up with the demand for these services while avoiding a massive increase in electricity consumption that would be required by a fixed-efficiency scenario (see Figure 1.2).

Faster computation in datacenters is fundamentally limited by SoC energy efficiency because of package thermal limits that have prevented a decade of naive performance improvements. Twenty years ago, desktop computer performance could be rapidly increased by taking advantage of Dennard scaling to increase clock rates and deepen processor pipelines. In the early 2000s, these efforts hit a “power wall” that was famously described in a 2001 presentation [3]. As shown in Figure 1.3, SoC power densities were rapidly becoming untenable; silicon chips would certainly melt and cease to function far before reaching power densities equivalent to those of the surface of the sun. These thermal limits forced fundamental changes in SoC design that have persisted even as server chips have replaced desktops as the main drivers of high-end processor development. While servers may be able to deploy more sophisticated and exotic cooling technologies, the cost is often prohibitive, and so improvements to processor energy efficiency remain the primary means of increasing the performance of datacenter SoCs within their power budget.

In addition to demands for performance, datacenter designers have become increasingly attuned to energy costs as a significant operating cost in themselves. As datacenters have ballooned into their own industry, designers looking to reduce costs must consider the total

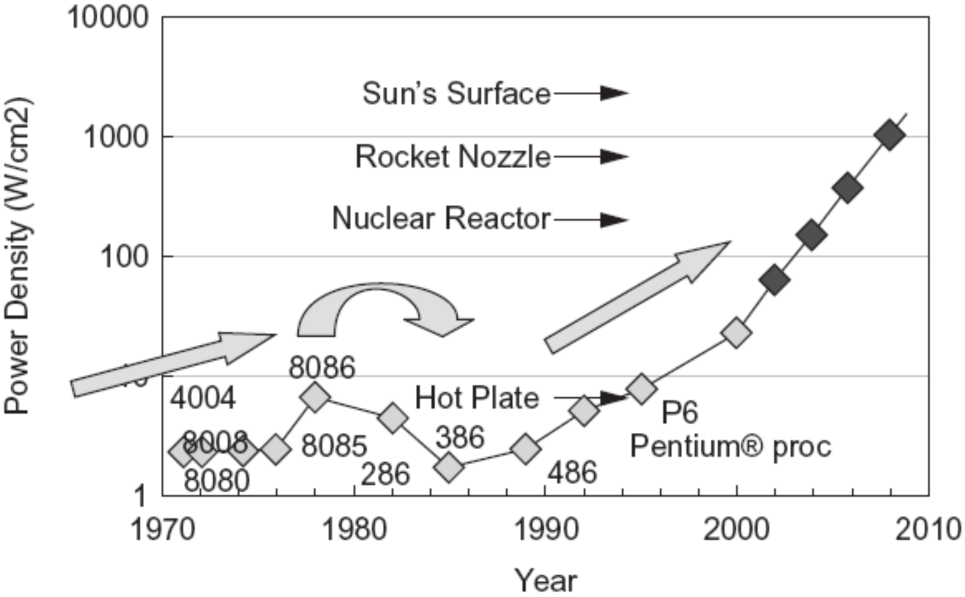


Figure 1.3: Predictions of silicon power density in 2001 as first summarized by Shekhar Borkar of Intel [3] (© 2001 IEEE). As explained by Pat Gelsinger, then CTO of Intel, "If scaling continues at present pace, by 2005, high-speed processors would have power density of nuclear reactor, by 2010, a rocket nozzle, and by 2015, the surface of the sun."

cost of ownership (TCO) to remain competitive. The entire operating cost over the lifetime of the datacenter, from the cost of land to the hiring of building security, must be considered together. While TCO estimates can vary considerably based on model assumptions, several studies show that electricity and cooling costs can be 10-20% of typical TCO [4, 5]. Furthermore, new datacenter designs focused on energy efficiency can reduce TCO, even if the hardware itself may be more expensive [6]. Both improved performance and overall cost reduction demand more energy-efficient silicon for datacenters.

1.1.2 Energy Efficiency in Mobile Devices

The explosion of phones and tablets have created a massive market for performant, energy-efficient mobile SoCs that barely existed a decade ago. The processors in these devices must service the occasional demand for high-performance compute, but their most important objective is high energy efficiency that will allow long-lasting battery life. Portable batteries have not undergone volumetric scaling to keep pace with Moore's Law. Even as battery energy densities have increased over time, manufacturers have largely refrained from dramatically increasing battery capacity across product generations, instead trading off the improved capacity for thinner and lighter form-factors. Improvements to battery life therefore must be enabled by improvements to SoC energy efficiency.

Consumers intuitively understand that even a highly performant processor is of no use if the battery of their device has lost charge. Surveys consistently show that long battery life is one of the most desired phone features; one report indicates that 92% of those surveyed say that battery life is important when considering a new smartphone purchase [7]. These consumer preferences easily align with the goals of smartphone and tablet vendors to make their devices more useful. Accordingly, SoC energy efficiency improvements remain critical drivers of utility improvements for mobile form factors.

1.1.3 Energy Efficiency in IoT Devices

The Internet-of-Things (IoT) is a shorthand for widespread deployment of Internet-connected devices embedded in everyday objects. This new field envisions entirely new classes of SoCs, deployed in a diversity of applications from agriculture to healthcare to transportation, which can improve functionality and efficiency. While some IoT devices have been incorporated into large, stationary appliances that consume significant power, many applications are only useful when the device is fully mobile and battery-powered. IoT devices that draw milliwatts of power can be powered by small batteries, and can find use in drones, industrial sensors, and consumer appliances. However, even more energy-efficient devices that consume just microwatts of power could enable devices powered by tiny batteries, small solar cells, or piezoelectric energy harvesters. This smaller form-factor and footprint could enable new categories of applications, such as wearable devices, implantable devices, and ubiquitous sensor networks.

Innovation in IoT applications will therefore be driven by improved energy efficiency; reductions in energy footprint will not just make existing IoT SoCs perform existing tasks more effectively, but will allow their application to entirely new problems. The power consumption of existing microcontrollers is often dominated by the processor, even in devices with embedded radios [8], illustrating the critical role that improving the energy efficiency of digital logic must play in growing the Internet of Things.

1.2 Fundamentals of Fine-Grained Adaptive Voltage Scaling

Supply voltage and operating frequency are two key constraints of digital systems. Supply voltage can be adjusted, changing the parameters of the supplied transistors; this is known as voltage scaling. Dynamic voltage scaling adjusts supply voltage during circuit operation; because adjusting the voltage impacts the best frequency achievable by the digital block, voltage and frequency are often adjusted simultaneously, so this technique is commonly known as dynamic voltage and frequency scaling (DVFS). Adaptive voltage scaling (AVS) performs DVFS in response to some measured change in conditions in a feedback loop executed at runtime. AVS can be used to save energy because of the fundamental power-performance characteristics of the devices under voltage scaling.

The switching power consumption P_{dyn} of digital gates operating in saturation is given by the relationship

$$P_{dyn} = \alpha C V_{DD}^2 f \tag{1.1}$$

where α is the proportion of gates switching each cycle, C is the total switching capacitance, V_{DD} is the supply voltage, and f is the switching frequency. Therefore, total dynamic energy E_{dyn} per cycle, which is independent of the switching frequency, is given by

$$E_{dyn} = P_{dyn} \cdot 1/f = \alpha C V_{DD}^2 \tag{1.2}$$

So a reduction in supply voltage corresponds to a quadratic reduction in dynamic energy.

Static power consumption cannot be accurately predicted by supply voltage with a simple analytical model, but more complex models and published data both yield a super-linear relation; that is, a reduction in supply voltage results in a proportionally greater reduction in leakage power. The energy savings of this effect are mitigated because the best achievable operating frequency also decreases with lower supply voltage, so leakage energy must be summed over a longer cycle time. The operating frequency of digital logic is proportional to the ON current I_{ON} of the devices used in the digital gates. While accurate models of the dependence of frequency and supply voltage can be quite complex, the relationship can be approximated as linear, an approximation that fits well with measured data in modern processes [9, 10]. This means that even though devices leak for longer, the super-linear relationship of voltage to static power consumption implies that total leakage energy per cycle will nonetheless decrease with voltage.

To summarize these effects, when voltage is decreased, performance decreases linearly, but energy consumption drops even more. This fundamental relationship has enabled decades of voltage-scaling innovation [11]. Nearly all modern SoCs employ some form of voltage scaling, in which reduced performance is traded for increased energy efficiency some or all of the time. A common application of AVS is to reduce voltage to track workload demand; when peak performance is not needed, supply voltage can be reduced to save energy. AVS already represents a powerful tool to improve energy-efficiency in many classes of devices.

1.2.1 Voltage Scaling Granularity

A key determinant of the utility of voltage scaling is the available *granularity* of voltage scaling in the system. Granularity refers to the scale at which voltage changes can be applied, and the extent of available granularity is generally determined at design time.

Granularity can broadly be described in two separate ways. The first type is *temporal granularity*, which refers to the speed at which changes in operating conditions can be detected and a change in voltage and frequency actuated in response. The importance of fine temporal granularity is illustrated in Figure 1.4. In a system with coarse temporal granularity, the system is slow to respond to rapid, frequent changes in workload, and in some cases is unable to respond at all without violating performance goals. This leads to wasted

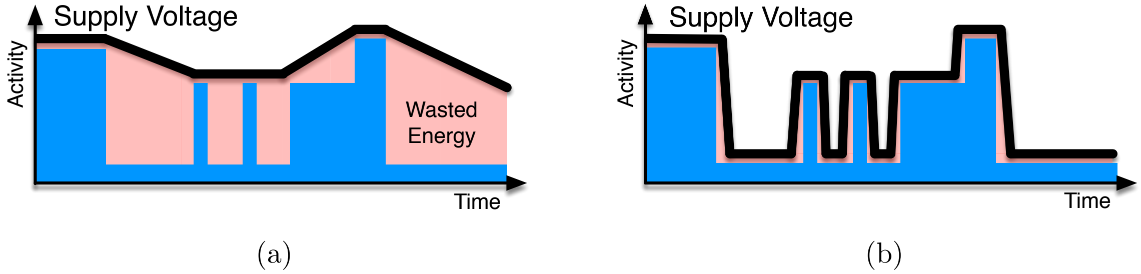


Figure 1.4: An example showing the additional energy savings of fine-grained AVS in time (b) compared to the coarse-grained AVS baseline (a).

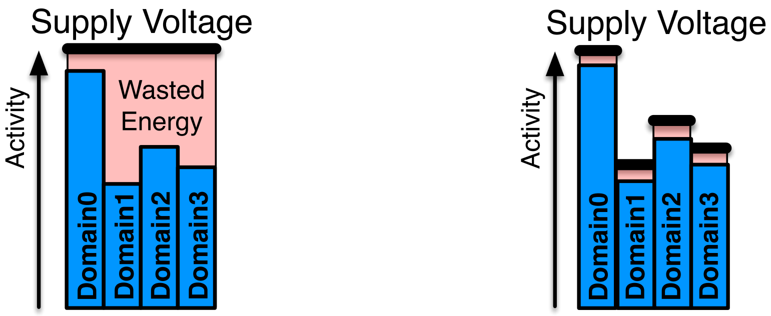
energy, periods of operation during which the system is operating at a higher voltage than necessary to meet the performance requirement. A system with finer temporal granularity, on the other hand, is able to rapidly adjust the voltage to track rapid workload changes, spending very little time at a higher voltage than necessary. A control loop with bandwidth that exceeds the frequency of workload changes can achieve the best energy savings.

The second type of granularity is *spatial granularity*, which refers to the number and size of voltage domains on a chip. Fine spatial granularity requires many small voltage areas, each of which can adjust their voltage independently, while a system with coarse spatial granularity has just one or a few independent voltage areas. The benefits of decreased spatial granularity are shown in Figure 1.5. The system with coarse spatial granularity has a single voltage domain, and so it cannot reduce its operating voltage if the most demanding performance constraint would not be met, even though other portions of the system could meet their performance needs at a lower voltage and frequency. In contrast, the system with fine spatial granularity can adjust the voltage of each domain independently to match each particular performance target, saving energy without degrading performance. In addition, finer spatial granularity permits better tracking of fine-grained temporal changes, because independent voltage domains may exhibit time-varying behavior that would not be discernible if they were grouped into a single domain. Spatial partitioning corresponding to the smallest spatial variation in workload can achieve the best energy savings.

Modern commercial systems typically implement only coarse-grained AVS in both time and space. Typical SoCs have a handful of large voltage domains, with adaptive feedback to change operating conditions acting at millisecond timescales. This dissertation intends to demonstrate the benefits of fine-grained AVS to motivate broader adoption of such designs.

1.2.2 Globally Asynchronous, Locally Synchronous Design

Implementing fine-grained spatial AVS requires many independent voltage domains. Because each of these domains may be operating at a different voltage, and the clock frequency of each domain should be tuned to operate at the best possible frequency for each voltage, each clock cannot be assumed to have any known frequency or phase relationship to any



(a) (b)

Figure 1.5: An example showing the additional energy savings of fine-grained AVS in space. (a) shows the wasted energy resulting from a coarse-grained AVS implementation, while (b) shows how energy can be saved with a fine-grained approach.

other. The clock periods are therefore asynchronous, an architecture known as Globally Asynchronous, Locally Synchronous (GALS) [12]. GALS as originally proposed consists of synchronous "islands" connected by a fully asynchronous network that is not clocked at all, but it is now common to refer to designs that consist of many internally synchronous but independent blocks as GALS designs as well [13]. GALS designs need not implement fine-grained voltage scaling, but the GALS design style is well-suited to the system requirements for FG-AVS.

1.2.3 Challenges of Fine-Grained Adaptive Voltage Scaling

FG-AVS has not yet seen widespread adoption because of several critical technical barriers. These challenges force design tradeoffs that must be carefully considered before implementing an FG-AVS system.

The foremost difficulty in implementing FG-AVS is designing circuits that can supply independent voltages to many different domains, and ensuring that the supplied voltage can switch quickly between operating modes. Voltage regulators have large design overhead, and many regulators introduce unacceptable energy losses in voltage conversion that reduce or eliminate the energy savings of AVS. Furthermore, in GALS designs, local clocks must now be generated and provided to each independent domain. The introduction of many asynchronous clock domains requires careful consideration of circuits to safely transfer data between domains; these circuits often introduce latency or other overheads. All of these implementation requirements have costs in silicon area and design complexity, increasing the challenge of meeting area budgets and confidently verifying correct operation of the design.

These challenges to implementing FG-AVS are discussed in more detail in Chapter 2. The design innovations presented in succeeding chapters demonstrate how these challenges can be overcome to implement practical systems with FG-AVS.

1.3 Technology Trends

The hurdles to implementation of FG-AVS have prevented its widespread use in commercial products thus far. However, continued trends observed in modern SoCs and advanced process nodes have exacerbated the need for FG-AVS because of the additional benefits provided by the design style. Severe process variation, changes in typical workloads, and challenges in designing very large SoCs all contribute issues that are ameliorated by the implementation of FG-AVS. These trends, in addition to the continued need for energy-saving innovation, provide additional motivation for FG-AVS adoption in modern technologies.

1.3.1 Process Variation

Process variation refers to statistical differences in transistor parameters away from the nominal design point. Variation can be observed at many different scales: some parameters vary over the run of a process, while others vary from wafer to wafer, die to die, or spatially within a die. Some process variation is purely stochastic, varying from one device to the next. Random dopant fluctuation (RDF), or differences in the number of dopant atoms in each transistor, is the primary cause of this stochastic variation. The effects of RDF are worse at deeply scaled process nodes because there are smaller numbers of dopant atoms per device. This means that the same absolute variation in the number of atoms leads to a larger relative change in device characteristics [14]. Deeply scaled processes also continue to operate at lower nominal voltages, further worsening variation [15]. Furthermore, new sources of variation such as random telegraph noise (RTN) can exceed the effects of RDF at cutting-edge process nodes [16]. These effects result in an extremely challenging design environment, requiring large margins to guarantee correct operation.

Design techniques necessary to perform FG-AVS are ideally suited for resilient operation that can tolerate and adapt to process variation. Fine granularity in space allows each voltage domain to adjust its operating condition in response to spatial variation. Small time granularities permit continuous adaptation to time-varying conditions such as RTN. These techniques combine to reduce necessary margin, increase resiliency, and save further energy.

1.3.2 Workloads

All processor workloads exhibit phases, portions of the program with different instruction mixes and operating characteristics. FG-AVS is best suited to saving energy compared to its coarse-grained alternative when workload phases vary rapidly and frequently, presenting opportunities for fine-grained tracking to which a less aggressive system could not adapt. In recent years, bursty workloads of this type have become much more common, driven by the move to cloud services, mobile devices, and new IoT implementations [17–20]. Cloud service providers often support workloads like search or database lookups, which may happen infrequently, triggered by user request. Similarly, mobile devices such as tablets and phones are frequently idle, even when actively being used; durations of relative inactivity are punctuated by key presses or other external stimuli that demand a fast response. IoT devices are rarely triggered directly by users, instead relying on stimulation from the environment to actuate computation. The sensor inputs that cause execution are frequently bursty as well, activating upon intermittent and unpredictable changes in the environment, while devices that activate at regular intervals tend to operate for very short durations. In all of these computing environments, frequently varying workloads are ideally suited for the energy-saving benefits of FG-AVS.

1.3.3 SoC Design

The tremendous engineering effort required to bring a design to a completed layout, combined with the cost of mask generation and fabrication in modern processes, have made modern silicon production an extremely expensive endeavor, and costs are only increasing as process technology continues to scale. Design techniques that can reduce engineering work are increasingly important to mitigate these nearly prohibitive costs.

One key driver of engineering cost in a large SoC is physical design and timing signoff, the process of ensuring that a design will meet all timing constraints across dozens or hundreds of process corners. One study estimates that physical design and the associated signoff tasks can cost up to $20 million for a large chip in a 16 nm process [21]. These costs continue to grow as transistor counts (and so the number of timing paths) increase with process scaling, but they are also being driven by continued increases in overall chip area, which can exceed 800 mm^2 in cutting-edge GPUs [22]. Traditional designs require timing closure over large, synchronous voltage areas, with a huge number of timing paths that incur long tool runtimes to analyze and resolve [23]. In contrast, GALS designs employing FG-AVS consist of many smaller, independently clocked domains. As there are no long synchronous paths that cross large areas of the chip, GALS designs effectively eliminate the need for global timing closure, reducing engineering effort to the much simpler task of closing timing within each synchronous block. This reduction in design effort will continue to increase in impact as the drive to pack more transistors into a die further complicates physical design.

1.4 Dissertation Outline

This work presents a series of circuits, architectures, software, and test chip implementations demonstrating the feasibility of implementing systems capable of FG-AVS. Realistic systems are designed and fabricated in a modern process technology, with measurements confirming the energy-saving benefits of the approach.

Chapter 2 details the primary barriers to implementing systems with the ability to perform FG-AVS. These challenges, including integrated voltage regulation, clock generation, clock-domain crossings, overheads, and design complexity, are explained in detail, together with prior work.

Chapter 3 introduces a novel design of an integrated switched-capacitor voltage regulator that is able to meet the demands of FG-AVS while achieving high conversion efficiencies and reasonable power density. A scheme for per-domain adaptive clock generation, which is required for high regulator efficiency, is also detailed, and the implications of these circuits on the design of the full SoC are outlined. These circuits are the core enabling technologies of our FG-AVS implementation.

Chapter 4 describes hardware and software required for effective power management for FG-AVS. Different methods of estimating workload are presented, and the hardware architectures chosen to implement fully integrated feedback are described. The chapter concludes with a discussion of different energy-saving algorithms that can be implemented in an AVS controller.

Chapter 5 discusses solutions to the difficulty of safely communicating information between asynchronous clock domains, a frequent requirement in FG-AVS systems. A novel scheme for safe data movement based on the concept of pausible clocking is explained, and implementation details and simulation results are provided to confirm the efficacy and practicality of the approach.

Chapter 6 describes the synthesis of integrated voltage regulation, adaptive clock generation, and power management hardware into a series of four test chip implementations. The design and implementation details of each system are summarized, and measurement and simulation results that provide evidence of the efficacy of FG-AVS are shown.

Chapter 7 concludes the work by summarizing the key contributions and suggesting potential future directions.

Chapter 2

Challenges of Adaptive Voltage Scaling

Fine-grained adaptive voltage scaling is a powerful technique to save energy in SoCs, but commercial adoption has been limited because of the significant challenges and overheads that occur when implementing a system capable of FG-AVS. Generating many independent voltages, quickly switching between different voltage modes, and supplying these voltages to the independent voltage domains of the FG-AVS system requires integrated voltage regulators, circuits that are difficult to design with high efficiency. Implementing the GALS design style required for FG-AVS poses challenges in clock generation for each voltage domain and data movement between asynchronous domains. Addressing these issues to build a robust design imposes area overheads and increases design complexity. This chapter details these challenges with consideration of prior work in each of these areas and discussion of various design alternatives that could allow effective implementation of FG-AVS.

2.1 Voltage Regulation

The implementation of FG-AVS requires the delivery of multiple different voltages to different areas of the chip, and the particular voltages used must be dynamically reconfigurable. These multiple voltages are supplied by voltage regulators, circuits that convert one voltage to another. Typically, one or two input voltages will be downconverted by one or more voltage regulators to some number of lower voltages for use by the different voltage areas of the SoC.

Several important metrics are used to evaluate the effectiveness of voltage regulators. The efficiency of the regulator is the ratio of power delivered by the regulator to power supplied. Highly efficient regulators (with efficiencies near one) reduce wasted power. Most regulators are designed to function at a particular conversion ratio, a set of discrete ratios, or a range of conversion. This determines the output voltages available for use. The transition time required to switch between one output voltage and another is limited by circuit characteristics, and some regulators incur additional transition losses while switching between operating modes.

Most commercial SoCs employ discrete voltage regulators on separate dice adjacent to the supplied SoC. These discrete regulators are generally very efficient, achieving conversion efficiencies of 90% or greater, and can often generate many different output voltages. However, discrete regulators are fundamentally unsuited to FG-AVS. The use of large passives to achieve high efficiencies and reduce output noise results in slow transition times between voltage modes due to the large time constants of these devices, making fine temporal granularities impossible to achieve. Furthermore, most SoC designs are IO limited, and have only a fixed number of IOs available for power supplies. This limits the spatial granularity of AVS when using discrete regulation, because only a small number of independent supplies can be connected from external regulators to the voltage domains in the SoC. Finally, the proliferation of discrete regulators to generate many independent voltages would increase system costs, which could potentially force other design compromises that would reduce the overall benefit of FG-AVS.

Integrated voltage regulation is therefore a de facto requirement for FG-AVS systems. By building regulators into the same die as the SoC they are supplying, system costs are reduced, transition times are shortened due to the smaller passives, and many voltages can be generated from just a few external supplies, ameliorating any issues with IO count. The remainder of this section details various alternatives for integrated regulators and implementation considerations in systems that employ them.

2.1.1 Integrated Regulation

Integrated voltage regulators are well-suited for FG-AVS, but their design and implementation presents an additional set of challenges. Voltage regulators require passive components, which might include capacitors, inductors, or both elements, to store charge for switching or to reduce noise. Implementing high-quality passives on die is not always possible; the presence of parasitic resistance and capacitance from lower-quality implementations results in lower regulator efficiencies. Designing regulators that can maintain fast transition times while still attaining reasonable efficiency is the key challenge of integrated regulation.

Some designs, particularly those requiring inductors, pursue a “semi-integrated” approach that integrates the regulator circuit with the exception of one or more passive elements which are implemented externally [24–26]. This design point is a compromise between discrete and integrated regulation, and while it enables better temporal and spatial granularity than the baseline, it is not as fine-grained as a fully integrated approach. Furthermore, such integration can require costly and complicated package and circuit board engineering, which can increase system cost.

The ideal integrated regulator achieves fast mode transitions and high conversion efficiency while avoiding additional process or packaging requirements and limiting area overhead. Three circuit techniques for regulation, low dropout regulators (LDOs), buck converters, and switched-capacitor regulators, each achieve a different subset of these goals.

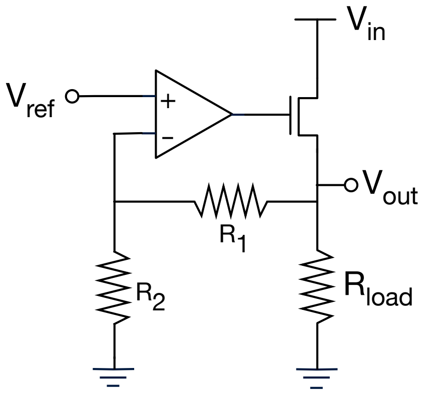


Figure 2.1: A simple LDO circuit.

Low Dropout Regulators

One power delivery circuit is known as a low-dropout regulator (LDO), a type of linear regulator that can deliver an output voltage that is only slightly lower than the input supply. A simple LDO circuit is shown in Figure 2.1. The circuit regulates the output voltage by feedback control that eliminates any difference between a scaled version of the supplied reference voltage V_{ref} and the generated voltage V_{out} observed by the load. This circuit is straightforward to design and has low area overhead. The header transistors can often be repurposed from existing power gating circuitry, further reducing the additional area required for implementation, although in some cases the amount of on-chip decoupling capacitance required to provide a robust supply can be significant [27]. The complete integration of the devices also allows fast mode transitions, and a continuous voltage range can be generated at the output simply by changing the reference voltage. For these reasons, LDOs have seen adoption in industry for use in AVS implementations, especially when only small voltage reductions are required [28, 29].

The main disadvantage of LDOs is their poor efficiency when supplying low voltages. Because the same current that flows to the load must also flow through the header transistor, the voltage drop across the header causes power to be wasted. The maximum theoretical conversion efficiency of the circuit is therefore directly proportional to the ratio of the input and output voltages. Silicon measurements confirm that LDO efficiency at low voltages is quite poor, contributing to substantial reduction in any AVS savings [30]. Accordingly, while commercial implementations make frequent use of LDOs to temporarily reduce the power consumption of the supplied block, they are not an effective regulation technique to achieve significant energy savings.

Buck Converters

Switching regulators are able to achieve higher conversion efficiencies than linear regulators because power-delivery transistors are switched on and off instead of constantly incurring

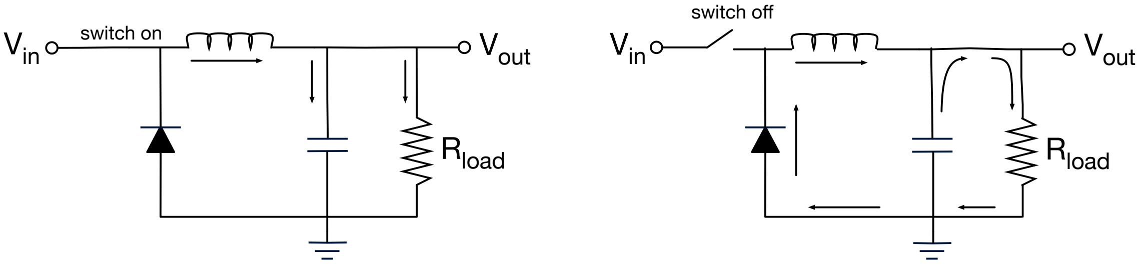


Figure 2.2: The two operating phases of a buck converter circuit [31]. The arrows denote the direction of current flow.

resistive losses. One type of switching regulator is an inductor-based buck convertor, an example of which is shown in Figure 2.2. By storing energy in the passive elements in the circuit, the modulated input voltage is regulated into a constant output voltage.

Buck converters are able to achieve high conversion efficiencies only if implemented with high-quality inductors, which are difficult to design in standard processes. Without additional expensive fabrication steps such as those investigated in [32] and [33], efficiencies tend to be low [34]. Several commercial implementations have achieved high conversion efficiencies via careful package and system co-design, with tightly coupled inductors integrated either directly into the chip package or stacked into a special cutout in the printed circuit board [24–26]. These solutions compromise on cost and increased physical dimensions in order to achieve high conversion efficiencies. Furthermore, typical buck converter control schemes are designed for relatively high load currents; large variations in load current can require different control methods, increasing design complexity.

Switched-Capacitor Regulators

Switched-capacitor regulators are an alternative to buck converters that use capacitors, not inductors, as the primary energy-storing element in the design. Since the quality of integrated capacitors is intrinsically better than that of inductors using the same area resources, switched-capacitor designs are able to achieve higher efficiencies than their fully integrated buck converter counterparts [35]. A schematic of a simple two-phase switched-capacitor circuit is shown in Figure 2.3. By toggling the transistors in two phases as marked in the figure, charge is transferred onto the flying capacitor and then to the output at a stepped-down ratio from the input voltage.

Switched-capacitor regulators achieve best efficiency when they downconvert the input voltage at a fixed ratio determined by their topology [36]. Accordingly, implementations typically generate only a small number of discrete output voltages, though recent designs can produce many output levels at the cost of increased control complexity [37, 38]. The design of switched-capacitor regulators also involves a tradeoff between energy efficiency and power density, a measure of power delivered for a given capacitive area [39]. Since

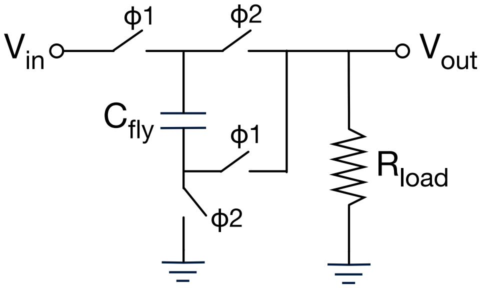


Figure 2.3: A switched-capacitor voltage regulator.

lower power densities lead to large area overheads, integrated implementations of switched-capacitor regulators often have low conversion efficiencies (although better than those of LDO implementations) [40–43]. These challenges have prevented adoption of switched-capacitor regulators in commercial SoCs.

Additional Challenges of Integrated Voltage Regulation

Regardless of implementation details, all systems employing integrated voltage regulators face additional design challenges. Foremost among these are the additional thermal constraints imposed by the inclusion of regulators into the die. Since regulators cannot be perfectly efficient, they consume some power, which manifests as heat that must be dissipated. This additional thermal load is much easier to deal with when the components are discrete and not physically co-located with the SoC itself. The lower efficiency of most integrated regulators leads to more waste heat generation, exacerbating this effect. Compensating for these additional thermal overheads can require lower-performance operation from parts of the SoC, which may be an unacceptable tradeoff.

Integrating voltage regulation to enable fine-grained spatial AVS also poses challenges pertaining to total power delivery and supply noise mitigation. Regulators that supply many small voltage domains may need to be overprovisioned relative to those that supply a single larger domain, because each smaller domain can operate at a higher power density than the chip as a whole. Furthermore, decoupling capacitance, both intrinsic and explicit, that was previously shared across the entire die now must be partitioned into much smaller segments tied to each voltage domain, either requiring additional area for decap or additional margin for more severe load spikes. These system considerations further complicate the implementation of robust integrated voltage regulation.

2.1.2 Multi-rail Switching

An alternative to integrated regulation that can still enable FG-AVS is multi-rail switching. Instead of providing a single external supply and downconverting it for use in each voltage domain, designs implementing multi-rail switching provide a small number of fixed external supplies and distribute them throughout the die. Each voltage area connects to each distributed supply by power header transistors, and a controller turns on one header at a time to set the supply voltage. Switching a domain between voltage modes is then achieved by toggling the relevant headers to that domain.

Multi-rail switching avoids entirely the complexities and tradeoffs of integrated regulation. Although there may be a small amount of resistive loss through the header transistors, there is otherwise no efficiency penalty as would be the case for regulated supplies. Furthermore, switching between different voltage modes can take place very quickly, and because large passive elements are not required for regulation, the area overhead is relatively small. These characteristics make multi-rail switching a design technique well-suited to enable FG-AVS.

This approach, however, has its own trade-offs. First, the nature of multi-rail switching allows for only a small number of discrete voltages to be supplied (most implementations use only two voltages), limiting the range of voltage scaling. While there are no explicit area overheads for large passive elements, the routing resources required to distribute multiple supplies throughout the chip can complicate the design of both the package and the power grid, potentially leaving less metal available for other purposes and impacting area budgets. Other complications arise from the switching events themselves, when a voltage domain is transitioned from one rail to another. If this switching is done too quickly, short-circuit current can temporarily flow between the two rails, causing waste power consumption and supply integrity problems. The sudden addition or subtraction of current load from a rail as an entire voltage domain is connected or disconnected can result in very large di/dt droop. Mitigating this droop can require large amounts of on-die decoupling capacitance, adding area overhead.

Several prior research efforts have implemented multi-rail switching, using different techniques to address these concerns. The authors of [44] implement dual-rail supplies that power 167 independent voltage domains, with a small processor core in each domain. They avoid di/dt events by halting the core during mode transitions, improving supply integrity but increasing transition costs. The authors of [45, 46] propose a "shortstop" technique that mitigates supply noise by making use of additional dirty supply rails used only during voltage transitions. This technique is effective if only a single domain is switching at a time, but the additional routing resources required for more rails may be prohibitive. The authors of [47] use an integrated linear regulator to slowly change the voltage level during transitions, avoiding short-circuit current and di/dt issues but greatly slowing the transition time. None of these approaches fully addresses the drawbacks of multi-rail switching, which may be why the technique has not yet seen commercial adoption.

2.1.3 Voltage Dithering

Multi-rail switching and some voltage regulator designs are only able to supply a small number of discrete voltages, but it is desirable to have a continuous range of voltage levels available to best meet any required performance target while operating at the lowest voltage

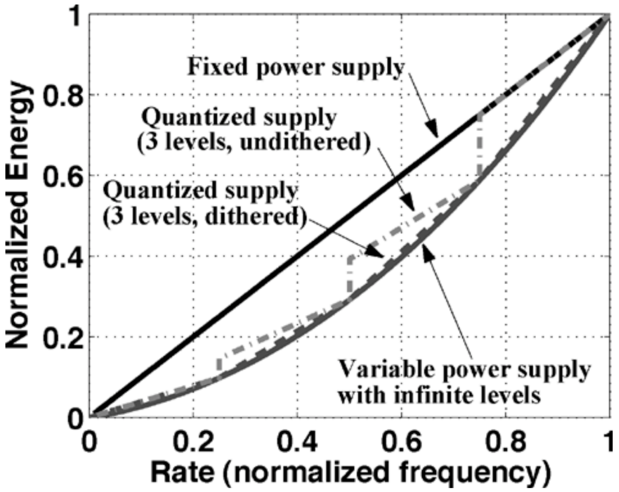
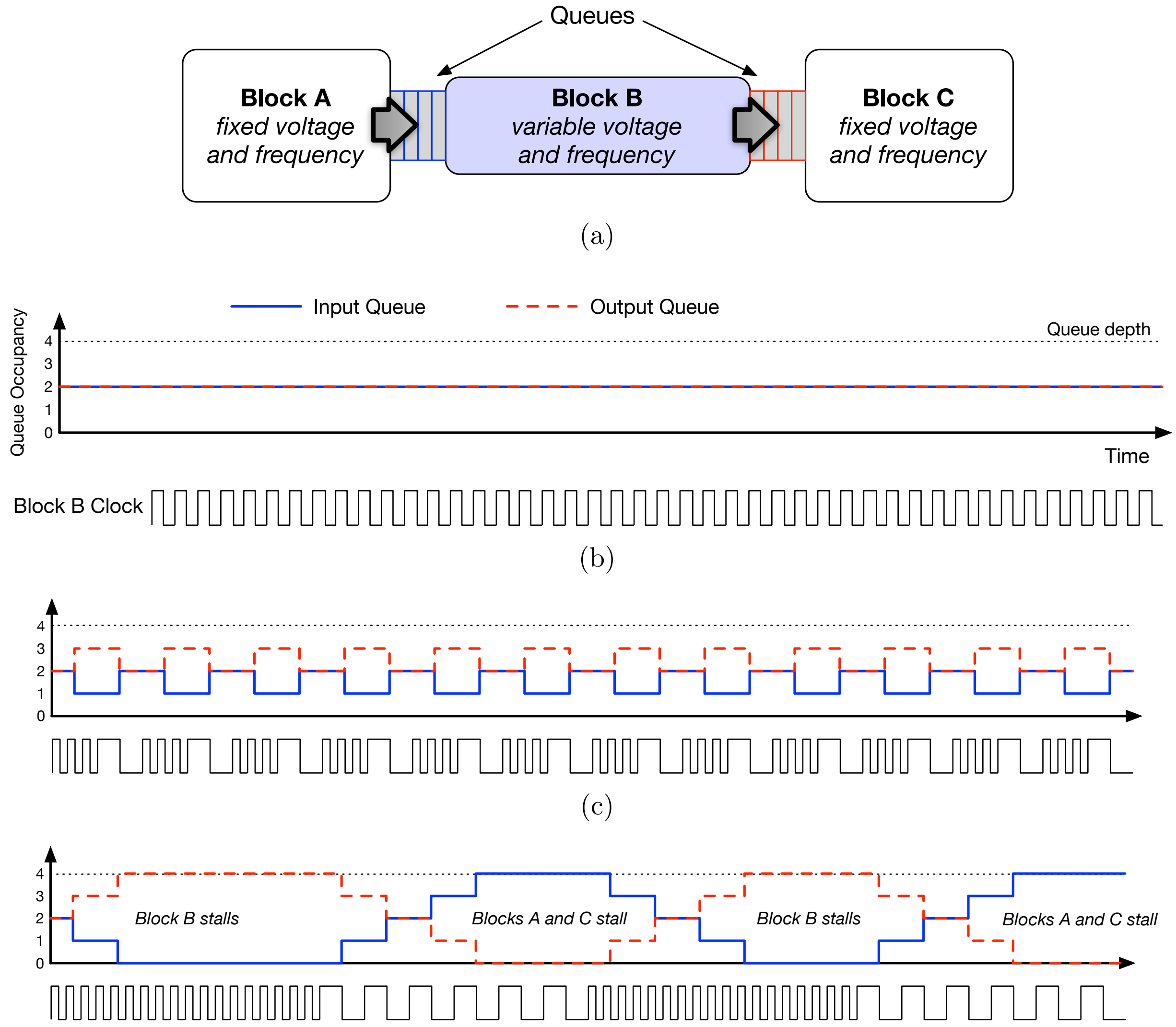


Figure 2.4: The energy cost of voltage dithering. Dithering at various duty cycles results in a linear interpolation between two points on the energy-frequency curve, resulting in marginally higher energy consumption than a regulator with arbitrary output voltage levels [48] (© 2006 IEEE).

possible. A technique known as voltage dithering can be used to overcome this issue. Dithering involves rapidly switching between two discrete voltage levels, with the duty cycle of this toggling determining an intermediate "virtual" voltage and frequency midway between the two actual operating conditions. By varying this duty cycle, voltage dithering provides a simulacrum of continuous voltage scaling. As shown in Figure 2.4, dithering linearly interpolates the virtual voltage and frequency between the two discrete points, enabling effective energy consumption only marginally higher than the case of continuous levels of regulation.

The disadvantages of voltage dithering are threefold. First, there is some efficiency loss compared to the regulated case; in situations in which the two discrete voltage levels are far apart, the effective conversion efficiency can be substantially lower than efficiency at the discrete operating points. Second, switching between different voltage modes incurs transition costs, and frequent switching increase these costs substantially. Finally, rapid dithering may simulate operation at an intermediate voltage and frequency, but prolonged operation at the discrete modes can nonetheless cause rate-balancing mismatches that degrade performance. For example, consider the system shown in Figure 2.5a, in which a voltage-scaled block is attempting to meet a clock frequency target f to rate-balance with its neighboring supplier and consumer blocks operating at this rate. The only voltages available, however, correspond to operating frequencies of $0.5f$ or $1.5f$, so the voltage is dithered with a 50% duty cycle to achieve a virtual frequency of f. If the dithering can take place at a high frequency as shown in Figure 2.5c, then the system correctly behaves as if it were operating at frequency f. However, due to transition costs of voltage mode switches or limitations in the transition time of the voltage regulators, this behavior may not be feasible. If instead voltage is dithered more slowly as shown in Figure 2.5d, the producer and consumer occasionally stall



(a)

(b)

(c)

(d)

Figure 2.5: An example showing a drawback of dithering cause by rate imbalance. Consider the system shown in (a), in which a variable-voltage Block B communicates with two fixed-voltage Blocks A and C. If Block B is able to operate at the same frequency as its neighbors as shown in (b), then data flows through the system without issue. In some cases, this operating condition may not be available or efficient, requiring dithering between two adjacent voltage/frequency pairs. If the dithering takes place rapidly as in (c), then the input and output queues do not fill, and Block B correctly simulates a virtual operating mode that matches rates with its source and sink. However, if dithering is slower as in (d), then the pipeline stalls as the queues fill and empty during the rate mismatch. Note that these examples do not account for the latency penalty of the asynchronous crossings between the blocks, which would further exacerbate rate-balancing issues.

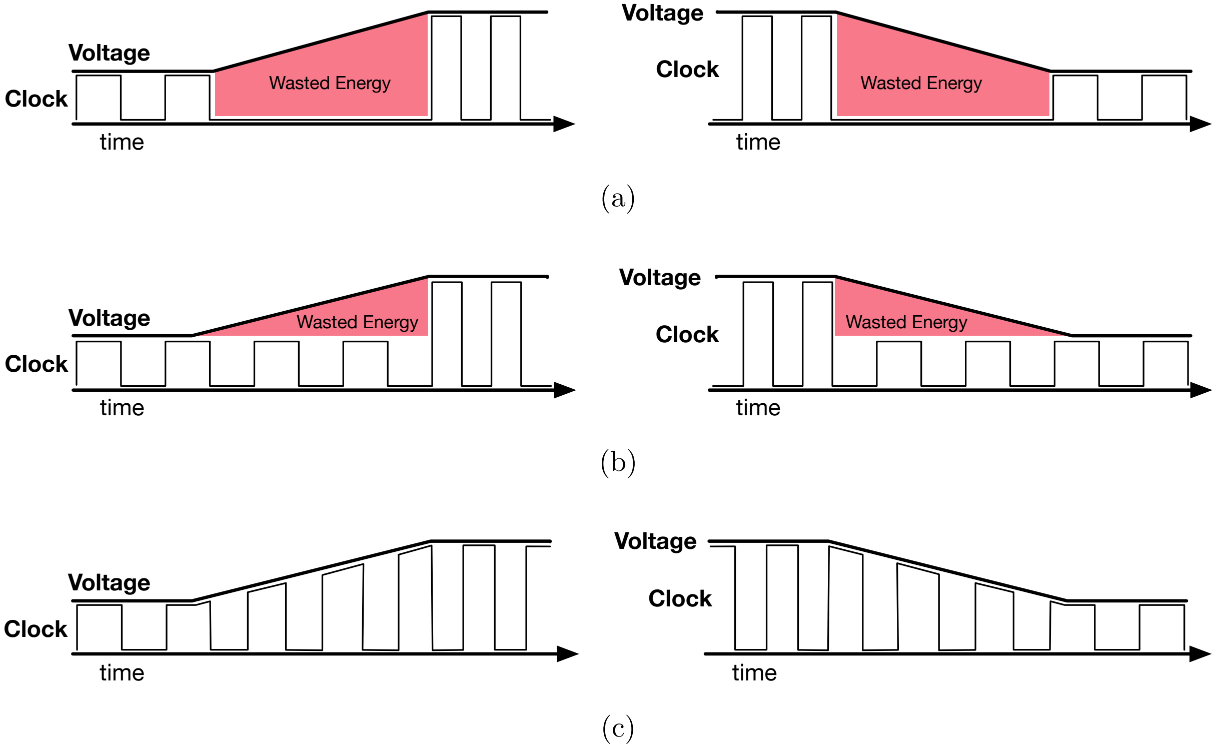
as the frequency changes, a performance issue caused by producer-consumer queues filling due to the imperfect simulation of operation at frequency f. Because of these tradeoffs, the use of voltage dithering must be carefully considered in systems with only discrete voltages available.

2.2 Clock Generation

Clock generation is another key aspect of FG-AVS design. To achieve high energy efficiency, each voltage area must be supplied with a clock running at the fastest frequency that will allow correct operation at its current voltage level. Operating at a lower frequency than is possible for a given voltage results in wasted energy. Furthermore, these clocks must be able to change frequencies when the supply voltage changes. In the case of fast voltage-mode transitions, the frequency should transition quickly as well to avoid timing issues or wasted energy at lower-frequency operation. These constraints require changes to traditional clock generation techniques for digital systems.

Most SoCs employ phase-locked loops (PLLs) that multiply a slow reference to generate an appropriately fast clock that can then be used by synchronous digital logic. PLL design is the subject of considerable research effort to improve the noise characteristics of the generated clock, increase accuracy, and reduce the impact of the circuit on the overall system. For example, IBM modified the PLL in its POWER7 microprocessor with a skitter circuit that measures the phase of the clock and gives warning when it nears a timing violation, using digital control to then adjust the clock and avoid the margin hazard [49]. Other approaches include all-digital PLLs [50–53], injection-locked multipliers [54, 55], multiplying delay-locked loops (DLLs) [56–59], and phase-picking DLLs [60]. These approaches tend to require considerable area and power. Furthermore, PLLs often require three cycles or longer to re-lock to a new frequency, which can require halting the digital circuit entirely until the voltage transition is complete. Even if the new clock frequency can be quickly generated, the PLL circuit may not be able to adjust its output frequency as the voltage changes, requiring operation at the slower frequency during the mode transition. Either of these cases wastes energy and increases voltage mode transition costs (see Figure 2.6). For these reasons, traditional PLLs are not well-suited to clocking FG-AVS systems.

Research efforts have explored alternative clock generation schemes. One option is to generate a fast global clock and then divide the clock locally as appropriate for each voltage domain. This approach requires distributing a fast, always-on clock globally, incurring power overhead. It can also only provide clocks that are ratios of the reference, which may not be well-suited for the available voltages. As an alternative, global clock generation can be elided entirely, with local circuits generating a clock in each voltage domain based on the local conditions of that domain. Eliminating global clock distribution saves area. These local circuits can generate clocks from a lookup table corresponding to the operating condition, or they can employ adaptive clocking techniques, responding to changes in the local supply voltage by changing the generated clock frequency directly. These adaptive clock circuits, commonly



(a)

(b)

(c)

Figure 2.6: Waveforms demonstrating the energy cost of different clocking schemes during rising and falling voltage transitions. In (a), the clock is halted during a voltage transition, wasting considerable energy. In (b), the clock operates at the lower frequency during the transition. In (c), an adaptive clocking scheme continuously adjusts the clock period during the voltage transition.

deployed for droop detection in large SoCs [61–64], have the advantage of mitigating noise and variability as well as responding to voltage changes. Furthermore, they can vary their output frequency continuously during a mode transition, allowing uninterrupted operation with little transition overhead as shown in Figure 2.6c. Fine-grained spatial partitioning further increases the noise resilience of these systems [65], and commercial implementations have used hybrid adaptive clock generators that respond to local voltage fluctuations while still tracking a fixed external reference [66]. Local adaptive clock generators are ideally suited to FG-AVS systems.

2.3 Clock Domain Crossings

An important challenge in the adoption of fine-grained GALS design styles that enable FG-AVS in space is the difficulty of safely transferring data between asynchronous clock domains. Fundamental to this difficulty is the danger of metastability in flip-flop state elements. Flip-flops rely on feedback inverters to store data, which settle into a stable condition when storing a 1 at one of the inverter outputs and a 0 at the other. A metastable condition also exists, however, when the stored voltage level at each output is midway between V_{DD} and ground. In this case, the feedback holds these invalid values until some noise event unpredictably perturbs the circuit towards one of the stable states.

Flip-flops are only susceptible to metastability when their input data changes near the rising clock edge. Synchronous designs guard against metastability by timing closure at design time. By ensuring that no register-to-register propagation delay is too slow, digital designers guarantee that data will stop transitioning before the setup time of the flip-flop, preventing the metastable condition. In a clock domain crossing, however, the input data signal is fully asynchronous to the clock of the flip-flop, and so no design-time guarantee can be made about the relative arrival time of these two signals. Metastability can lead to incorrect operation by resolving to an incorrect value, or causing timing failures that propagate metastability into other parts of the logic. Because of this possibility, circuits that reduce or eliminate the danger of metastability must be used at every clock-domain crossing.

The most common way to address metastability at asynchronous boundary crossings is to add several series flip-flops on a signal line that crosses clock domains and clocking these extra flip-flops with the clock of the receiving domain, as shown in Figure 2.7. The probability of metastability resolving by some time t has been empirically shown to follow the exponential relation

$$P(t) = Ke^{-t/\tau} \tag{2.1}$$

where K and τ depend on details of the technology and flip-flop implementation [67]. This synchronizer circuit delays the sampling of the signal in the receiving clock domain by several cycles, allowing time for any metastable condition to resolve. The addition of a single flip-flop to the receive path increases the tolerable time t by a full clock period and dramatically reduces the probability of metastability-induced fault due to this exponential relation. Technology-specific parameters and the number of circuit instances can be used to determine whether two series flip-flops will suffice, or whether more might be needed to meet the desired MBTF constraint. In practice, two series flip-flops are generally sufficient for test chips, while production silicon that may have thousands of synchronizers across millions of chips that must be guaranteed to work correctly for many years can require three or more stages of synchronization. No number of stages can completely eliminate the probability of a fault. Other circuits for clock domain crossings are reviewed in Chapter 5, but every technique imposes some latency penalty relative to the synchronous baseline.

An additional impact of design partitioning for fine-grained GALS design is the necessity of using backpressure at the partition interfaces. Backpressure, or the ability of a pipeline to stall the stages behind it until it is ready to accept more data, is typically implemented in synchronous systems using ready-valid interfaces. Consider a computational pipeline with several stages. In a synchronous implementation, data can step forward in the pipeline once

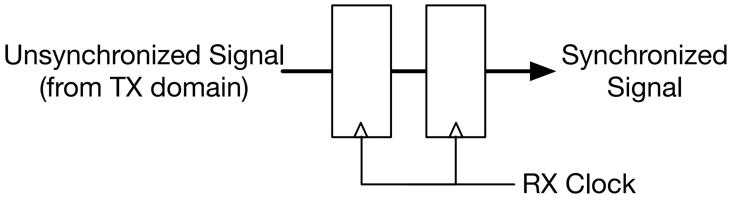


Figure 2.7: Synchronization via series flip-flops.

per cycle; a valid bit may be added so that the pipeline advances only when valid data is supplied. In an asynchronous implementation, however, each pipeline stage may operate on a different clock, and the relative frequencies of those clocks are not known. Accordingly, a stage may not have completed its computation by the time the next stage has advanced one cycle; conversely, a stage may complete its computation early, before the next stage is ready to accept the data. In a system employing FG-AVS, these relative frequencies can vary arbitrarily over time. To properly control the flow of the data through this asynchronous pipeline, each stage must be able to exert backpressure on the previous stage. A ready-valid interface at each stage ensures that data only advances both when it is valid and when the next stage is ready for it. This logic is necessary in asynchronous communication, but it can add substantial overhead compared to the simple valid-only approach.

The combination of increased interface latency and the requirement for backpressure can impose significant performance overhead as designs are rearchitected into the fine-grained GALS paradigm. This retrofit is particularly challenging for large, tightly coupled designs, such as multiple-issue out-of-order cores with many pipeline stages and frequent bypass paths between them that rely on fixed, deterministic pipeline latencies to achieve high performance. Other types of designs, such as signal processing pipelines, are less sensitive to latency and have less internal feedback, making them more amenable to such partitioning. Nonetheless, the performance degradation of these fully decoupled designs compared to their synchronous counterparts must be an important consideration in FG-AVS adoption.

2.4 Area Overhead

One of the main drawbacks of FG-AVS is the area overhead required for its implementation. Each circuit described in the previous sections requires silicon area that would not otherwise be needed in a coarse-grained implementation. Additional area overhead is imposed by circuits such as level shifters and the metal resources of a redesigned power grid. Area remains a scarce resource on SoCs; area used for FG-AVS cannot instead be used for increased parallelism, larger caches, or other microarchitectural features that could increase SoC performance. This section summarizes all of the potential sources of additional area overhead in designs implementing FG-AVS, though the actual area impact is highly dependent on design and implementation details.

2.4.1 Integrated Voltage Regulation

Integrated voltage regulators that can supply the many voltages required for FG-AVS can impose considerable area overhead. Most techniques require large passive elements to achieve high conversion efficiencies, and the power densities of these circuits are low enough that their area can often be significant. Accordingly, recent SoCs implementing integrated voltage regulators report area overheads of 16-61% over the regulated area [10, 40, 42, 43, 68]. Smaller voltage domains may also require more decoupling capacitance, either as part of the regulators or implemented separately.

2.4.2 Clock Generation

The requirement to generate many independent clocks may incur area overhead, but it can simplify design as well, making the overall impact unclear. If independent local clock generators are implemented in each voltage domain, the number of clock generation circuits can increase dramatically. However, these circuits may be simpler and smaller than the centralized PLLs they replace. Furthermore, clock tree circuitry is simplified when the need for large clock trees distributed globally throughout the chip is removed.

2.4.3 Clock Domain Crossings

The logic required for safe clock domain crossings results in additional area for several different reasons. First, the logic required to implement clock crossings, such as additional flip-flops for synchronization, would not be required in a synchronous design. The implementation of queues at the interface also requires additional area. Even if the synchronous version of the design already has a queue where the clock crossing is to occur, the additional latency of the asynchronous boundary means that a deeper queue is required to achieve full throughput at the interface, and even deeper queues might be needed to maintain dataflow with varying clock rates, especially if techniques such as voltage dithering are frequently used. Finally, digital blocks may require extensive redesign to implement backpressure where none was needed in the synchronous implementation of the block. The cost of this varies considerably depending on the design, but it can be substantial.

2.4.4 Level Shifters

The potential difference in supply voltage between neighboring, communicating voltage domains can lead to unsafe transmission of digital values from the lower-voltage domain to the higher-voltage one, as the high value driven from the low-voltage domain may not exceed the switching threshold of the receiving device in the high-voltage domain. This would result in an incorrect low value being read across the interface. To prevent this situation, level-shifter gates must be inserted on any signal that crosses between voltage domains that could experience this voltage difference during operation. An example of a level shifter is

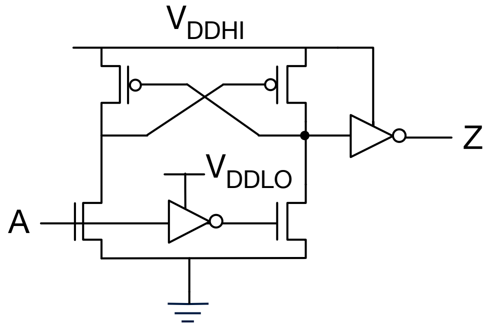


Figure 2.8: A level-shifter circuit that converts signal A in voltage domain V_{DDLO} to buffered signal Z in voltage domain V_{DDHI}.

shown in Figure 2.8. While each level shifter is small, many may be required in FG-AVS designs with many voltage domains and signal crossings, so the addition of these gates can impose substantial area overhead.

2.4.5 Power Grid

The complications inherent in routing many different supplies to the appropriate areas of the chip can significantly increase the amount of metal resources required for power distribution. If multi-rail switching is implemented, the costs of duplicative supply rails multiply the requirements directly. Even if regulators are integrated, however, additional rails are required to route the input supplies to the regulators and then route the generated supplies to the appropriate power domains. The presence of level shifters, which require multiple supplies, further complicates power grid design. These overheads use routing tracks for power distribution that could otherwise be used for clock distribution or signal routing. Depending on the routing intensity of the digital design, this can potentially impact chip area.

2.5 Design Effort

The extensive design requirements for FG-AVS impose a cost overhead related to increased design complexity that requires greater engineering effort. This overhead results from many factors that extend beyond the need to design and implement new voltage regulation, clock generation, and asynchronous boundary crossing circuits. This section notes several additional factors that make the design of FG-AVS systems more challenging.

One area of additional design effort is seen during synthesis of the digital blocks in the design. The inclusion of many voltage and clock domains in an FG-AVS system complicates clock constraints that may have otherwise been simple in a synchronous design. Care must be taken to correctly constrain each clock domain and ensure correct constraints at clock domain crossings. Voltage areas must also be correctly defined so that level shifters can be inserted between them. These additional constraints increase the effort needed to correctly synthesize the design.

Physical design also becomes more complicated in an FG-AVS design. In a synchronous design, there is considerable flexibility in floorplanning the design to satisfy timing and other constraints. In an FG-AVS design, on the other hand, the area available for the logic and macros inside each voltage area is much smaller, and so the placement of these blocks is highly constrained. Irregular macros can become difficult to place because oblong voltage areas may have poor power connectivity and suffer from IR drop or other issues. The relative placement of each voltage area to its neighbors must be carefully considered, both so that communication paths are kept short and so that the power grid remains robust. Additionally, the presence of many integrated voltage regulators and local clock generators requires much more effort to realize their legal physical placement and correct integration into the design. All of these constraints require considerable engineering effort.

Another aspect of the increased design effort is system verification. Because each voltage domain can operate in several different modes, independent of other domains on the chip, the number of potential operating conditions grows exponentially as the number of voltage domains increases. This results in dramatically increased effort to ensure power grid reliability and thermal safety as well as adequate performance. Simulating power consumption becomes much more complicated when the power state of each voltage area is not static, but instead changing frequently over time. Furthermore, verifying the correct behavior of asynchronous boundary crossings is a difficult and error-prone endeavor, and requires entirely new methodologies beyond those used for standard synchronous RTL verification. This added verification effort to ensure correct operation may be the most challenging aspect of FG-AVS.

Chapter 3

Integrated Voltage Regulation

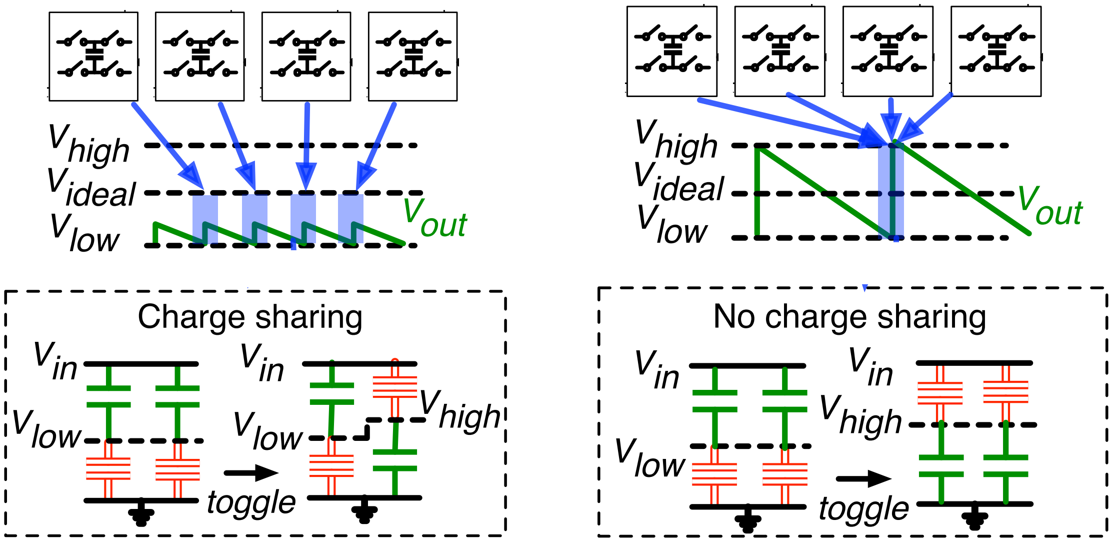
Integrated voltage regulation is a key enabling technology for FG-AVS. This chapter describes an integrated switched-capacitor voltage regulator design appropriate for integration in an FG-AVS system. To achieve high conversion efficiencies, the integrated regulator must be paired with a responsive adaptive clock generator circuit, which is also well-suited for the fast clock frequency transitions required for FG-AVS. These circuits impose constraints on the design of the FG-AVS system; these system implications are discussed in detail, and the tradeoffs of this approach to voltage generation and clock distribution analyzed.

3.1 Simultaneous-Switching Switched-Capacitor Voltage Regulation

Switched-capacitor voltage regulators are a favorable choice for integrated voltage regulation because high-quality integrated capacitors are much easier to implement than inductors, while switching regulator efficiencies exceed those of linear regulators. However, as described in Section 2.1.1, switched-capacitor regulators typically achieve conversion efficiencies that are not high enough to justify their overhead. Energy savings from AVS are reduced by conversion losses to the extent that the additional area and design complexity required for their implementation were not worthwhile. Accordingly, designing a switched-capacitor regulator with reasonably high efficiency is a critical component of a realistic FG-AVS system.

3.1.1 Traditional Switched-Capacitor Regulation

Because each switching event in a switched-capacitor circuit is perturbing its output voltage with a discontinuous addition of charge, the output of switched-capacitor regulators tends to ripple. Traditional switched-capacitor designs employ a technique known as interleaving to suppress this voltage ripple and produce a relatively flat output supply that is well-suited for synchronous digital logic [40, 41, 69–72]. By dividing the total flying capacitance in the system into many smaller unit cells, and switching each of these unit cells out of phase



(a)

(b)

Figure 3.1: Interleaved and simultaneous-switching switched-capacitor DC-DC converters [10]. In the four-phase interleaved converter shown in (a), each unit cell switches out of phase with the others, resulting in a relatively flat output voltage but incurring charge-sharing losses. In the simultaneous-switching converter shown in (b), all unit cells switch at once, eliminating charge-sharing losses but resulting in a large output voltage ripple.

with the others, the relative amount of charge delivered onto the supply with each switching event is small, and the output voltage ripple is minimized as shown in Figure 3.1a. High interleaving phase counts of 16, 32, or even greater can be used to suppress the ripple and produce a steady output voltage [39, 73, 74]. However, this interleaved approach suffers from charge-sharing losses as each unit cell shares charge across the flying capacitance of the others when it switches. These intrinsic charge-sharing losses comprise up to 40% of the overall energy losses in the voltage conversion [75].

The conversion losses of switched-capacitor voltage regulators are composed of four parts, three of which depend on the switching frequency of the design [39]. The intrinsic charge-sharing loss P_{cfly} of switched-capacitor designs is inversely proportional to the switching frequency of the design, because a slower switching frequency means more charge being transferred for each switching event. The bottom-plate conduction loss P_{bottom} caused by the parasitic capacitance of the flying capacitor to ground and the parasitic gate capacitance P_{gate} of the switches are both directly proportional to the switching frequency. The conduction loss P_{cond} through the switches does not depend on switching frequency. Switched-capacitor designs typically operate at the optimal switching frequency to minimize the total losses in the system, operating fast enough to reduce charge-sharing losses but not so fast that the parasitic losses dominate.

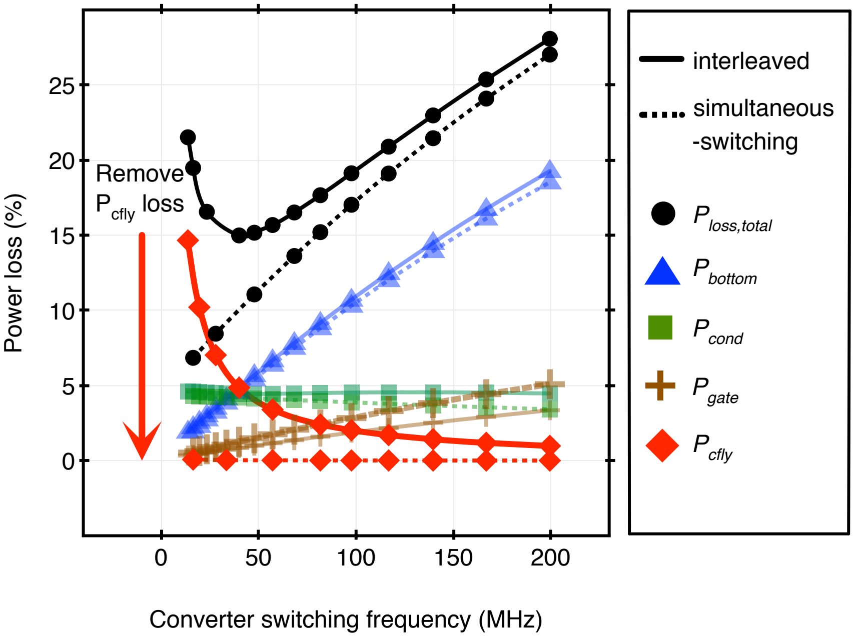


Figure 3.2: Analytical results showing the efficiency improvement of the simultaneous-switching switched-capacitor voltage regulator [10] (© 2016 IEEE).

3.1.2 Simultaneous-Switching Switched-Capacitor Regulation

An alternative approach called simultaneous switching is shown in Figure 3.1b. Instead of interleaving the unit cells, the entire flying capacitance of the regulator is switched at once, eliminating all charge-sharing between different unit cells present in the interleaved approach [10]. The elimination of this loss term also means that the only losses remaining in the system are either directly proportional to switching frequency or agnostic to it, so the switching frequency of the system can be reduced, further shrinking remaining losses. Figure 3.2 shows analytical results comparing the losses from an interleaved system with those of a simultaneous-switching regulator driving an ideal resistive load. The simultaneous-switching switched-capacitor DC-DC (SS-SC) converter is able to achieve total losses of less than 10% by slowing its switching frequency. In real systems, the load is not purely resistive, but instead has some capacitive component, so charge-sharing losses are not eliminated entirely. Figure 3.3 shows a simulated example of how the inclusion of a capacitive load in the SS-SC circuit diminishes some of the advantage of the simultaneous-switching design and increases the most efficient switching frequency relative to a purely resistive load. Nonetheless, the energy savings over the interleaved approach remain substantial, and the optimal switching frequency is lower in the SS-SC design.

Simultaneous-switching designs reduce the loss components associated with switched-capacitor regulation, but they generate an output voltage that ripples substantially around an average. This rippling supply is not well-suited to a traditional synchronous digital

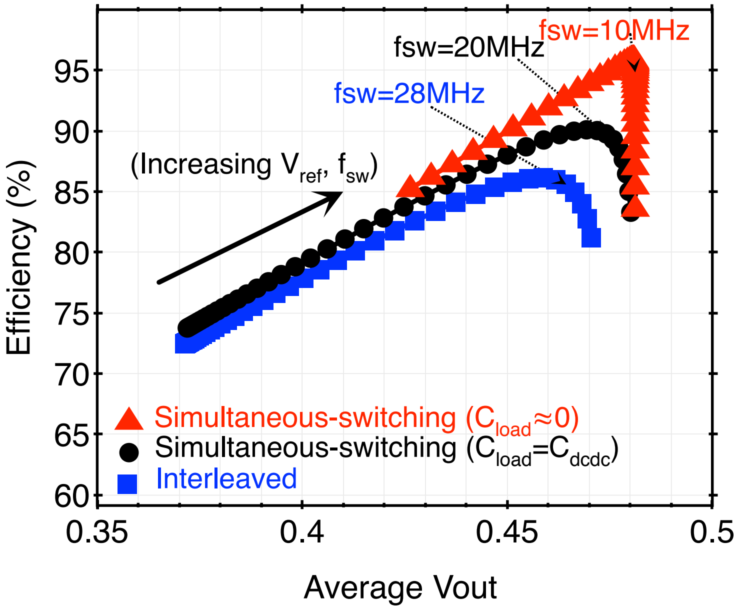
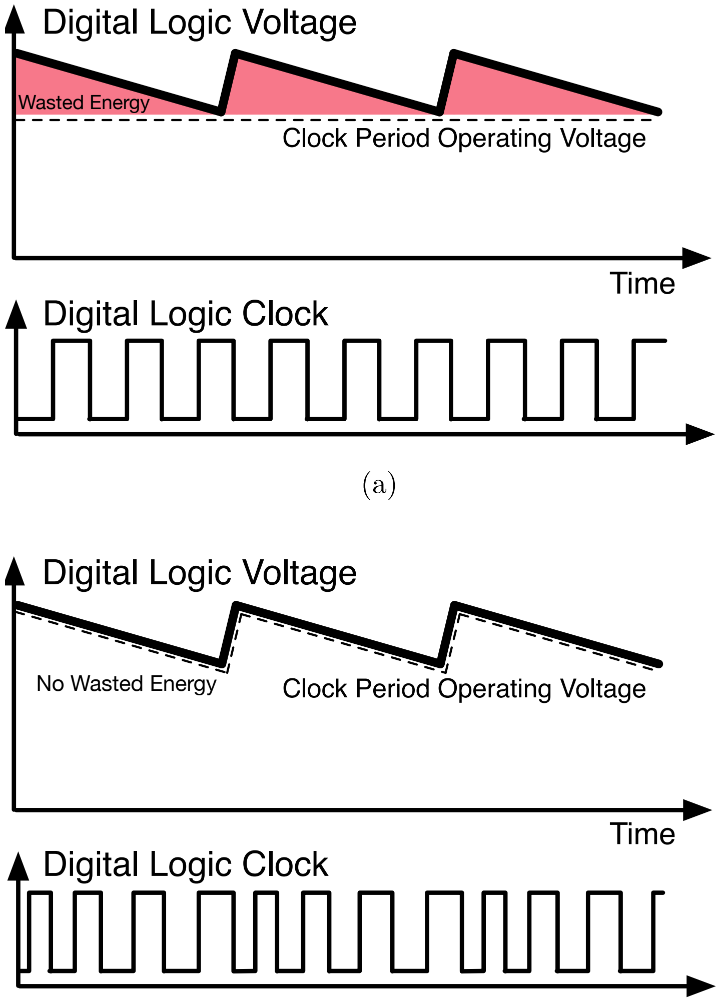


Figure 3.3: Simulation results showing the impact of load capacitance on converter efficiency and switching frequency [10] (© 2016 IEEE).

system operating at a fixed clock frequency, because the digital clock must operate at a slower frequency corresponding to the lowest voltage of the ripple in order to guarantee safe operation. This would result in wasted energy as the rippling voltage exceeds its minimum, as shown in Figure 3.4a, and these over-voltage losses would exceed the savings from the elimination of charge sharing. Systems employing SS-SC regulators therefore require the clock supplied to the digital load to adapt to the changing voltage in order to achieve reasonable efficiencies [75]. As shown in Figure 3.4b, by changing the clock on a cycle-by-cycle basis to match the instantaneous operating conditions of the load, the over-voltage losses can be eliminated.

3.1.3 Regulator Design

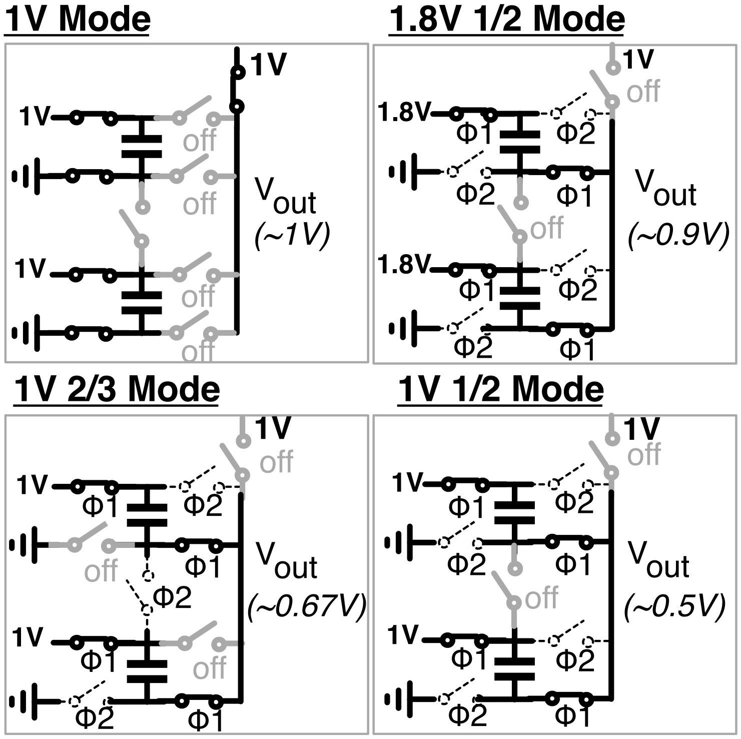
Traditional switched-capacitor regulators achieve peak efficiencies when supplying output voltages at discrete levels determined by their topology. While SS-SC regulators do not generate a fixed output voltage, their efficiency still peaks at a particular average output level. As a wide range of voltages are desirable for FG-AVS, a reconfigurable switching network with a topology similar to that of [39] was implemented. The design can operate in four different modes that generate voltages from fixed 1 V and 1.8 V supplies. The 1.8 V 1/2 mode downconverts the 1.8 V input in a 2:1 ratio, resulting in an average output voltage around 900 mV. The 1 V 2/3 mode and 1 V 1/2 mode each downconvert the 1.0V input, resulting in average output voltages of roughly 667 mV and 500 mV respectively. A bypass mode connects the 1 V input directly to the output rail. The design re-uses the flying



(a)

(b)

Figure 3.4: Sample voltage and clock waveforms of SS-SC systems. In (a), the clock does not adapt to the voltage ripple, so it must be margined for the lowest supply voltage, resulting in wasted energy at higher voltages. In (b), the clock adapts to the voltage ripple, allowing the circuit to speed up when the voltage is higher than the minimum.


1V Mode

1.8V 1/2 Mode

1V 2/3 Mode

1V 1/2 Mode

Figure 3.5: The four topologies of the reconfigurable SS-SC converter [10] (© 2016 IEEE).

capacitance and switches, reconfiguring the switching pattern between each topology to avoid area overhead (see Figure 3.5).

The phases and configurations of the switches in the SS-SC regulator are set by a finite-state-machine controller. This logic is responsible for configuring and reconfiguring the topology of the regulator, as well as switching between the two operating phases to pump charge onto the output of the converter. The regulator topology is determined by setting a control register that, when changed, triggers a reconfiguration between voltage modes. To generate the toggle clock that is distributed to the switches in the regulator, a comparator circuit acting on a 2 GHz clock detects when the regulated output voltage drops below a fixed external reference. When the comparator triggers, control logic produces the next toggle clock edge, switching the SS-SC toggle clock phase and causing more charge to be supplied, boosting the voltage. As a different reference voltage is needed for each mode, different comparators must be implemented to function in the appropriate voltage ranges (see Figure 3.6). In the event that one switching event does not sufficiently increase the generated voltage to bring it above the reference voltage, additional logic triggers further switching events after a delay, ensuring that the voltage will eventually be boosted back up to nominal levels. The state machine controller is diagrammed in Figure 3.7.

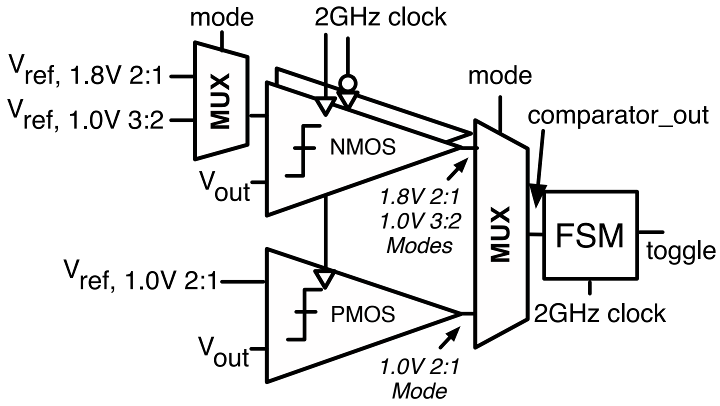


Figure 3.6: A diagram of the comparators used in the SS-SC controller [10] (© 2016 IEEE). Different comparators are needed to accommodate the voltage ranges of the references for the three different switching modes.

3.2 Adaptive Clocking

Section 2.2 described the utility of distributed, localized clock generation in the implementation of the GALS design style required for FG-AVS systems. Furthermore, because SS-SC converters generate a rippling supply voltage, only a clock that adjusts to this ripple on a cycle-by-cycle basis to supply the digital logic can avoid wasted energy in these systems. Accordingly, a local adaptive clock generator was implemented that can be paired with the SS-SC converter for FG-AVS systems [68].

A block diagram of the clock generator circuit is shown in Figure 3.8. The adaptive clock generator is free-running and does not require any external reference. The circuit generates clock edges via a D flip-flop, which is set or reset with pulse generators to toggle between one and zero. After a rising clock edge is generated, the edge propagates through the first delay unit and then triggers a reset pulse on the flip-flop, causing the output to fall. This edge then continues through the second delay unit and triggers a clock pulse on the flip-flop, resulting in a rising edge at the output. This design allows the duty cycle of the generated clock to be adjusted.

The adaptive clock generator is supplied by the same rippling output voltage that is used to power the digital logic in the system. The delay units therefore respond to voltage changes similarly to the digital logic; the propagation delays that trigger each clock edge face the same environment as the digital logic paths in the design. Each delay unit is comprised of four delay banks, and each bank consists of a chain of a particular cell in addition to multiplexors and control logic that allow the number of cells in the delay chain to be selected. The first bank uses a custom buffer cell design to balance rise and fall times. Each of the remaining banks consists of a standard cell that was commonly observed in the synthesized critical paths of the digital logic. Each bank uses a different combination

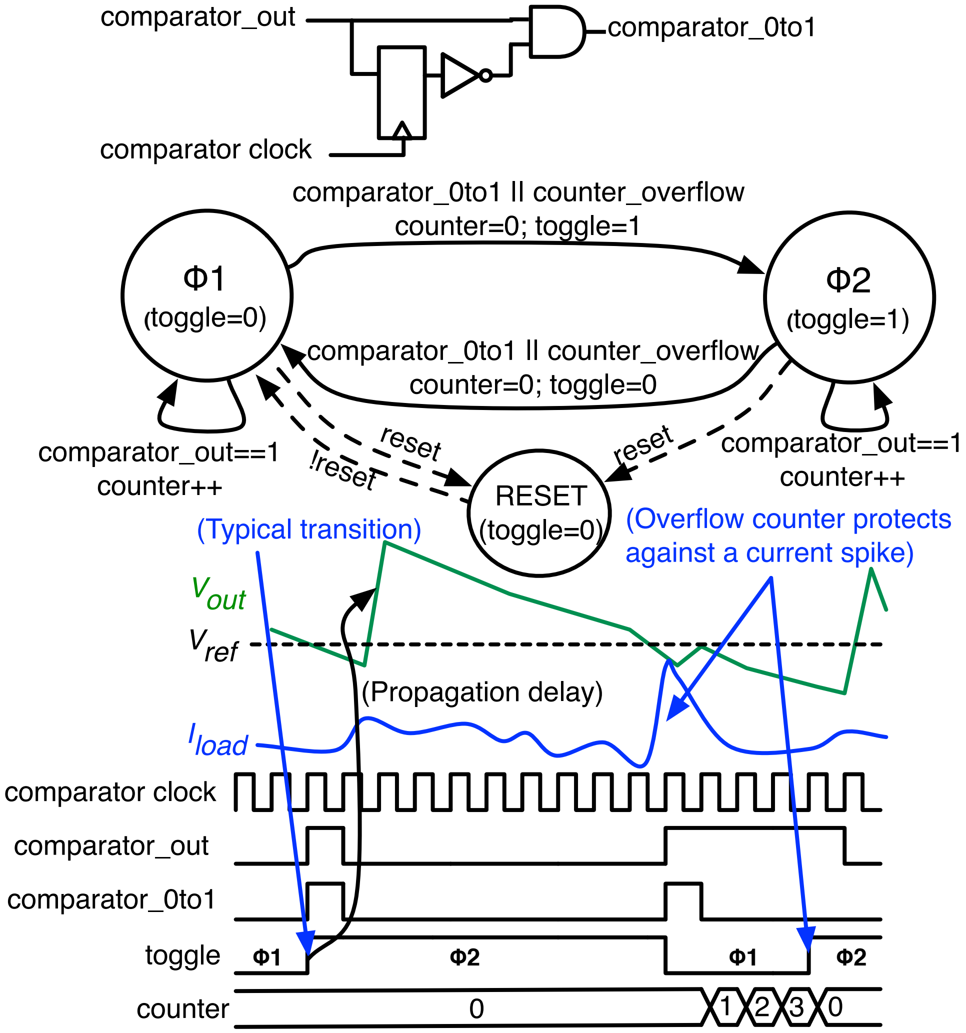


Figure 3.7: The state machine used in the SS-SC controller and example waveforms of its operation [10] (© 2016 IEEE). If the current demand increases as the converter is toggling, the voltage may not rise about V_{ref}, so an additional counter triggers further switching until the voltage increases.

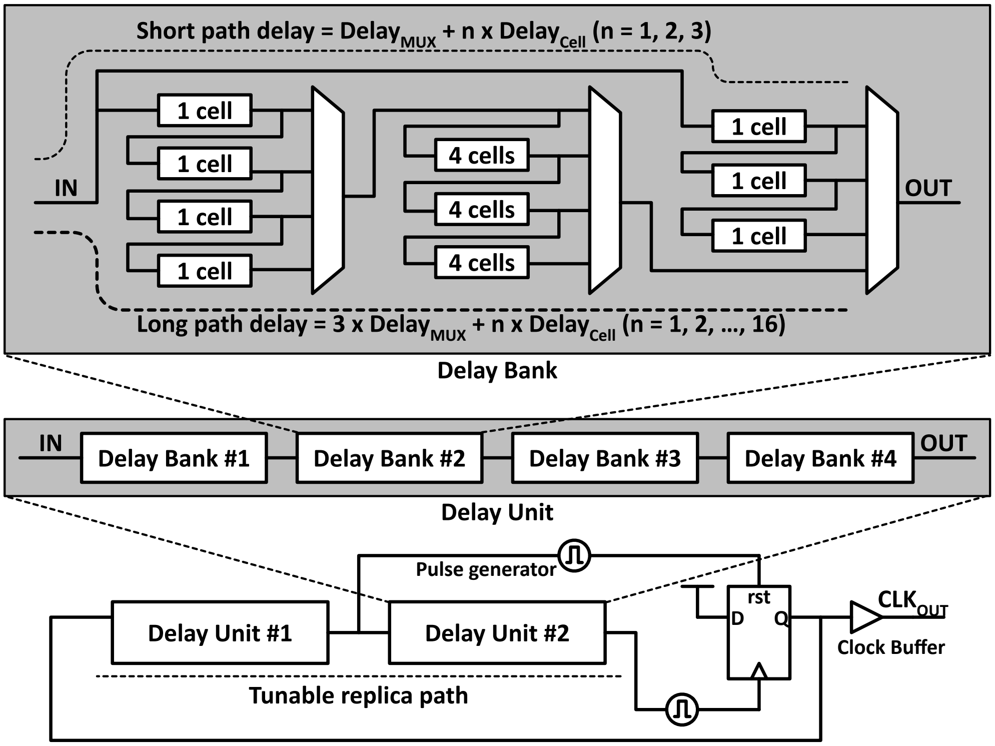


Figure 3.8: A block diagram of the adaptive clock generator [68] (© 2017 IEEE).

of pMOS/nMOS ratio and gate length, as these characteristics affect the voltage-frequency relationship of the cells. By selecting different mux settings, the delay paths can be tuned until they match the delay characteristics of the critical paths in the digital logic supplied by the generated clock, ensuring that the clock edges are generated at the appropriate frequency for the instantaneous voltage produced by the regulator.

3.3 System Implications

The design and implementation choices of SS-SC voltage regulation and local adaptive clock generation constrain the design space of the FG-AVS system in which these circuits are implemented. This section explores the implications of these circuits on clock infrastructure, operating constraints, and physical design.

3.3.1 Clocking

Both the design of the SS-SC regulator and the adaptive clock generator have implications for the design of the various clocks that must be distributed in the design. In the case of the integrated voltage regulator, several fixed-frequency clocks must be provided to each independent regulator in the system, to supply the finite state machine controller and lower-bound comparator. In addition, the generated toggle clock that is used to alternate between operating phases must be distributed to the switches in the design. To keep the size of the individual switches and capacitors reasonable, and to ease physical design, it is common to partition the entire flying capacitance into several unit cells, even in the simultaneous-switching implementation. In order to ensure that all of these unit cells do indeed switch simultaneously, eliminating charge-sharing losses between them, the toggle clock must be distributed to every unit cell in a balanced clock tree that matches delays throughout the design. This is well within the capabilities of standard place-and-route tools, but clock constraints and physical placement may need to be adjusted to achieve an appropriately balanced tree.

The adaptive clock generator design also has implications for the full system. Most importantly, the cycle-by-cycle adjustment of each clock edge to track the voltage ripple means that clock domains must be fully asynchronous with one another; there can be no deterministic frequency or phase relationship between neighboring domains, even if they are operating at the same nominal voltage. This eliminates some options for low-latency clock domain crossings (see Chapter 5), and requires that synchronization circuitry between clock domains be able to tolerate fully asynchronous arrival times. The tunable delay paths also impose requirements for system bringup and testing, as the mux settings must be adjusted on a per-core and per-domain basis to achieve the best tracking of the local critical paths in the presence of random and spatially varying process variations. This extensive tuning regime may require extensive one-time or boot-time calibration or even the addition of dedicated built-in self-test (BIST) circuitry to be practical in mass-production scenarios.

3.3.2 Insertion Delay

Adaptive clock generation depends on the ability of the clock generator to accurately sense the conditions of the logical clock sinks to which its generated clock is distributed. Any mismatch in process, voltage, or temperature (PVT) between the actual critical path and the replica circuit can exacerbate misalignment of the voltage-frequency curves between these two paths. This mismatch requires additional margin in the generated clock edges to guarantee correct operation, slowing the clock and increasing the energy consumption of the clock domain due to its slower execution time. Unfortunately, even if the replica path is perfectly matched to the critical path of the circuit, the presence of clock insertion delay causes additional timing issues.

Insertion delay is the time required for clock edges launched at the generating source to arrive at the register sinks after propagating through the clock tree. The delay imposes a mismatch in time between when the adaptive clock generator senses the operating condition to produce a clock edge and when that clock edge is actually used by the digital logic. If this mismatch in time does not result in a mismatch in operating condition, then the insertion delay does not pose any issue. However, the presence of the rippling voltage supply implies that the voltage will be different between these two instants, causing a mismatch that must ultimately be addressed with additional margin. In fixed-clock circuits, the presence of insertion delay actually reduces the impact of voltage changes because the propagation delay through the clock tree changes the clock duration, an implicit adaptation known as the clock-data compensation effect [76–78]. In adaptive clock circuits, on the other hand, the different voltage-delay relationship of the clock tree compared to design logic causes harmful timing effects that are exacerbated as insertion delay increases.

The deleterious effects of insertion delay on the timing constraints of an adaptive clocking system under voltage changes are illustrated in Figure 3.9. The diagram is labeled with units of arbitrary delay. In this example, the replica paths in the adaptive clock generator perfectly track those in the critical path over the voltage range of interest, but the clock insertion delay varies less with changes in voltage than either of these paths because the clock tree tends to be wire-dominated relative to logic paths in the design. (The relative voltage-delay differences between the clock tree and the logic paths are exaggerated in the example.)

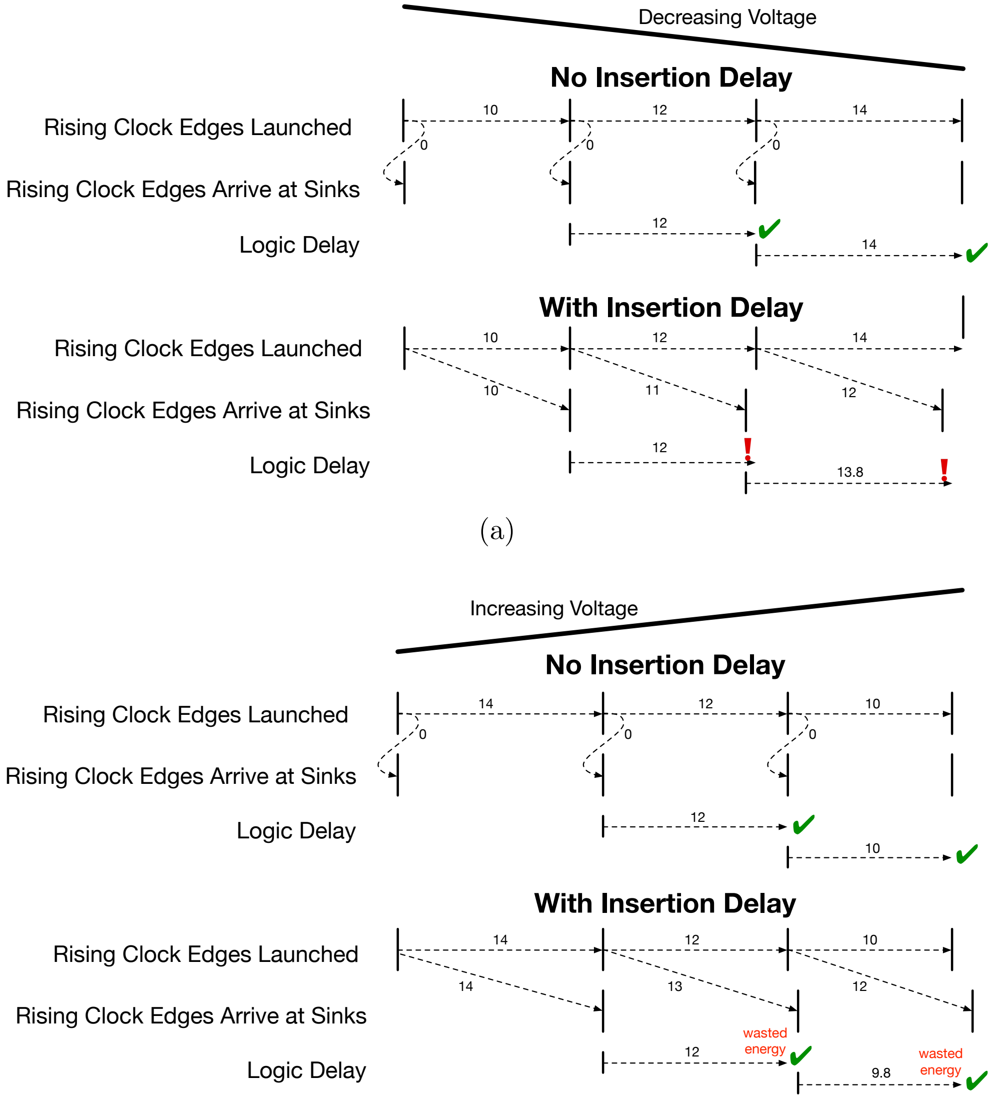
Figure 3.9a shows the impact of a decreasing supply voltage on the setup time constraint of the critical path. This represents the common case, as the SS-SC supply is slowly decreasing for most of its operation. In the ideal case where no insertion delay is present, the replica path can match the critical path exactly. In the presence of insertion delay, however, the lesser delay sensitivity of the clock tree speeds up the arrival of the clock edges relative to their delayed generation, causing setup time violations. The replica timing paths in the adaptive clock generator would have to be slowed to guard against this failure, adding margin that reduces system efficiency.

Figure 3.9b shows the impact of an increasing supply voltage on the timing of the system. This type of sharp voltage increase occurs at each phase transition of the SS-SC regulator. Here, the reduced voltage sensitivity of the clock tree unnecessarily delays the clock edges. This results in wasted energy as the logic operates faster than necessary.

Insertion delay has an adverse effect on the energy efficiency of the SS-SC system under adaptive clocking. Aside from the time-consuming design of custom clock trees, the only way to ensure small insertion delays is to design voltage areas to be relatively small and compact. This both reduces the number of clock sinks and so the dimension of the clock tree, but also decreases the propagation time through long clock wires. Insertion delay is an important limitation on adaptive clocking, constraining its use as appropriate for FG-AVS but poorly suited to implementations with coarse spatial granularity.

3.3.3 Minimum Operating Voltage

As discussed in Section 2.1, the ability of switched-capacitor converters to efficiently generate only a few discrete output voltages prevents continuous voltage scaling, although the consequences of this are mitigated by the ability to dither between discrete modes. However, SS-SC regulators suffer from an additional drawback resulting from their rippling supply. Digital logic operates with greater energy efficiency at lower voltages, but logic and SRAMs are constrained in their operating range by some minimum operating voltage V_{min}, below which the circuits no longer function correctly. SS-SC converters are not well-suited to


Decreasing Voltage
No Insertion Delay
Rising Clock Edges Launched
10
12
14
0
0
0
Rising Clock Edges Arrive at Sinks
Logic Delay
12
14
With Insertion Delay
Rising Clock Edges Launched
10
12
14
10
11
12
Rising Clock Edges Arrive at Sinks
Logic Delay
12
13.8
Increasing Voltage
No Insertion Delay
Rising Clock Edges Launched
14
12
10
0
0
0
Rising Clock Edges Arrive at Sinks
Logic Delay
12
10
With Insertion Delay
Rising Clock Edges Launched
14
12
10
14
13
12
Rising Clock Edges Arrive at Sinks
wasted energy
Logic Delay
12
wasted energy
9.8

(a)

(b)

Figure 3.9: Timing diagrams demonstrating the effects of clock insertion delay under a changing supply voltage and adaptive clock. In this example, the adaptive clock generator is assumed to perfectly track the voltage-delay characteristics of the digital logic. The numbers are arbitrary, representative delays. In each case, in the absence of insertion delay results in logic arrival times that match the arrival of the next clock edge, while the presence of insertion delay causes a mismatch. In (a), the voltage is decreasing, so insertion delay causes setup time violations because the clock edges propagate through the clock tree more quickly than the slowed logic propagation times. These violations require additional clock margin to eliminate, increasing energy cost. In (b), the voltage is increasing, so insertion delay causes the logic propagation to complete early, meeting timing but resulting in wasted energy.

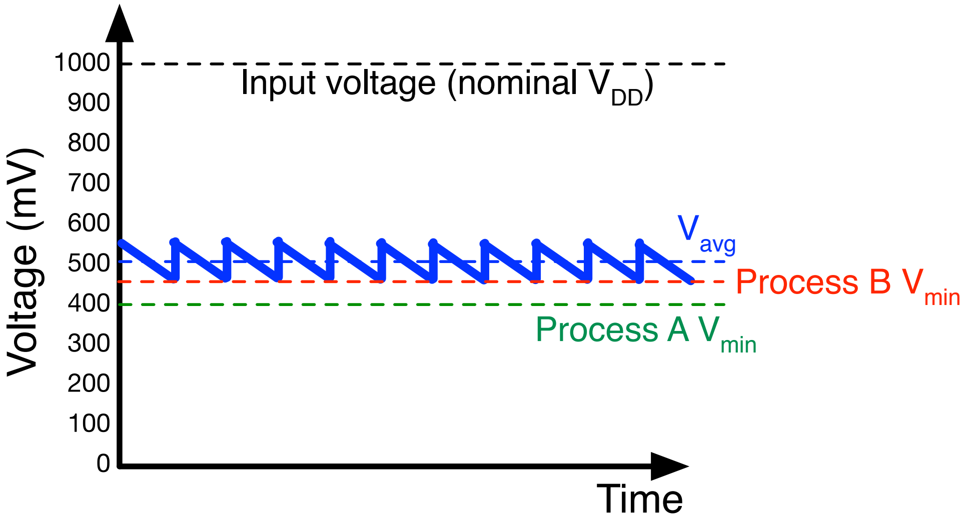


Figure 3.10: A sample SS-SC waveform showing the impact on achievable V_{min}. The SS-SC regulator operates in a 2:1 step-down mode from the fixed 1 V supply; it has a 100 mV ripple around the average voltage V_{avg} of 500 mV. In Process A, $V_{min} = 400$ mV, but the SS-SC cannot achieve this output voltage because it is limited to simple ratios of the input for high-efficiency conversion. In Process B, $V_{min} = 450$ mV. The SS-SC regulator is operating at the lowest possible voltage without violating V_{min}, but its average voltage remains 50 mV higher than a non-rippling regulating technique.

operating at V_{min} for two reasons. First, it is unlikely that the particular voltages corresponding to simple ratios of the input supplies correspond precisely to V_{min}. As ratios with denominators greater than two or three require increasingly complex switching topologies that decrease conversion efficiency, the lowest operating ratio will likely be above V_{min} by some unneeded margin. Even if this is not the case, however, the rippling voltage prevents continuous operation at V_{min} because the entire ripple cannot drop below this lower bound. These effects are illustrated in Figure 3.10. A rippling supply generated with a simple down-conversion ratio cannot achieve as low an average operating voltage as a continuous fixed supply. If operation at V_{min} is desired for improved efficiency, this discrepancy can result in a comparative loss of energy efficiency. Note that this effect would also be seen at the highest, most performant operating voltage V_{max}, but the SS-SC regulator design presented in the previous section operates in a non-rippling bypass mode at its highest voltage.

3.3.4 Physical Design

The use of SS-SC converters and adaptive clock generators imposes several constraints on the physical design of the system. The unit cells of the voltage regulator should be placed near the supplied domain to minimize the resistance between the regulators and the digital

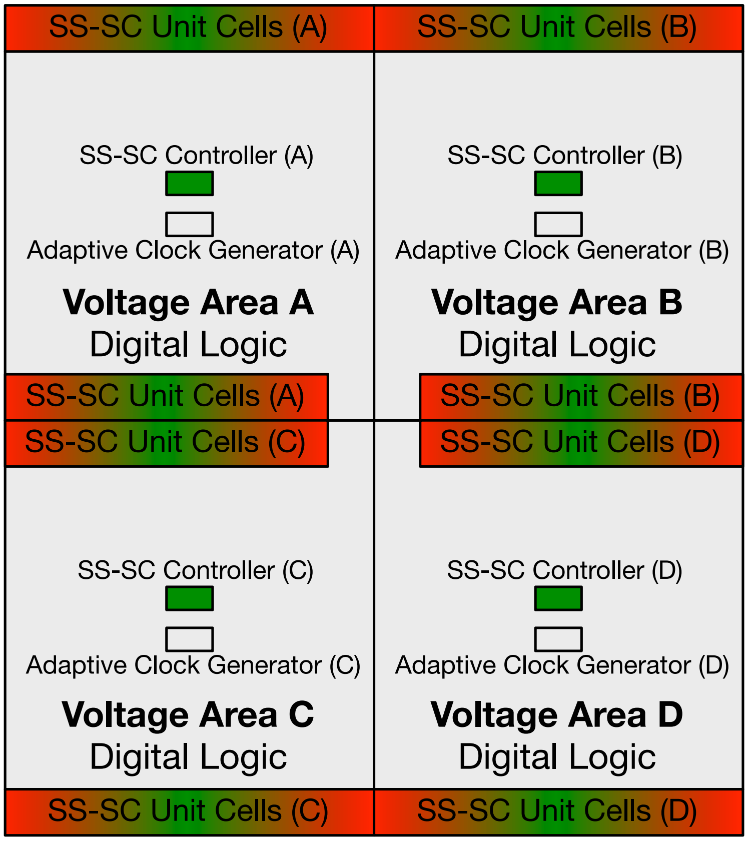


Figure 3.11: An example floorplan showing four independent voltage areas supplied by SS-SC converters and adaptive clock generators. Each generated output voltage need only be distributed to the local voltage area, but blocks shaded green require a fixed 1V supply and blocks shaded red require a fixed 1.8V supply. The SS-SC unit cells surround the voltage area they supply to minimize IR drop, and the SS-SC controller and adaptive clock generator are placed centrally to each voltage area to help meet clock constraints. The central cutout allows routing of digital connections between the domains.

logic they supply. For the most robust power supply, the unit cells may need to surround the voltage domain or even be interspersed with the logic inside it. The controller for the SS-SC regulators should ideally be centrally located among the unit cells so as to most easily balance skew from the generated toggle clock (see Figure 3.11). These layout goals can pose challenges for the design of the power grid, which must provide the input supplies to the regulators, deliver the generated voltage, and supply both fixed and generated supplies to the controller. The input supplies to the SS-SC converters must be robust with minimal inductance. This may require special consideration in package design as well as integrated decoupling capacitance.

For the most accurate tracking of the critical paths in the digital logic, the adaptive clock generator should be placed within the voltage domain. To reduce insertion delay, the generator circuit should be placed as centrally as possible. To improve the accuracy of the replica paths, the generator must be located near the actual critical paths of the logic so that it senses similar PVT conditions as shown in Figure 3.11. Alternatively, the replica timing circuits can be distributed to the critical parts of the core, with the clock generator circuit located centrally. This can improve tracking, but has the undesirable side effect of adding additional wire delays to the clock generation as the signals are aggregated in the central generator.

Chapter 4

Integrated Power Management

The execution of FG-AVS requires active power management to adjust the voltage and frequency of each domain to track its instantaneous workload requirement. Power-management software and hardware can take advantage of voltage generation, clocking, and synchronization schemes designed for FG-AVS to save energy. This chapter explores the design space of integrated power-management strategies that can enable FG-AVS.

The domain of power management encompasses a wide range of techniques and goals. Platform power management may involve controlling dozens of different components, including memory, discrete graphics cards, disks, and cooling fans. Within a single die, power-management schemes can be responsible for maintaining power and thermal limits, compensating for variation, and mitigating aging. This chapter restricts consideration of power management to schemes that attempt to save energy by modulating the voltage and frequency of a single SoC. These power-management implementations generally follow the feedback loop shown in Figure 4.1. One or more sensors provide information about the state of the digital logic of interest on the SoC. The data from these sensors is fed into a power-management unit (PMU), a controller that implements some algorithm intended to use this state information to actuate changes in operating mode and save energy. When this algorithm determines that a change in operating condition is beneficial, the PMU updates voltage and frequency settings accordingly. This chapter considers each part of this feedback loop in turn to survey the design space for FG-AVS power management.

4.1 Estimating Workload

FG-AVS saves energy only if voltage and frequency can be decreased when performance requirements are reduced. For many blocks, determining overall activity to make an informed estimate of instantaneous workload allows a useful approximation of workload demand. If a block has low activity, this is likely because it has little work to do, and so its performance requirements can be relaxed. Conversely, a highly active block should maintain its performance to avoid slowing the whole system. Information about the microarchitectural state

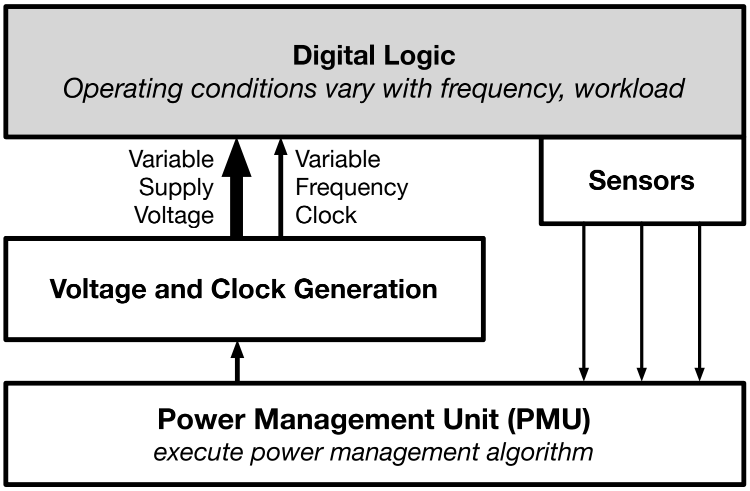

Digital Logic
Operating conditions vary with frequency, workload
Variable Supply Voltage
Variable Frequency Clock
Sensors
Voltage and Clock Generation
Power Management Unit (PMU)
execute power management algorithm

Figure 4.1: A typical feedback loop for SoC power management. Only a single varying voltage domain is shown for simplicity.

of particular blocks can also be used to speculatively inform power-management algorithms, allowing predictions to be made about future performance needs. All of this information is critical for implementing the FG-AVS feedback loop that allows energy savings.

This section describes two methods for estimating workload, power measurement and counter instrumentation, and discusses their tradeoffs and appropriate use in FG-AVS systems. In addition to standard considerations of area, power, and design overhead, these circuits must be evaluated based on the speed at which they can provide state information to the FG-AVS controller. Fast sensing is critical to enabling FG-AVS in time because more energy is saved if changes in workload demand can be detected and responded to quickly.

4.1.1 Measuring Power

SoC power measurement is commonly required because thermal constraints impose an absolute power budget that must be maintained for the entire SoC. Measuring the power of an individual voltage domain can also provide useful information about the workload of that domain: a higher instantaneous power consumption corresponds to a greater workload at that point in time. Power measurement can therefore be used as feedback in power-management algorithms to save energy with FG-AVS.

One straightforward way to measure the power consumption of a power supply is to insert a sense resistor in series with the supply and measure the voltage drop across the resistor. This approach can achieve high accuracy, but it has several drawbacks that make it poorly suited for use in FG-AVS workload estimation. Some earlier systems implement the sense resistors separately from the chip to ensure precise resistance values [79, 80]. However, these added off-chip components increase system cost and complexity, and their current profile is filtered by the parasitics of the package and board, precluding fast measurement.

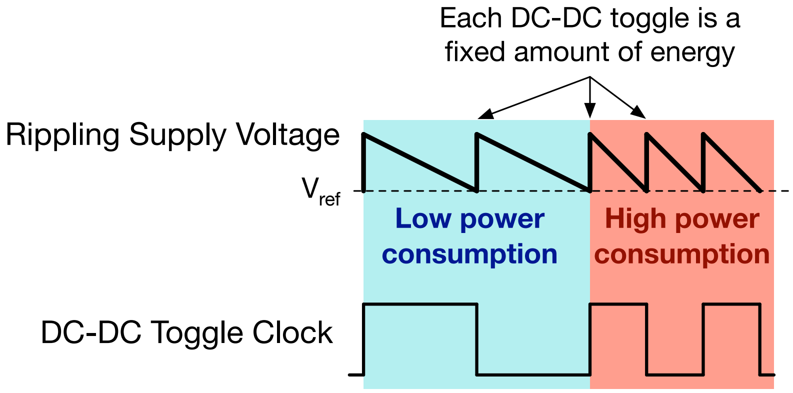


Figure 4.2: The basis for measuring power consumption in a system with SS-SC converters and a rippling supply voltage [68] (© 2017 IEEE).

Furthermore, these designs can only measure the power of voltages supplied from off-chip, so they cannot measure the power of small voltage domains supplied by on-chip regulators. For these reasons, this technique is no longer favored in commercial SoCs. The sense resistors can instead be integrated on-die per voltage domain [81], but large resistive areas may be required to achieve the necessary precision. Implementing analog-to-digital converters to measure the sensed voltage is also expensive in area, power, and design effort. These intrusive techniques are poorly suited to the needs of FG-AVS systems.

An alternative, non-invasive power measurement technique can be implemented in systems with SS-SC converters as described in Section 3.1.2. These regulators switch phases when the generated voltage drops below a fixed reference. The total switching capacitance of the system is also fixed, and so the amount of energy supplied at each switching event is a constant that can be used to estimate instantanteous power consumption of the supplied domain. The basis for this measurement technique is illustrated in Figure 4.2. When the power consumption of the supplied domain is low, the supplied voltage takes longer to drop to the lower bound, and so the switching frequency is reduced. High power consumption increases the steepness of the output voltage drop and increases the switching frequency. The SS-SC regulator toggle clock that triggers these switching events is already generated by the controller for use by the DC-DC unit cells. The instrumentation of this toggle clock with a simple digital counter can therefore provide a lightweight, non-invasive estimation of power consumption by the supplied domain [82]. This technique has little overhead, and can determine a core power measurement by counting just a handful of SS-SC clock toggles, allowing fast sensing of core activity. Unlike prior regulator-based measurement schemes, this measurement require no off-chip components [83] and the output voltage is not perturbed [84], making it ideally suited for FG-AVS power management.

4.1.2 Microarchitectural Counters

Power measurements provide a useful proxy for estimating workload, but other techniques can also give insight into this information for use in power management. The totality of information about the activity of a digital block consists of cycle-by-cycle records of whether every node in the system has toggled. In practice, counting activity on a small subset of total nodes can provide a very accurate estimate of the activity of an entire block. Unlike direct measurements of power consumption, counters can be updated and sampled every cycle, and the latency in reading the information is limited only by the time needed to synchronize the data and transport it to the PMU. Counters are lightweight and simple to implement in digital logic. Furthermore, a diversity of counters can provide detailed information about activity, unlike a single power measurement, potentially allowing better prediction of future changes in workload demand. For these reasons, many implementations estimate power using a model derived from instrumented counter activity and calibrated to in situ monitoring of process and temperature, avoiding the overheads of direct measurement [85–87].

In general, the most effective counters measure control signals and handshake signals, in which a single toggle is a marker of a large amount of activity in the block. These counters can be polled or sampled intermittently by a power-management unit. In the context of microprocessor design, these circuits are referred to as microarchitectural counters. This section considers common microarchitectural counters that can instrument a typical microprocessor for use in power management. While microarchitectural counters are common in commercial systems, these counters are often designed for debugging and performance tuning by the platform engineers, and may not be accessible for power management [88]. It is critical that the microarchitectural counters proposed here be easily and quickly accessible by the hardware PMU for effective FG-AVS.

Processor Core Counters

Processor cores can be instrumented with microarchitectural counters that record their utilization. These counters can apply to the entire core, such as an "instructions retired" counter that increments whenever an instruction is successfully retired. They can also be distributed to particular functional units, which may be useful in systems with high spatial granularity. Examples include arithmetic operation counters, floating-point operation counters, issue counters, and branch mispredict counters. These counters in aggregate can provide considerable detail on system activity. For example, a program region with many instructions retired and a high number of floating-point operations clearly indicates high workload demand.

Note that the presence of certain counters may be mandated by the instruction set architecture (ISA) specification. For example, the RISC-V ISA requires each hardware thread to maintain a cycle counter, a counter of instructions retired, and a real-time clock counter. Making these processor-visible counters available to the PMU provides a useful starting point for power management.

Memory System Counters

Activity in the processor memory system is often a strong indicator of activity in the processor itself. Accordingly, each level of the memory system can be instrumented with counters, from the L1 caches to the last-level cache (LLC). Counters can include cache hit and miss counters in aggregate and by bank, as well as a breakdown of whether the cache activity is a load or a store and the originating hardware associated with the request. This information can allow detailed prediction of near-future workload demand. For example, a high number of L1 load misses can indicate a immediate period of low processor activity before the data is returned. (Store misses only cause the processor to stall if the write buffer is full or the memory system is otherwise overwhelmed with requests.) LLC load misses can indicate a longer period of inactivity in the hardware that generated the memory request [20], while L1 misses accompanied by higher-level cache hits indicate activity in the memory system. The information derived from memory system counters can be combined with that of core counters to provide a detailed picture of the activity of the whole system.

Instruction Sequence Counters

More sophisticated counter implementations examine the processor instruction stream for specific instructions or sequences that may indicate inactivity. Some instruction sets have instructions specifically designed for this purpose. For example, the RISC-V and ARM ISAs feature a wait-for-interrupt (WFI) instruction that indicates that the processor has no immediate work to complete. A counter that increments whenever this instruction is decoded, or otherwise tracks time spent in a WFI state, is an excellent marker of processor inactivity. Instructions that jump to the existing program counter location (PC) effectively act as WFI instructions as well [89]. Even if not defined in the ISA explicitly, certain instruction sequences may be frequently emitted as compiler idioms that correspond with low activity. A simple state machine could count instances of these sequences to indicate processor state to the PMU.

Queue Counters

As noted in Section 2.3, adjacent voltage domains must have queues between them that allow backpressure for flow control. These queues can be instrumented as indicators of activity by measuring the fullness of each queue over time. If the queues flowing into a voltage domain are full, and those directed out of the domain are empty, this indicates that the block has a higher activity demand than it is currently able to meet. Conversely, if the outgoing queues are full and the incoming queues empty, then the block can be slowed to save energy without impacting the performance of its neighbors. Queue-based counter implementations are well-suited for local feedback control in systems with many independent voltage domains [44, 90]. Queue counters are not restricted to processor implementations, but can be employed in any FG-AVS system.

4.2 Integrated Power Control

Once sensors have taken measurements that can be used to estimate workload demand, a controller must process this data to determine whether a change in operating condition would save energy. This control can be governed by either hardware, software, or some combination, with many different options for implementation. FG-AVS requires a controller that can read the sensed data and respond quickly to track fine-grained changes in workload.

Commercial SoCs often govern their voltage/frequency states at the operating system level. A thread responsible for computing the next operating state of the system is scheduled onto one of the system cores, and updates are provided at the time granularity of OS quanta, typically 10 ms or longer. The OS cannot access detailed information about the hardware implementation, as such detail is typically abstracted away through the Advanced Configuration and Power Interface (ACPI) standard that is implemented by most power management systems. ACPI requests are serviced by a PMU, which is typically implemented as a separate control processor. For example, the power management controller integrated into Intel's Sandy Bridge processor runs multiple simultaneous algorithms to respond to operating system requests while optimizing power and performance and guarding against thermal emergencies [85]. This OS control is insufficient for FG-AVS because of the unpredictability of OS scheduling and the slow rate at which changes to operating system conditions are actuated.

An alternative approach modifies program compilers to insert hooks indicating changes in workload composition that can be interpreted by hardware for changes in voltage and frequency [91–93]. The authors of [94] propose extending this technique to dynamic recompilation, in which these instructions are inserted based on the measured operating conditions after an initial profiling pass. However, schemes for FG-AVS control that rely fully on the compiler or the instruction stream cannot adapt to the actual runtime operating conditions of the SoC, which may vary considerably based on process, temperature, and activity elsewhere in the system. Furthermore, most application binaries must be compatible with a variety of hardware ecosystems, limiting the utility of hardware-specialized code sequences for all but constrained embedded environments.

Fast and accurate FG-AVS response is best suited to some form of dedicated hardware control. The hardware PMU must be able to read from counters, registers, or memory information about the state of the digital hardware in each voltage domain it controls in order to determine the appropriate operating condition for each domain. The PMU can take the form of a finite-state machine, either fixed or reprogrammable [44]; FSMs are fast and inexpensive to implement, but are not fully programmable. Alternatively, a fully-featured programmable processor can be dedicated to power management.

4.2.1 Z-scale Power Management Unit

This section describes a suitably performant but lightweight PMU implementation for FG-AVS systems [68]. The proposed hardware PMU implementation consists of a 32-bit 3-stage

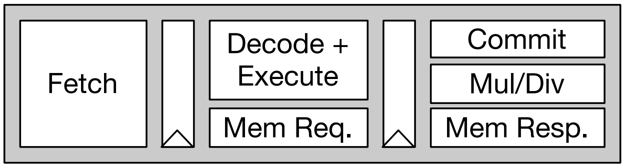


Figure 4.3: The Z-scale processor pipeline [68] (© 2017 IEEE).

single-issue in-order RISC-V processor based on Z-scale [95] as shown in Figure 4.3. The core forgoes caches in favor of an 8 KiB scratchpad memory, which is 128 bits wide and mapped into a portion of the physical memory space of the system. The PMU supports the RV32IM instruction extensions and is fully programmable via the RISC-V software toolchain. It is designed to reside in a fixed-voltage "always-on" domain for use in controlling the operating states of the remaining domains in the system.

The three-stage design minimizes gate count while enabling sufficient performance to enable fine-grained power management. The first stage of the pipeline fetches an instruction out of a 128-bit line buffer that reduces read-port contention on the single-ported scratchpad by storing four consecutive RISC-V instructions (the value of the program counter is calculated in the previous cycle). The second stage of the pipeline decodes a RISC-V instruction, reads the register file, and executes an ALU instruction. Branches are resolved in the second stage, so the instruction in the fetch stage is flushed when a branch is taken. Writeback is isolated into the third stage of the pipeline, reducing the fanout delay on the write port of the register file. The result in the third stage is bypassed to the second stage, eliminating the need for stalls in some cases. The memory stage and the multiplication/division pipeline stages are also in the third pipeline stage, although only one of these three subsystems will be active at a time, as their use triggers a stall in the second stage until the result of the instruction is written back to the register file. The multiply and divide units minimize hardware resources such that only one bit of the operation is completed each cycle, and so 32 cycles are required to compute any multiplication or division result. Loads and stores directly access the scratchpad; since the scratchpad is 128 bits wide, the pipeline must properly swizzle the load data and store data. An extra pipeline register was added in the arbiter between instruction and data memory requests to eliminate a long critical path and speed the achievable cycle time of the design. The entire design (excluding the scratchpad memories) uses just 18K gates, making the implementation overhead small relative to large application processors, accelerators, and caches.

For fast power-management feedback loops, it is critical that the PMU be able to read counters and actuate changes in voltage and frequency with low latency. The Z-scale PMU accomplishes this by mapping all system control registers, including both counters used to sense activity changes and control registers used to change the operating condition of a voltage domain, into its control status register (CSR) space, so that reading or writing from these registers accesses the appropriate system registers directly in hardware. CSR reads and writes are natively supported in the RISC-V ISA, so standard instructions can be used to write programs for power management. An alternative implementation memory-maps all system control registers, allowing the PMU core to access key system state with standard load and store operations.

4.2.2 Distributed Control

The previous section describes a single PMU that controls the voltage and frequency states of the entire system. This approach is practical for small designs with a limited number of independent voltage areas, but it can quickly become ungainly in systems with tens or hundreds of independent domains. A single PMU core may not be able to simultaneously process counter information from a large number of sources, causing latency in AVS response times, and the time taken to aggregate information over an entire die into a centralized controller further slows the speed of feedback. Instead of this single, centralized controller, distributed control assigns a local PMU to each voltage domain. These local PMUs, which usually operate in a separate, "always-on" voltage domain, use a limited amount of information about the state of the domain they oversee to determine when to change the operating mode of that domain. Implementations have demonstrated the feasibility of distributed control in SoCs featuring many homogeneous cores [96].

Distributed controllers can introduce new design challenges that would not be observed in a centralized approach. Because no single controller is observing the whole system, the local controllers may make decisions that are optimal locally but do not achieve a global maximum for energy efficiency. Worse, it is possible for distributed control systems to lose stability, resulting in unwanted, inefficient oscillations between voltage modes [90]. Ensuring the stability and optimality of distributed control systems is a nontrivial task that dramatically increases the complexity of designing and programming these systems [97]. Hybrid approaches, such as pairing distributed controllers with a global overseer, or making each distributed controller responsible for several neighboring domains, can mitigate these issues. It is likely that such a hybrid design, including elements of both local and global AVS control, is best suited for large FG-AVS systems.

4.3 Power-Management Algorithms

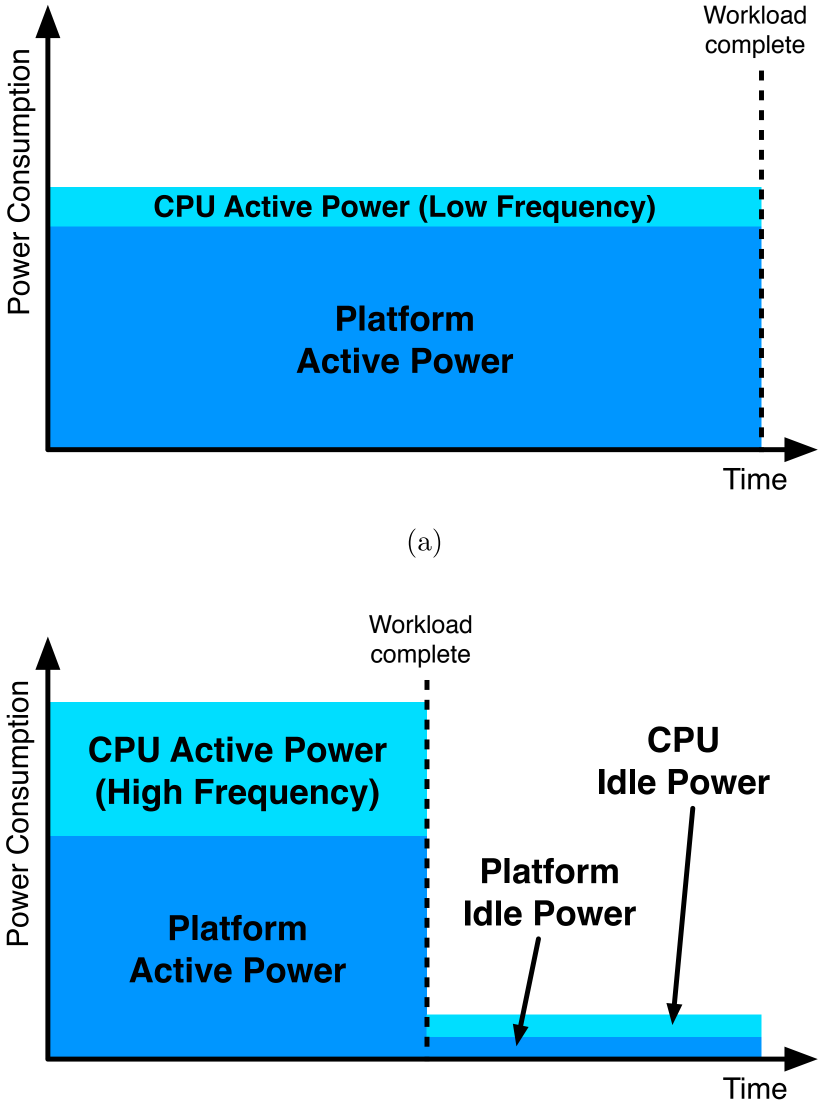
Systems designed to enable FG-AVS must employ some sort of algorithm to determine how best to change the voltage and frequency of each domain over time to improve energy efficiency. The possibilities for such algorithms are highly constrained by the sensors available to estimate activity or power as well as the implementation details of the PMU. The choice of algorithm is therefore an important consideration in system design as well as operation.

As explained in Section 1.2, digital logic is more energy efficient when operating at low voltages, and the minimum energy point for such logic is usually well below the minimum voltage V_{min} at which the system can operate. It may therefore seem apparent that the best energy efficiencies can be achieved by operating systems at the lowest possible voltage at all times. However, this approach does not account for several important constraints that tend to suggest operation at higher voltages, complicating the power-management optimization.

Continuous operation at V_{min} is usually not practical because of the loss of performance that low-voltage operation entails. Most systems operate under some performance constraint, which can be considered as a deadline by which some fixed amount of work must be completed. This constraint can result from a variety of scenarios. Some systems process workloads with a fixed, predictable latency requirement, such as the decoding of compressed video that must achieve a constant framerate. Other systems may have softer constraints, but a common goal is to complete user workloads as quickly as possible to present the most performant experience. Alternately, a deadline may be established to guarantee apparent responsiveness (such as responding to a touch or keypress before the human brain perceives the latency). In multicore systems running distributed multithreaded workloads that occasionally synchronize to a barrier, the deadline is effectively established by the thread with the most work, and other threads can slow as long as they do not become the slowest in the system. These varied performance constraints are often determined at the operating system level, with deadlines passed down to power management hardware so that the voltage and frequency can be adjusted to meet them. Such deadlines often require execution at higher, less energy-efficient voltages.

Even in scenarios without an explicit performance constraint, it may still save energy to complete workloads as fast as possible, a technique known as “race to halt”. Given that digital logic operates less efficiently at high frequencies, speeding execution to save energy may seem counterintuitive. In many systems, however, a large amount of static power is consumed not just by leakage and always-on systems in the SoC, but by other platform components such as memory, disk, fans, and other peripherals. As shown in Figure 4.4, even though the SoC is less efficient while operating at a high voltage, completing the workload as soon as possible allows large portions of the system to be powered down, saving energy in aggregate. The effectiveness of race-to-halt power management has been confirmed in several empirical studies of commercial systems [99–101]. A more recent evaluation shows that even in platforms with power-hungry CPUs, a tradeoff nonetheless exists between the static platform power consumed at low frequencies and the active CPU power consumed at high frequencies, resulting in a minimum energy point at some frequency greater than the minimum (see Figure 4.5) [98]. These platform considerations impose an implicit performance constraint on nearly all SoC workloads, encouraging fast completion both for improved performance and to save energy.

The goal of FG-AVS is therefore to save energy while still meeting some performance constraint. Unlike traditional voltage scaling as currently implemented in commercial systems, FG-AVS algorithms are generally not focused on the slow feedback loops involved in adjusting to changing deadlines as mandated by the operating system. FG-AVS control operates at a finer granularity, attempting to reduce voltage when possible to save energy with minimal impact on overall performance. This is illustrated in Figure 4.6. While coarse-grained voltage control has already reduced the voltage from its maximum because of a relaxed performance constraint, fine-grained control can further reduce this voltage during



(a)

(b)

Figure 4.4: Sample power consumption plots showing the benefit of race-to-halt power management. In (a), the CPU voltage and frequency is reduced so the computation completes more slowly. This reduces the total energy used by the CPU, but the platform consumes a large amount of energy because it remains active for the duration of the computation. In (b), the CPU operates in a faster, less energy-efficient mode, allowing the platform and CPU to be put in a lower-power idle state more quickly and reducing overall energy consumption.

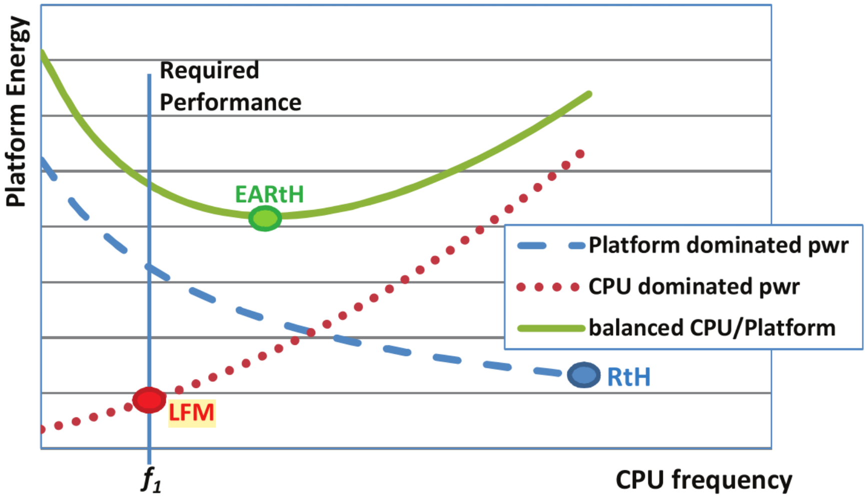


Figure 4.5: A plot showing the minimum-energy point achieved when accounting for both platform power and CPU power [98] (© 2014 IEEE).

periods of idleness without significantly impeding program progress.

Fundamentally, FG-AVS algorithms must set the voltage and frequency of each domain based on some prediction of the workload requirements of that domain in the near future. Algorithms typically use one or more sensing mechanisms to observe the present and recent past with the goal of predicting future activity. With perfect prediction and an instantaneous feedback loop, energy savings can be maximized without any performance loss. Slower feedback and mispredictions about future activity cost energy (from mistakenly operating at too high a voltage) or performance (from mistakenly operating at too low a voltage). The remainder of this section considers different classes of FG-AVS algorithms that can be used for system power management.

4.3.1 Interval-Based Algorithms

Interval-based algorithms collect data about the system over a fixed interval, and then analyze it to determine the appropriate voltage level in the next interval. These algorithms rely on the assumption that system activity will change relatively infrequently, so that successive intervals usually have similar behavior. In processor systems, common metrics for determining activity are memory traffic, cache misses [102, 103], instructions per cycle [104–106], or some weighted average of these parameters [107]. Most prior work has shown that despite the complexity of processor systems, observing a small number of counters is usually sufficient to accurately predict activity. On-chip networks designed for fine-grained voltage scaling can measure network traffic at each node and its neighbors to determine the voltage-frequency setting for the next interval [108]. A simple threshold can be used to determine the appropriate voltage setting.

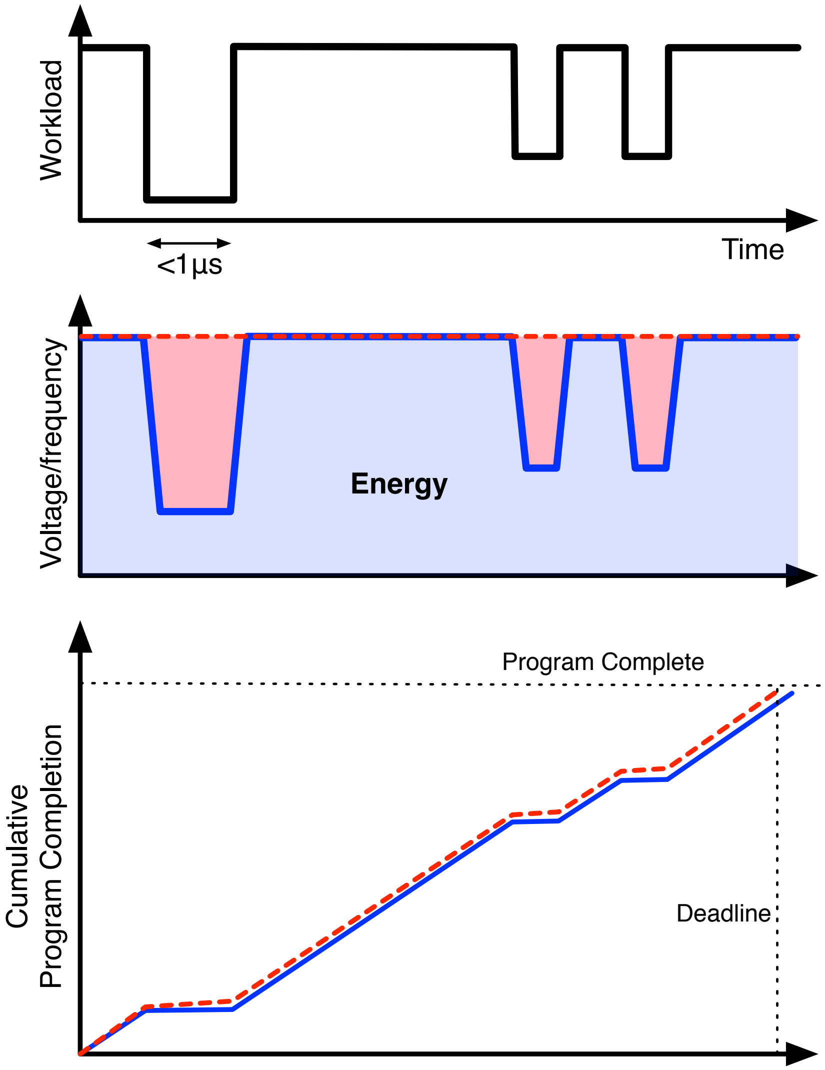


Figure 4.6: Applying fine-grained AVS to save energy. The coarse-grained AVS system (in red) has reduced the voltage and frequency as much as possible while still meeting the fixed deadline, but it cannot track rapid changes in workload. The fine-grained AVS system (in blue) can reduce voltage and frequency during periods of low activity, saving significant energy (shaded red) with minimal impact on execution time.

A more generalized approach performs feedback based on average queue depth observed over some interval. The authors of [109] propose a proportional-integral feedback controller based on queue depths in a generalized multi-voltage-domain system, changing voltage in response to queue utilization into and out of each domain.

Interval-based approaches are predictable and easy to understand. However, much of the prior analysis of these approaches assumes long interval durations of thousands or even millions of cycles over which to collect and average information. These longer intervals provide reliable information, but are likely to contain many different program phases and so are unsuitable for the rapid changes desired in FG-AVS. Depending on workload, interval-based monitoring with short interval durations may be less effective because activity can vary considerably at smaller timescales, so activity in one interval may not accurately predict the next.

4.3.2 Event-Driven Algorithms

Unlike interval-based approaches, event-driven algorithms change voltage in response to some particular event that represents a change in state. This technique has the advantage of fast response, as voltage changes are actuated as soon as possible, without waiting until the end of any interval duration. A related approach uses a rolling average, a hybrid of interval-based and event-driven techniques that performs some filtering of individual events but continually updates the result based on new data.

One common proposal for event-driven FG-AVS is to reduce the voltage when a last-level cache miss is detected and then increase it when the request returns [110]. This can save energy without impacting performance because processors typically stall for many cycles while waiting for the slower memory system to respond to a request. The authors of [111] adjust this approach by reducing voltage only when an LLC miss is followed by a low issue rate. The authors of [20] further note that only load misses are likely to result in a processor stall, and examine in more detail the cycle-by-cycle behavior likely to yield maximum energy savings. The utility of L2-miss scaling depends substantially on the ability of the processor to do useful work under the memory request. Many out-of-order cores are designed to exploit instruction-level parallelism to hide memory latency as much as possible. Nonetheless, this approach is a promising direction for FG-AVS, particularly for in-order systems.

Alternative event-driven approaches take advantage of different predictions of system behavior. The queue-utilization scheme described in the previous section can be modified to use a rolling average instead of set intervals [44, 97]. The authors of [44] also propose an alternative event-driven approach in which a certain number of stall cycles (as determined by full output queues or empty input queues) result in a voltage decrease. Multiprocessor systems can take advantage of existing thread-scheduling libraries to scale cores based on hints given by the compiler about relative workloads [112]. Each of these event-driven approaches use particular events to predict near-future system behavior for FG-AVS.

4.3.3 Activity Predictors

Rather than specific events or behavior over intervals, activity predictors provide a PC-based prediction of near-future processor activity. Using similar techniques to branch predictors, these hardware structures combine elements of sensing and power management. By using other counters in the system as metrics of overall workload each cycle, activity predictors build up a table of predictive data about which instructions will be followed by periods of high and low activity that can then be used to change the voltage ahead of time. One proposal records processor and memory activity over intervals to predict whether instruction sequences, when executed again in the future, will be compute-bound or memory-bound [113]. This information can then be used to slow the processor during memory-bound regions while speeding it during compute-bound regions.

Chapter 5

Fast Asynchronous Interfaces

A key barrier to FG-AVS in space is the difficulty of safely transferring data between two asynchronous clock domains. This chapter describes the advantages and drawbacks of the traditional solution to synchronization. Several alternatives to this traditional approach that can reduce the latency of this interface crossing are presented, including a novel bisynchronous FIFO based on the concept of pausible clocks.

5.1 Standard Bisynchronous FIFO

Effective fine-grained adaptive voltage scaling in space requires many voltage domains that can vary their clock rates independently to track local workload. As described in Section 2.3, fully asynchronous clock domains give rise to the danger of metastability when signals cross from one clock domain to another. Metastability is typically prevented by adding several series flip-flops on a signal line that crosses clock domains, which reduces its probability at the expense of additional interface latency. Despite the apparent simplicity of this synchronizer circuit, care must be taken for its correct use [114]. In particular, multi-bit buses cannot be synchronized using this scheme, with each bit in the bus passed through its own series FF. The danger of this approach is demonstrated in Figure 5.1. If the data in the bus arrives near the clock edge, the resulting word can be invalid even if each bit in the word is safely resolved to a 1 or 0. However, if the value of the bus is known to increase or decrease by no more than one bit per cycle, then this condition can be correctly guarded against if the data word is first Gray-coded. The Gray code pattern ensures that an increment or decrement of the binary word will result in only a single-bit change in the associated Gray code, ensuring that no invalid word can be received as long as the period between updates exceeds the flip-flop setup time. This allows safe synchronization of words in this limited case.

As arbitrary data words cannot be directly transmitted from one clock domain to another, a more sophisticated approach is required for communication between clock domains. The most common such circuit is the bisynchronous FIFO [115]. (The design is sometimes referred to as an asynchronous FIFO, although it does not employ asynchronous logic.) The circuit

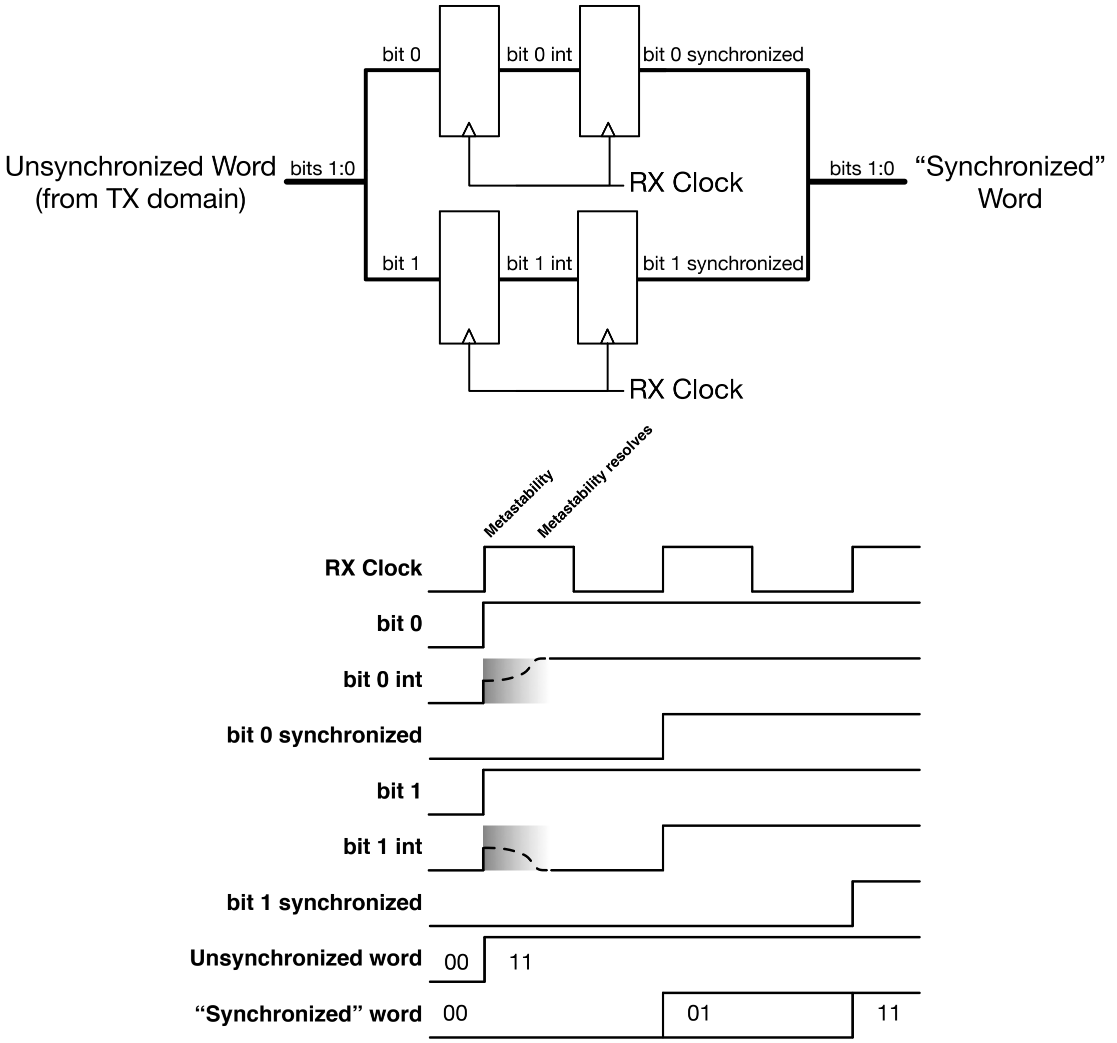


Figure 5.1: A waveform illustrating the danger of naive synchronization of data words via series flip-flops. In this example, the data word transitions from 00 to 11 near the RX clock edge, resulting in metastability. Each bit of the word resolves in the opposite direction, resulting in a cycle for which the output word is neither 00 nor 11.

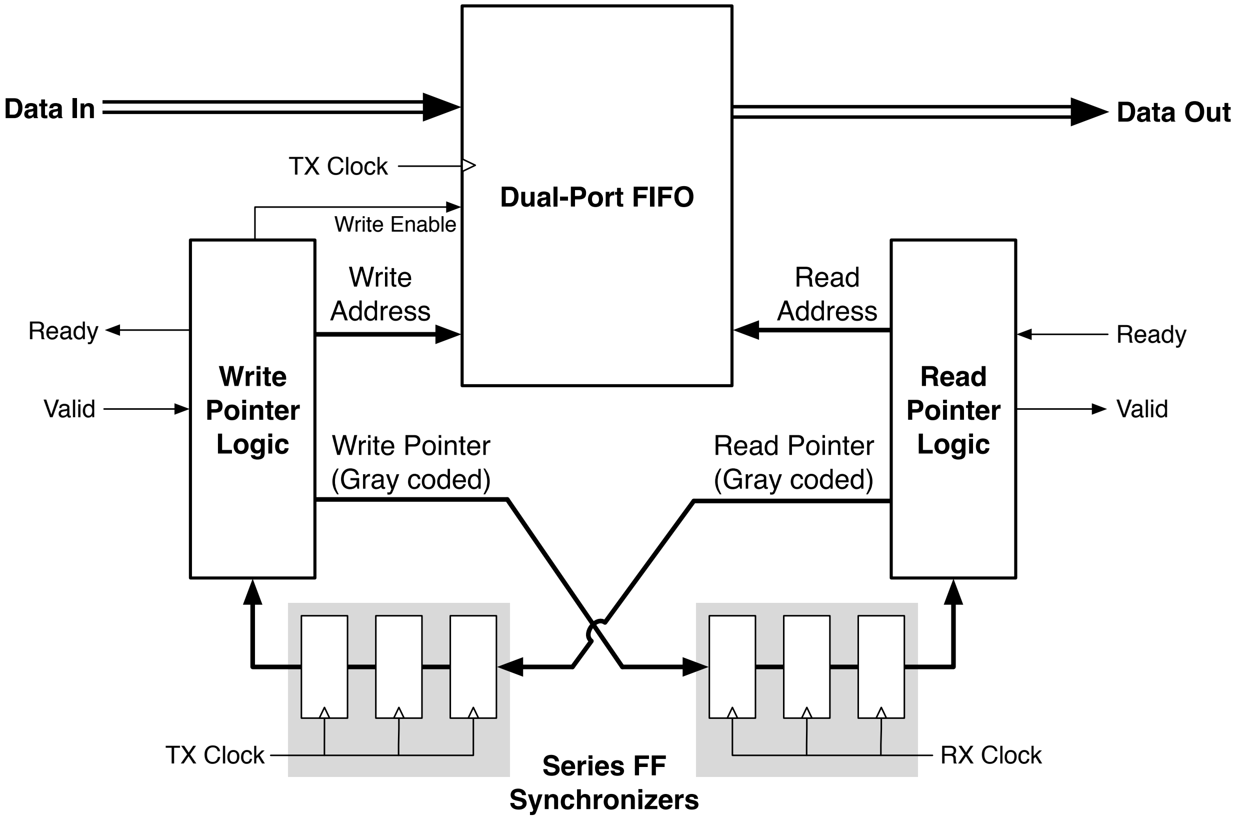


Figure 5.2: The standard bisynchronous FIFO.

is a modification of the synchronous FIFO queue with logic added for the asynchronous boundary. Communication with the transmitting (TX) and receiving (RX) domains is via standard ready-valid interfaces. The FIFO memory acts as a circular buffer, with read and write pointers tracking the location of valid data in the queue. The ready and valid signals at the interfaces are calculated based on the relative positions of the two pointers. Figure 5.2 shows a block diagram of the bisynchronous FIFO.

The safe transmission of data words across the interface is guaranteed by the logic that tracks the pointer positions. Data is stored in the FIFO by the transmitting domain only when it is known not to be full, and data is read from the FIFO in the receiving domain only when at least one valid word is known to be present. The synchronization challenge is not in the data words themselves, but instead in the read and write pointers. When a word is written to the FIFO and the write pointer incremented in the TX domain, the logic in the RX domain cannot simply access that state, as metastability could result. A similar issue occurs when the read pointer is incremented in the RX domain. Accordingly, the TX write pointer must be synchronized into the RX domain, and the RX read pointer must be synchronized into the TX domain. Because these pointers change only by incrementing, the Gray coding scheme described in the previous section can be employed to safely synchronize the pointer words into the neighboring domain via series flip-flops. These delayed representations of the pointers, once decoded, can then be used to update the logic in the opposite side of the FIFO.

The standard bisynchronous FIFO is widely used in academia and industry. It is a well-understood design comprising only minor modifications from synchronous queues, and it can be synthesized from standard cells and memories without any exotic circuits. It is robust to large differences in clock rate between the two domains, and the addition of series flip-flops beyond the two required at minimum can reduce the probability of metastability-induced failure to infinitesimal levels. However, the extra cycles required for pointer updates to be synchronized to the neighboring domain directly increase the latency of communication through the FIFO. In particular, when a word is read into the FIFO and the write pointer incremented, several cycles (equal to the synchronization latency of the pointer word) will elapse before that pointer increment is known to the receiving domain and the data can be safely read from the FIFO. In addition to the area overhead of the synchronizing flip-flops, therefore, the queue may need to be deeper than it otherwise would to provide proper flow control and allow for full bandwidth through the interface. (A shallower queue might prematurely fill and stall while pointer updates are still being synchronized so data can be read.) These drawbacks limit the performance of systems with many clock-domain crossings, and have spurred the development of alternative synchronization techniques.

5.2 Alternative Clock Domain Crossing Circuits

This section reviews several alternatives for transmitting data safely between clock domains while avoiding multi-cycle latency penalties.

5.2.1 Self-timed FIFO Interfaces

One approach to data synchronization builds on initial efforts to synchronize data across a mesochronous interface, in which two clocks have the same frequency but some unknown phase relationship. The fundamental insight of this approach, known as Self-Timed at Receiver Interface, or STARI, is that if the TX and RX frequencies are matched, then a simple dual-clock FIFO will suffice as a synchronizing interface. A single word will be read out of the FIFO and a new word written into the FIFO each cycle, so no logic is required to guard against overflow or underflow. This FIFO can be reduced to a single flip-flop that is read and written once per cycle, allowing for single-cycle latency between the domains (see Figure 5.3). The authors of [116] demonstrate how control logic can be implemented to enforce the necessary timing constraints.

This basic design can be extended to handle interfaces with arbitrary relative frequencies. A ratiochronous crossing, in which one clock is a rational multiple of the other, can be accommodated by multiplying the slower clock for use in the latch control logic, so that the flow control continues at the correct rate between domains. An arbitrary pair of frequencies

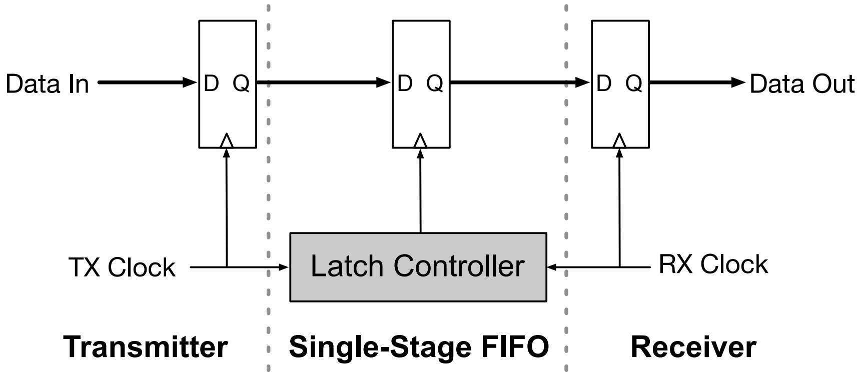


Figure 5.3: A single-stage synchronizing FIFO based on a carefully timed flip-flop [116].

can then be approximated as a ratio, so that this scheme can be used. Any inaccuracy in the approximation will manifest itself as phase drift between the two clocks, which can be guarded against by additional detector logic that skips a cycle of transmission when the relative phases approach an unsafe difference. This approach, however, is robust only when the clock frequencies in neighboring domains are stable over time. Frequent dynamic frequency scaling would invoke large overheads as the frequency multipliers at the interface were adjusted, and the use of adaptive generation would make it difficult to guarantee that the requisite timing constraints at the interface could be met under all possible operating conditions.

5.2.2 The Even-Odd Synchronizer

A different approach, known as even-odd synchronization, does not assume or estimate any rational relation between neighboring clock domains. The even-odd synchronizer samples the TX data twice per cycle into two different registers (see Figure 5.4). One register is written on the rising edge of the TX clock (the “even” sample) and the other is written on the falling edge (the “odd” sample). Since these two samples are taken out of phase, at most one of these updates can be at an unsafe time that would cause metastability in the receiving domain. The receiver can therefore safely accept the register that was most recently written at a safe time, guaranteeing a sub-cycle synchronization latency [117]. A frequency and phase detector in the receiving domain determine which samples are unsafe and therefore which register (even or odd) to read each cycle. This basic concept can be extended into a bisynchronous FIFO, similar to that described in Section 5.1 except that the series flip-flop synchronization of the read and write pointers is replaced by even-odd synchronizers.

The even-odd synchronizer works best when the frequencies of the TX and RX clocks are constant. Although the slow drift of plesiochronous clocks can be accommodated if the drift is predictable, changes in frequency otherwise require the halting of data transmission while

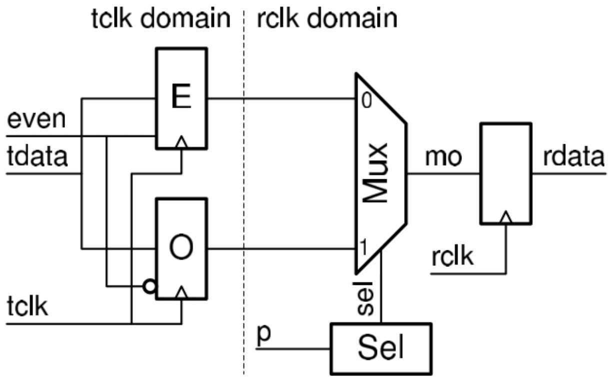


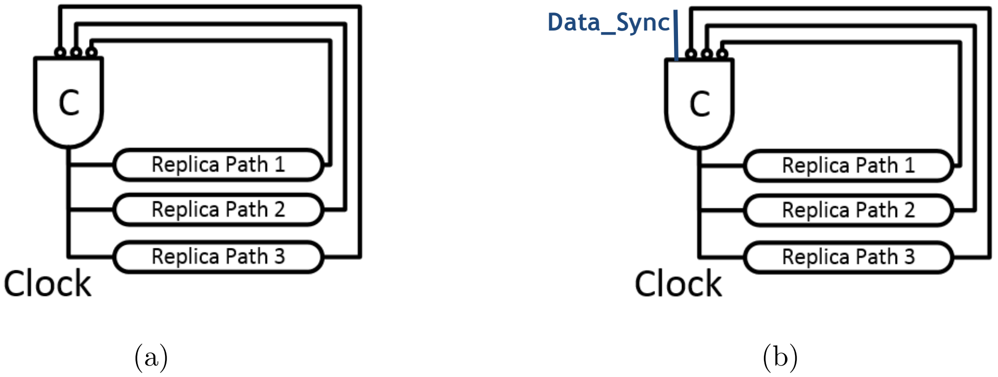
Figure 5.4: The even-odd synchronizer [117] (© 2010 IEEE).

the new frequency and phase are detected. Margining to ensure correctness in the case of changes in frequency or phase can slow the interface and require increasingly sophisticated phase detection logic, which can impose significant area overhead compared to a standard bisynchronous FIFO. Rapid, numerous frequency changes, such as may be the case in a system with adaptive clock generation, may render the even-odd synchronization scheme impractical.

5.2.3 Pausible Clocks

Pausible clocking is an alternative synchronization technique that modifies clock generation to handle the task of synchronization. Rather than delaying the transmitted data signal while metastability is resolved, pausible clocking delays the next clock edge of the entire clock domain to wait for the resolution of metastability [118]. This means that in the typical case, the latency of the data word through the pausible interface can be nearly as fast as that of fully synchronous logic.

Pausible clock generation is effected by modifying a typical local clock generator. Consider the free-running clock generator in Figure 5.5a, which contains several replica timing paths gated by a Muller C-element. This stateful logic gate, common in asynchronous logic, holds its value unless all inputs transition to another value. The circuit therefore safely generates a clock by delaying the next edge until the previous edge has propagated through the worst-case replica delay path. A pausible clock generator modifies this circuit with an additional sync input as shown in Figure 5.5b, so that the clock edge will not be generated until a synchronization circuit has toggled this signal to indicate that it is safe to do so with no danger of data metastability. In the typical case, this sync input will be immediately toggled after the previous clock edge, so the next clock edge will be generated as normal. Occasionally, the sync input will be delayed long enough to cause the clock generation to "pause" while synchronization is taking place. Eventually, though, synchronization will complete, the sync signal will toggle, and the next clock edge will be issued.



(a) (b)

Figure 5.5: Local clock generators. The standard circuit (a) can be modified with an additional input (b) to enable pausible clocking.

The sync input, which must toggle after each clock edge only when it is safe to generate the next one, can be generated in several ways. One common technique uses a mutual exclusion (mutex) circuit as shown in Figure 5.6 [119]. The mutex is an asynchronous element with two inputs and two outputs. Each output is high only if the corresponding input is high, but the circuit prevents more than one output from going high at a time. The mutex can be used in feedback to serve as the synchronizer in the pausible clock circuit (see Figure 5.7). The generated clock is inverted and sent to the r1 input of the mutex, while a Data_Sync signal indicating that new data has arrived from a neighboring clock domain serves as the other input. (This latter signal is asynchronous in this clock domain and can toggle at any time.) The mutex guards against a data request propagating into the circuit near the rising edge of the clock, which is when metastability can occur. If a request arrives near the rising edge of the inverted clock input, one of three outcomes is possible (see Figure 5.8). If the request arrives slightly later than the inverted clock edge, then the request will be delayed until the next clock transition, when it is safe for data to pass through without danger of metastability. If the request arrives slightly earlier than the inverted clock edge, then the clock transition will be delayed until the data has been safely captured. If the signals arrive at the same time, the mutex itself can go metastable, delaying both the clock and the data transition until the metastability resolves into one of the above cases. Even so, there is no danger of metastability in the data itself, and the circuit is guaranteed to eventually achieve correct operation. This pausible circuit can also be extended with an additional gating input to enter a "synchronous mode" that can be useful for test or scan as shown in Figure 5.9.

This basic pausible clocking scheme can be incorporated into an asynchronous FIFO as shown in Figure 5.10 [121]. This circuit uses two-phase (or non-return-to-zero) signaling to indicate the presence of new data and to acknowledge that this data has been written or read from the FIFO. An XOR gate therefore toggles high when there is a new request that has not yet been received by the FIFO, and the output of this XOR gate is used as the

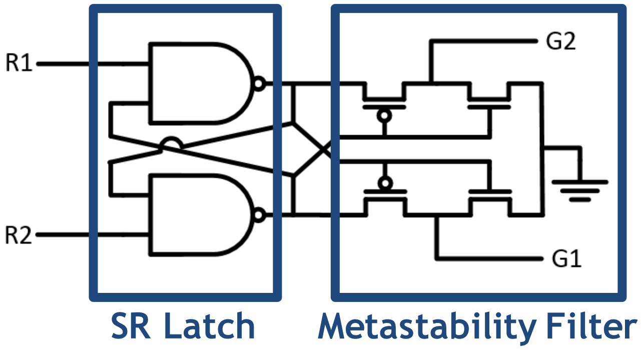


Figure 5.6: A mutual exclusion (mutex) circuit. Any metastability in the SR latch is blocked from the output by the metastability filter.

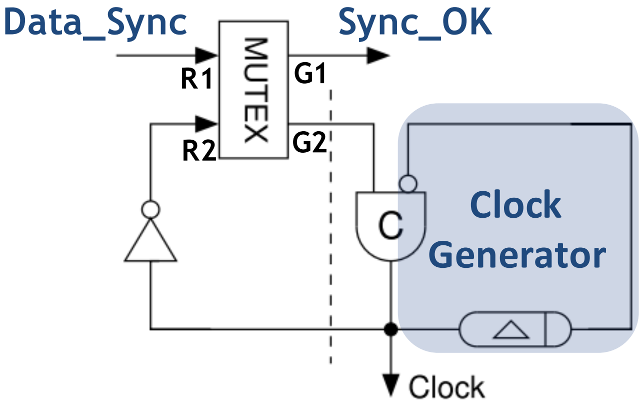
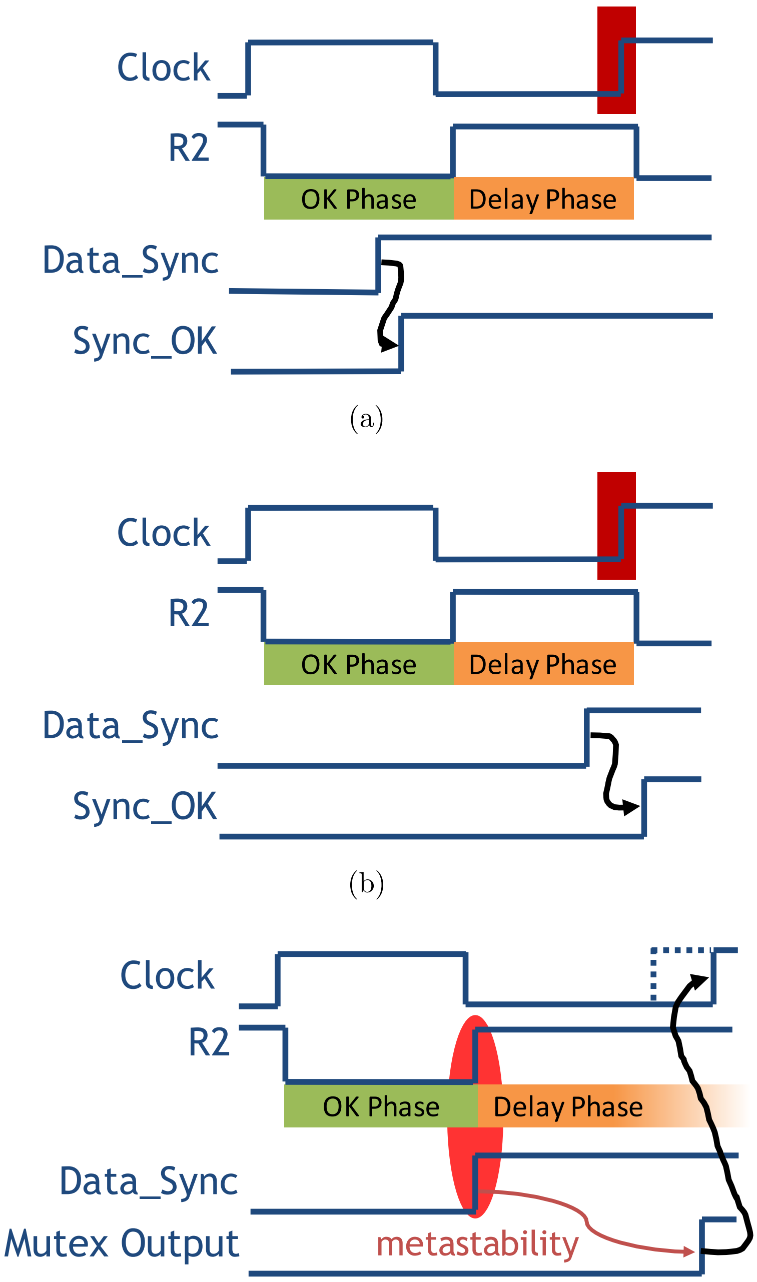


Figure 5.7: A mutex used to enable safe asynchronous boundary crossing via pausible clocks [120].

synchronization guard for the mutex circuit in the pausible clock generator. This circuit uses a fully asynchronous FIFO, which can be implemented using micropipelines [122], GasP [123], Mousetrap [124], or some other style of asynchronous FIFO. The circuit can achieve full throughput with average latency that is limited only by the propagation delay through the asynchronous FIFO.

5.3 Pausible Bisynchronous FIFO

As described in the previous section, prior implementations of pausible-clock-based clock crossing use a fully asynchronous FIFO to transmit data between clock domains. However, asynchronous FIFOs have several disadvantages that limit their utility as synchronizing elements between two synchronous domains. First, these FIFO implementations are not circular buffers, but true pipelines, and so incur additional energy and delay costs when



(a)

(b)

(c)

Figure 5.8: Waveforms showing the response of the pausible clock circuit to three different data arrival times. The red region near the clock edge represents the period during which it would be dangerous for the output Sync_OK signal to transition. If data arrives during the "OK" phase when r2 is low, then it is passed through (a). If data arrives during the "delay" phase, then it is delayed until after the next clock edge (b). If the data arrives at the boundary between these two phases, the mutex circuit can go metastable, potentially delaying the next clock edge (c).

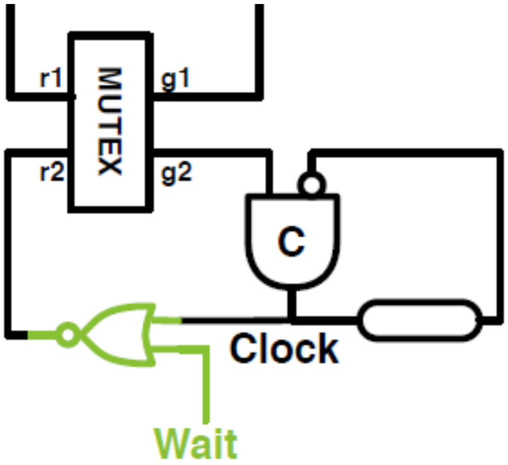


Figure 5.9: A pausible clock circuit with an additional gating input.

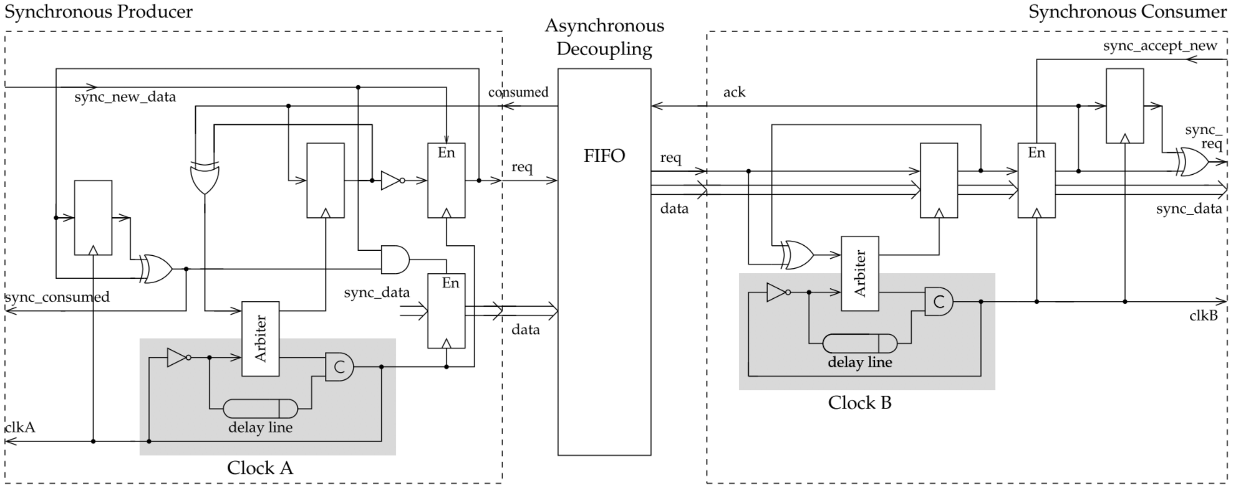


Figure 5.10: A pausible asynchronous FIFO [121] (© 2002 IEEE). A fully asynchronous queue buffers data in the FIFO.

moving data words through long pipelines. Circular buffers, on the other hand, do not move data, instead reading and writing based on pointer locations. Furthermore, asynchronous FIFOs often require exotic non-standard logic gates or complex timing constraints, neither of which are well-suited to use with standard VLSI synthesis or verification toolflows.

5.3.1 Circuit Design and Operation

Figure 5.11 shows a pausible bisynchronous FIFO that combines aspects of the standard bisynchronous FIFO with the pausible clocking circuits described in the previous section to safely transmit data between two clock domains with unknown, potentially varying frequency and phase relationships [120]. This design uses a standard synchronous FIFO based on a circular buffer. However, the read and write pointers are not synchronized between domains via series flip-flops, but instead using a novel control scheme based on *pointer increment* and *pointer acknowledge* signals. These two-phase signals toggle when the read or write pointer has been updated. Because this can happen at any RX clock phase, the toggle signal is gated by a latch, which is only made transparent when the mutex in the pausible clock circuit allows the edge to pass through, guaranteeing safety relative to the RX clock domain. The typical path for the pointer update is therefore quite fast, allowing data to be quickly read out of the FIFO once it is written. Because the acknowledge signal in response to each increment must also be synchronized, it can be several cycles before the increment line is ready to be toggled again. Accordingly, several increment-acknowledge pairs must traverse each domain. They are toggled in round-robin order by the control logic to allow full throughput through the FIFO. Each increment and acknowledge signal is synchronized through its own latch and mutex; the output of the mutexes is ANDed together and the result sent to the C-element of the clock generator, ensuring that all synchronization has completed safely before the next clock edge is generated.

The lettered circles in Figure 5.11 show the sequence of operations required to write a data word into the FIFO in the TX domain and read it in the RX domain. We assume that the FIFO is initially empty. The sequence begins when valid is asserted in the TX domain (A). As data is written into the FIFO, the write pointer logic increments its internal representation of the write pointer and toggles the next pointer increment signal in sequence (B). This toggle is then held at the latch in the RX domain until it is synchronized by the pausible clocking circuit (C). If the RX clock happens to be high when the toggle triggers r1 to transition high, then r2 will be low and g1 will immediately go high, opening the latch and de-asserting r1. If the RX clock is low, then r2 will be high, blocking the propagation of the signal at r1 through the mutex until the next clock edge has passed. In rare cases, r1 and r2 will transition high simultaneously, resulting in metastability and a possible delay of the next clock edge until the state resolves. Regardless, the safe transmission of the pointer increment signal is guaranteed to eventually occur. Once this propagation has completed, the read pointer logic updates its own internal representation of the write pointer. Since this now differs from its read pointer, the logic asserts valid and the data word is read out of the FIFO. (This triggers an update to the read pointer which must be synchronized to the TX

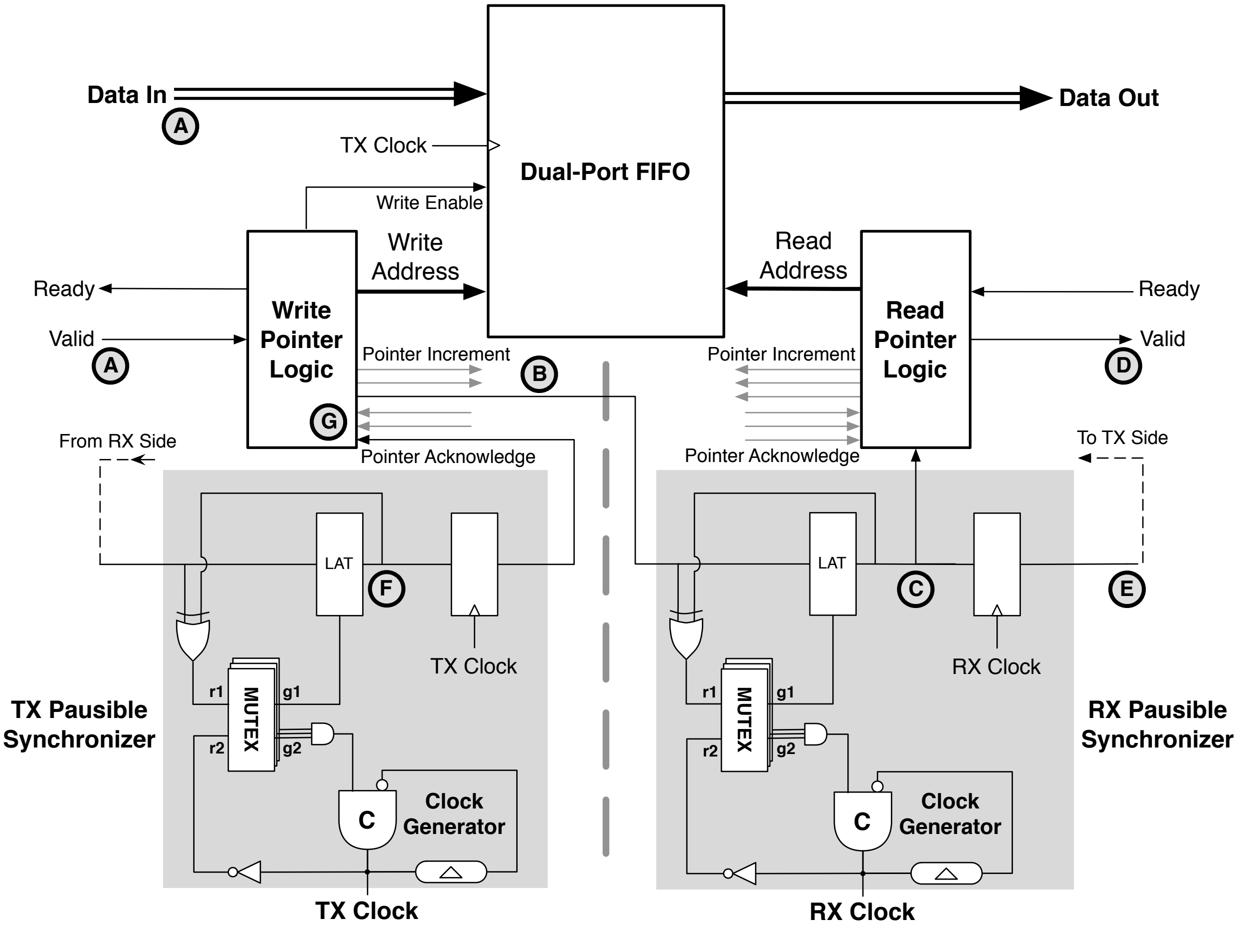


Figure 5.11: A pausible bisynchronous FIFO [120] (© 2015 IEEE). The lettered circles show the sequence of operations to move data through the FIFO.

domain; that sequence is not described in further detail.) Meanwhile, the pointer increment signal must be transmitted back to the TX domain as an acknowledge signal before the increment line can be freed for future reuse (E). This signal is synchronized through the same method in the TX domain (F), and is received by the write pointer logic after the next TX clock (G).

5.3.2 Timing Analysis

Because synchronization through the pausible bisynchronous FIFO can affect the generation of the clock in the receiving block, the design must meet several key timing constraints to operate as designed. This section analyzes these constraints to illuminate the tradeoffs and limitations of the design.

Figure 5.12 shows the key delays in the pausible synchronizing element of the system. t_{r2}

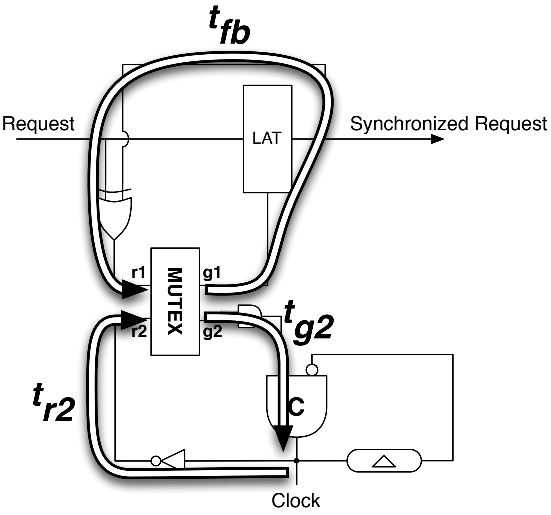


Figure 5.12: The key delays in the pausible clock circuit [120] (© 2015 IEEE).

is the delay from the output of the C-element to the r2 input of the mutex. t_{fb} is the delay from the r2 input through the mutex and around the feedback loop to the r1 input. t_{g2} is the delay from the r1 input through the mutex, AND tree, and C-element. The most important timing constraint in the system is that the sum of these three delays cannot exceed the desired time between clock edges, that is, half the cycle time T of the block:

$$T/2 \geq t_{r2} + t_{fb} + t_{g2} \tag{5.1}$$

If this constraint is violated, then any signal arrival immediately before the clock edge will result in a clock pause, forcing the average clock period lower than desired as shown in Figure 5.13. If the constraint is met, then only metastability that increases the propagation time through the mutex will result in a clock pause. If this cycle time constraint is exceeded, then some additional margin t_m will guard against a clock pause:

$$t_m = T/2 - (t_{r2} + t_{fb} + t_{g2}) \tag{5.2}$$

Only metastability resulting in an increase in t_{fb} greater than t_m can cause a clock pause.

A second constraint concerns the setup time of the combinational pointer logic. The low latency of the FIFO depends on the ability of the pointer logic to safely update the ready or valid signals in response to a pointer increment in the same cycle that the increment is received. The worst-case timing constraint for this logic is shown in Figure 5.14. If a metastability event resolves in favor of r2, then there is only a limited amount of time t_{CL} in which to combinationally evaluate the pointer update before the next clock edge is generated:

$$t_{CL} = t_{fb} + t_{g2}. \tag{5.3}$$

If this constraint cannot be met, an additional cycle of pipelining is required, increasing the latency through the FIFO by one cycle. If the setup time constraint in Equation 5.1 is met

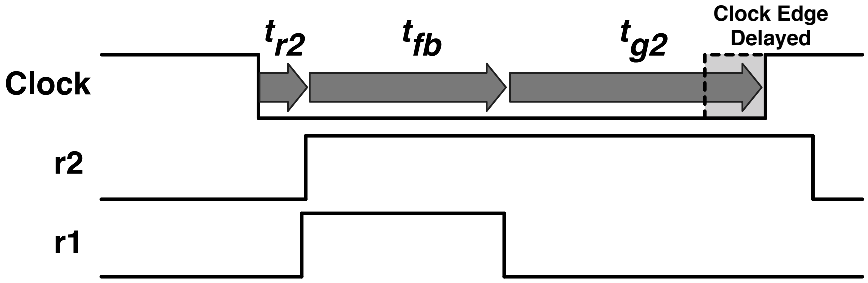


Figure 5.13: Waveforms showing the clock period constraint in the pausible clock circuit [120] (© 2015 IEEE). If this constraint is violated, the next clock edge will be frequently delayed, slowing the average clock rate.

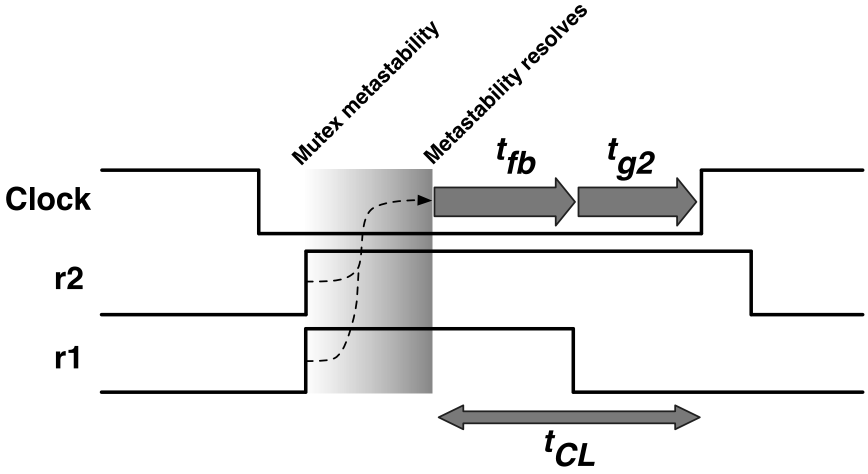


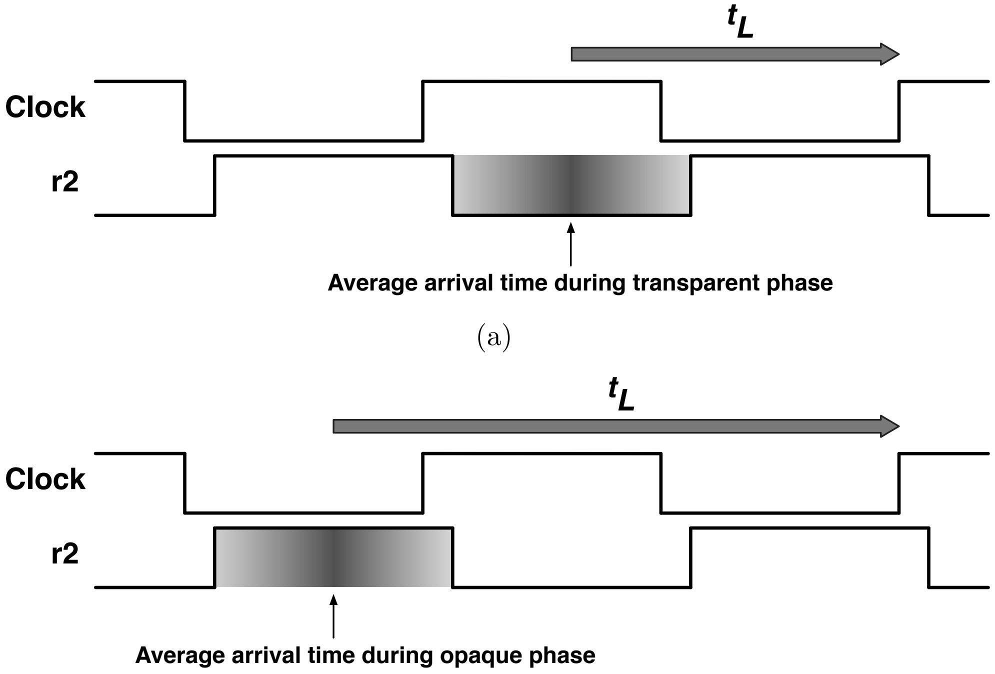
Figure 5.14: Waveforms illustrating the worst-case setup time for the combinational logic in the pausible interface [120] (© 2015 IEEE).

with additional margin t_m, that margin can be traded off in favor of deliberately increased t_{fb}, which increases t_{CL} at the cost of increased odds of a clock pause.

The average latency through the interface can also be calculated based on the above parameters. As shown in Figure 5.15, the average latency through the interface is $0.75T - t_{r2}$ if the request arrives when the mutex is transparent, and $1.25 - t_{r2}$ if the request arrives when the mutex is opaque. The average of these expressions therefore provides an overall average latency t_L:

$$t_L = T - t_{r2} \tag{5.4}$$

Increasing t_{r2} therefore decreases average latency through the interface because it shifts the transparent phase of the mutex closer to the next clock edge. Any excess t_m can be traded off in favor of deliberately increased t_{r2} to reduce average latency.


t_L
Clock
r2
Average arrival time during transparent phase
t_L
Clock
r2
Average arrival time during opaque phase

(a)

(b)

Figure 5.15: Waveforms showing the average latency through the pausible interface, as determined by taking the mean of the average latency during the transparent phase (a) and the average latency during the opaque phase (b) [120] (© 2015 IEEE).

Insertion Delay

The previous analysis assumed instantaneous propagation of the generated clock at the output of the C-element to all sinks in the design, but real implementations have some non-zero insertion delay t_{ins}. This insertion delay misaligns the mutex transparent phase, which can lead to unsafe value propagation near the rising clock edge as shown in Figure 5.16 [125]. If the insertion delay is smaller than t_m, then t_{r2} can be increased to match t_{ins}, realigning the mutex transparent phase. To guard against larger insertion delays, a lockup latch guarded by r2 can be added at the output of the synchronized request signal (see Figure 5.17). The latch guards against races while the clock propagates to its sink registers, and it does not increase the latency of the system because signals that do not race the clock would still have to wait for the next clock edge to be synchronized [126]. This latch therefore permits an extra half-cycle of insertion delay, although T_{CL} is decreased slightly because the propagation delay through the transparent latch must be subtracted from the combination logic setup time.

Assuming the presence of the lockup latch, the maximum tolerable insertion delay through the system is therefore

$$t_{ins} \leq T - t_{fb} - t_{g2} \tag{5.5}$$

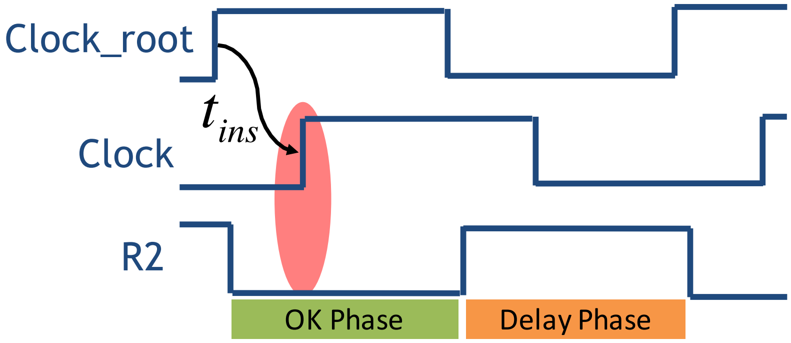


Figure 5.16: Waveforms illustrating the effect of insertion delay on the pausible clock circuit [120]. Insertion delay can misalign the phases of the circuit, resulting in unsafe transmission of data.

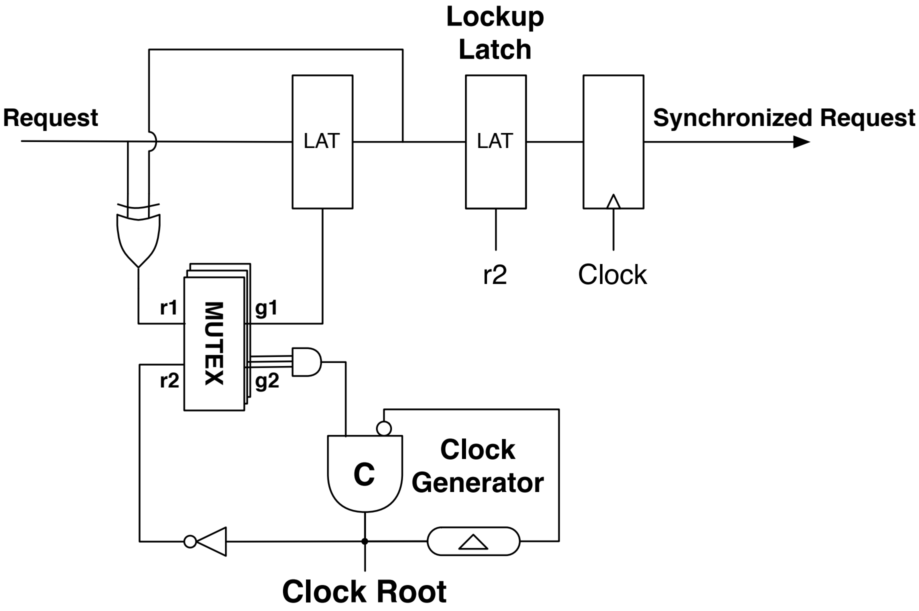


Figure 5.17: The pausible synchronizer with an additional latch to guard against the effects of insertion delay [120] (© 2015 IEEE).

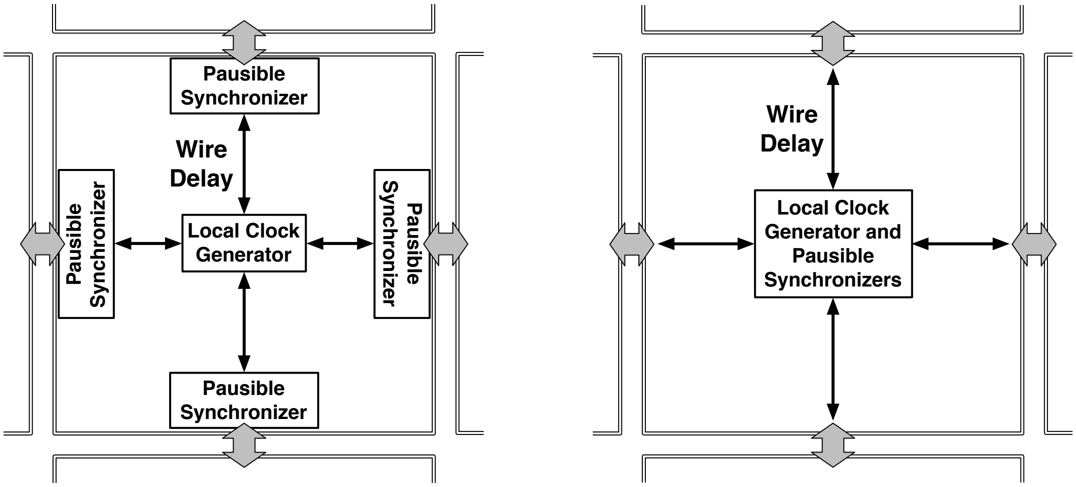


Figure 5.18: Two layout options for the local clock generator and synchronizer logic [120] (© 2015 IEEE).

The previous expressions for t_{CL} and t_L must also be adjusted for the presence of insertion delay:

$$t_L = T + t_{ins} - t_{r2} \tag{5.6}$$

$$t_{CL} = T/2 + t_{ins} - t_{r2} \tag{5.7}$$

Increased insertion delay increases average latency through the interface because the next clock edge is delayed relative to the transparent phase of the mutex, but it relaxes the t_{CL} constraint because of this same delay.

Wire Delay

The previous analyses assumed no additional wire delay between the location of the clock domain interface and the clock generation circuit. In practice, these two parts of the circuit may not be able to be physically co-located, leading to the two possible physical design options shown in Figure 5.18. If the synchronizer logic is located far from the clock generator, then some additional wire delay must be added to t_{r2} and t_{g2}, decreasing the maximum clock frequency and the maximum allowable insertion delay. If the synchronizer logic is placed near the clock generation circuit but far from the interface, then these constraints are not impacted, but latency equal to the wire delay will be added to the average latency of the interface. This second approach has the added benefit that the synchronization logic could be hardened along with the clock generator into a custom circuit, which could likely achieve better performance than a synthesized standard-cell-based design.

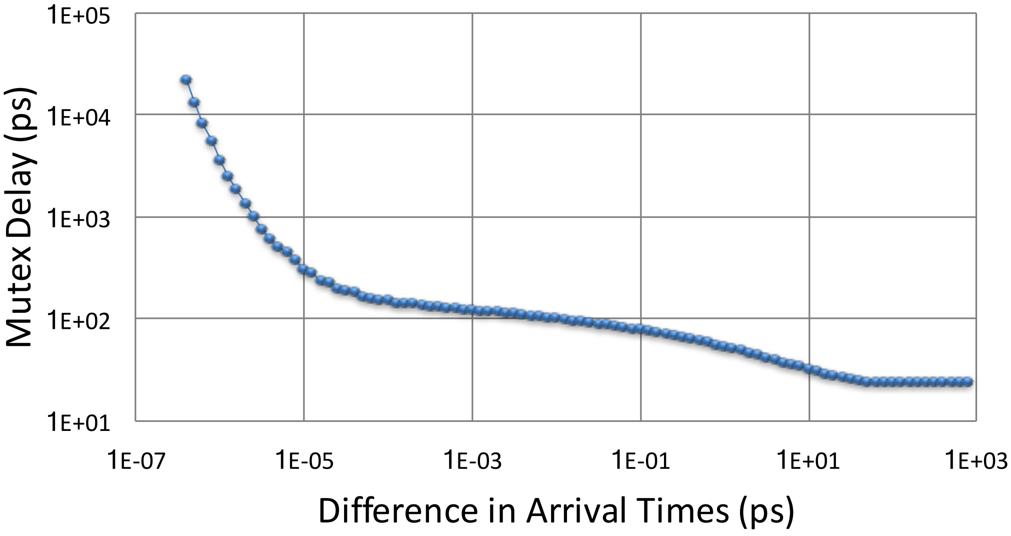


Figure 5.19: Simulated mutex delay for different input arrival times [120].

5.3.3 Implementation and Measurement Results

A 128-wide 8-element pausible bisynchronous FIFO was implemented in Verilog RTL for synthesis and simulation. The mutex circuit and local clock generators are described with behavioral models; the FIFO and pointer logic are fully synthesizable from standard cells. The mutex circuit model was based on simulation results from SPICE simulations of an implementation in a 28 nm CMOS process. Figure 5.19 shows the simulated impact of sweeping relative arrival times of the mutex inputs on delay through the circuit. Only input transitions that occur within a few tens picoseconds of one another cause any increase in delay through the mutex. Assuming a uniform distribution of arrival times, the average delay increase through the mutex is just 0.23 ps, and long pauses of 100 ps or greater occur less than one time in 10^6. Adding a small metastability margin t_m can further reduce this increase in the average clock period as shown in Figure 5.20. These simulations demonstrate the negligible average impact of the pausible clock technique on overall average cycle time. Figure 5.21 shows the results of RTL simulation using the mutex behavioral model.

The FIFO design was synthesized in the same 28 nm process to give timing and area estimates. The mutex circuits were defined as stateful elements with custom timing models so the synthesis tool would time their inputs and outputs appropriately. The timing restrictions described in the previous section were used to write custom timing constraints for synthesis. We chose to constraint t_{fb} and t_{g2} to 200 ps each. We also included a modeled insertion delay of 250 ps. Gate-level simulations of the synthesized netlist confirmed the correctness of the synthesis approach and were used to provide activity factors for back-annotated power modeling. These results were compared against baseline synchronous and standard bisynchronous FIFOs. Table 5.1 summarizes the results of the comparison.

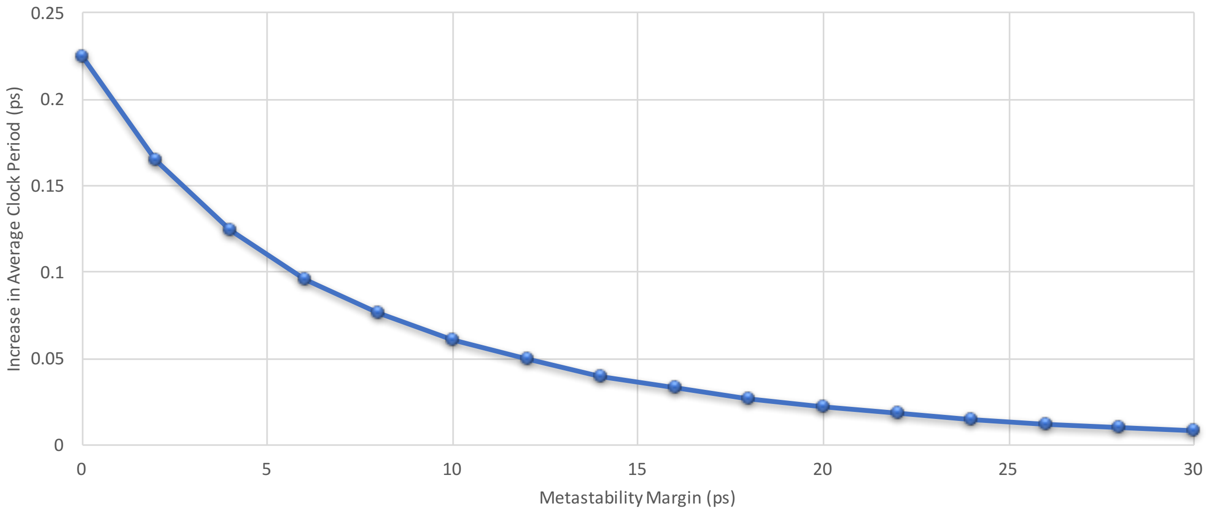


Figure 5.20: The effect of added t_m on average clock period perturbation.

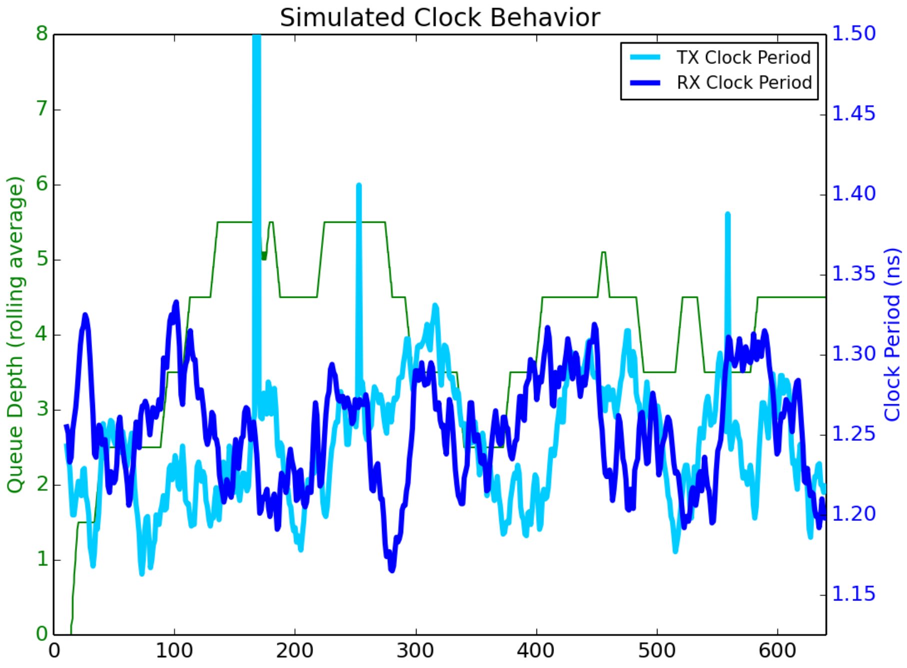


Figure 5.21: Sample simulation result. TX and RX clock period are randomly varied over time. The spikes in TX clock period are clock pauses triggered by simulated metastability.

	Average Latency (cycles)	**Area (μm^2)**	**Power (mW)**	**Energy (fJ/bit)**
Synchronous FIFO	1	4968	4.08	39.8
Standard Bisynchronous FIFO	4	5005	6.03	58.9
Pausible Bisynchronous FIFO	1.34	4808	5.41	52.8

Table 5.1: Pausible bisynchronous FIFO synthesis results and comparison [120].

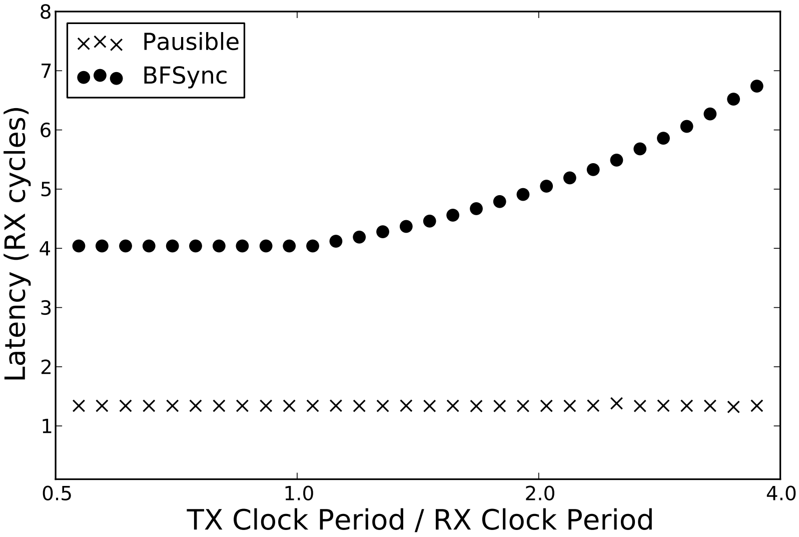


Figure 5.22: Simulated latency through the pausible FIFO with varying TX and RX clock periods [120] (© 2015 IEEE). BFSync is the standard bisynchronous FIFO.

Figure 5.22 shows the average latency through each interface as the ratio of the TX and RX clock is varied. The pausible FIFO achieves an average latency of just 1.34 cycles, which includes both the insertion delay and the wire delay described in the previous section that was estimated by the synthesis tool.

Chapter 6

Energy-Efficient SoC Design

The potential energy savings of FG-AVS systems are strongly dependent on the design choices and implementation overheads required to realize them. This chapter describes the implementations and measurement results of four testchips that implement aspects of fine-grained adaptive voltage scaling. Rather than testing individual components of FG-AVS systems, these testchips are fully-featured SoCs that can perform integrated adaptive control while executing realistic workloads. Demonstrating energy savings in realistic systems and use cases in silicon shows the real benefit of FG-AVS without obscuring any of the real costs and design tradeoffs of implementing such systems.

6.1 Raven-3: Efficient Integrated Regulation

The Raven-3 testchip features energy-efficient integrated voltage regulation, adaptive clocking, resilient SRAMs, and a fully-featured application processor [10]. A block diagram of the system is shown in Figure 6.1.

6.1.1 System Design and Implementation

The application core in Raven-3 implements the RV64G variant of the free and open RISC-V instruction set. The processor, an instance of the Rocket Chip Generator [127], is a 64-bit five-stage single-issue in-order pipeline (see Figure 6.2). The pipeline is carefully designed to minimize the impact of long clock-to-output delays of SRAM macros. For example, the pipeline resolves branches in the memory stage to shorten the critical path through the bypass path, but relies on extensive branch prediction (a 64-entry branch target buffer, a 256-entry two-level branch history table, and a two-entry return address stack) to mitigate the increased branch resolution penalty. The blocking 16 KiB instruction cache is private to the Rocket core, while the nonblocking 32 KiB data cache is shared between the scalar core and a vector coprocessor designed to accelerate data-parallel workloads. The Rocket core has a memory-management unit that supports page-based virtual memory. Both caches are

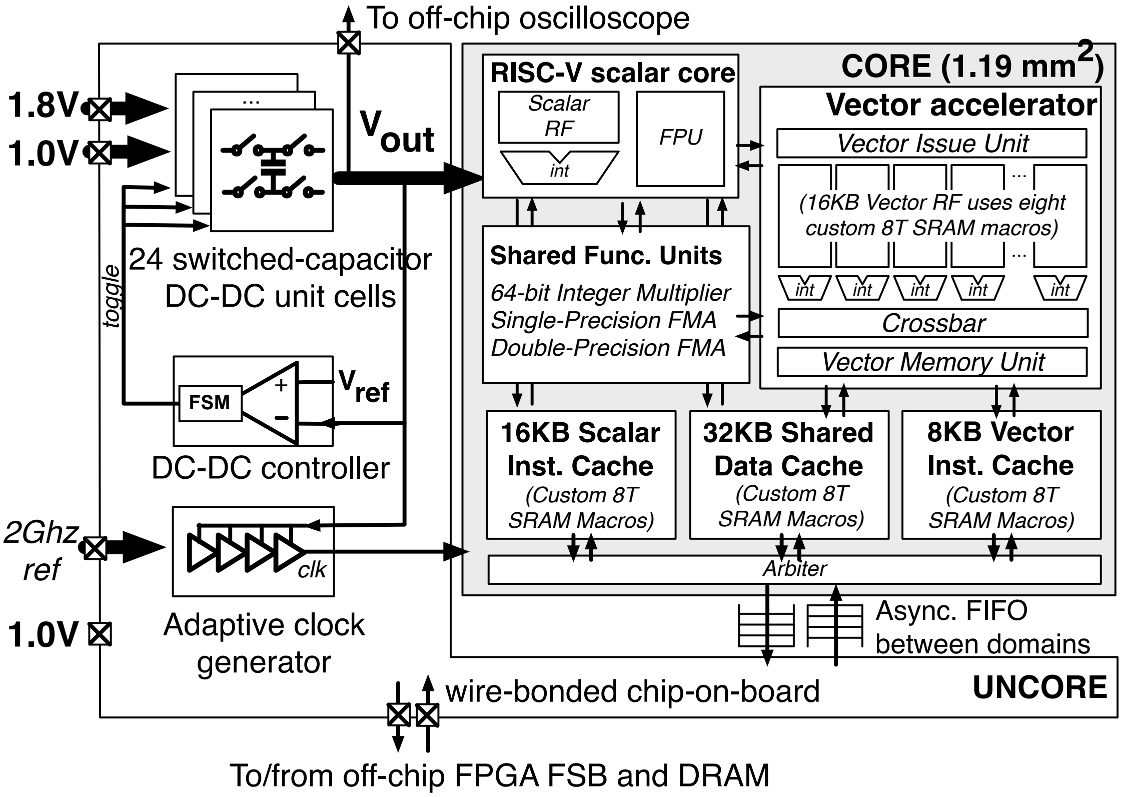

To off-chip oscilloscope
1.8V
1.0V
Vout
24 switched-capacitor DC-DC unit cells
toggle
FSM
Vref
DC-DC controller
2Ghz ref
clk
Adaptive clock generator
1.0V
RISC-V scalar core
Scalar RF
FPU
int
CORE (1.19 mm2)
Vector accelerator
Vector Issue Unit
(16KB Vector RF uses eight custom 8T SRAM macros)
int
Crossbar
Vector Memory Unit
Shared Func. Units
64-bit Integer Multiplier
Single-Precision FMA
Double-Precision FMA
16KB Scalar Inst. Cache (Custom 8T SRAM Macros)
32KB Shared Data Cache (Custom 8T SRAM Macros)
8KB Vector Inst. Cache (Custom 8T SRAM Macros)
Arbiter
Async. FIFO between domains
wire-bonded chip-on-board
UNCORE
To/from off-chip FPGA FSB and DRAM

Figure 6.1: A block diagram of the Raven-3 testchip [10] (© 2016 IEEE).

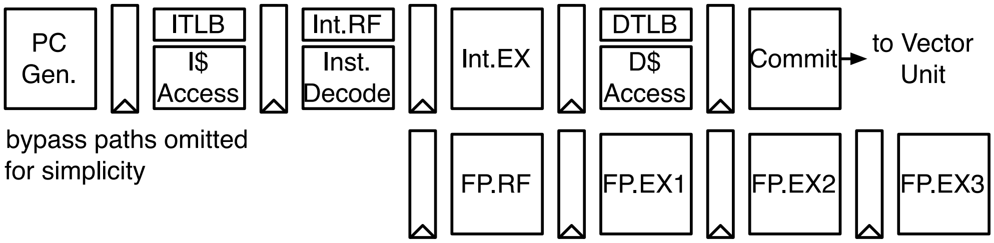


Figure 6.2: A simplified pipeline diagram of the Rocket processor [10] (© 2016 IEEE).

virtually indexed and physically tagged, and have separate TLBs that are accessed in parallel with cache accesses. The core has an IEEE 754-2008-compliant floating-point unit that executes single- and double-precision floating-point operations, including fused multiply-add (FMA) operations, with hardware support for subnormal numbers.

To reduce design complexity, the microprocessor is implemented as a tethered system. Unlike a standalone system, a tethered system depends on a host machine to boot, and lacks I/O devices such as a console, mass storage, frame buffer, and network card. The host (e.g., an x86 laptop) is connected to the target tethered system via the host-target interface (HTIF), a simple protocol that lets the host machine read and write target memory and

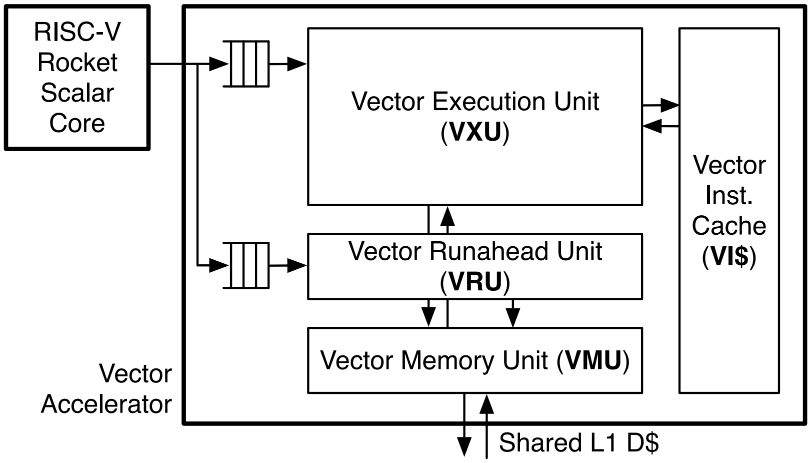

RISC-V Rocket Scalar Core
Vector Execution Unit (VXU)
Vector Inst. Cache (VI$)
Vector Runahead Unit (VRU)
Vector Memory Unit (VMU)
Vector Accelerator
Shared L1 D$

Figure 6.3: A block diagram of the Hwacha vector accelerator [10] (© 2016 IEEE).

control registers. All I/O-related system calls are forwarded to the host machine using HTIF, where they are executed on behalf of the target. Programs that run on the scalar core are downloaded into the target's memory via HTIF. The resulting system is able to boot modern operating systems such as Linux utilizing I/O devices residing on the host machine, and can run standard applications such as the Python interpreter.

The Hwacha vector accelerator, shown in Figure 6.3, is a decoupled single-lane vector unit tightly coupled with the Rocket core. Hwacha executes vector operations temporally (split across subsequent cycles) rather than spatially (split across parallel datapaths), and has a vector length register that simplifies vector code generation and keeps the binary code compatible across different vector microarchitectures with different numbers of execution resources. The Rocket scalar core sends vector memory instructions and vector fetch instructions to the vector accelerator. A vector fetch instruction initiates execution of a block of vector arithmetic instructions. The vector execution unit (VXU) fetches instructions from the private 8 KiB vector instruction cache (VI$), decodes instructions, clears hazards, and sequences vector instruction execution by sending multiple micro-ops down the vector lane. The vector lane consists of a banked vector register file built out of two-ported SRAM macros, operand registers, per-bank integer ALUs, and long-latency functional units. Multiple operands per cycle are read from the banked register file by exploiting the regular access pattern with operand registers used as temporary space [128]. The long-latency functional units such as the integer multiplier and FMA units are shared between the Rocket core and the Hwacha accelerator. The vector memory unit (VMU) supports unit-strided, constant-strided, and gather/scatter vector memory operations to the shared L1 data cache. Vector memory instructions are also sent to the vector runahead unit (VRU) by the scalar core. The VRU prefetches data blocks from memory and places them in the L1 data cache ahead of time to increase performance of vector memory operations executed by the VXU. The resulting vector accelerator is more similar to traditional Cray-style vector pipelines [129]

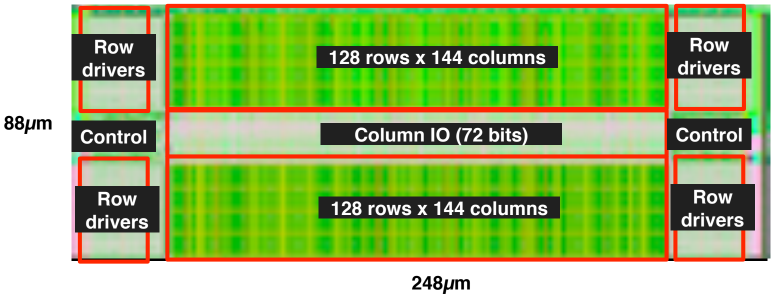


Figure 6.4: Annotated layout of the custom 8T SRAM macro used in the core voltage domain [68] (© 2017 IEEE).

than SIMD units such as those that execute ARM's NEON or Intel's SSE/AVX instruction sets, and delivers high performance and energy efficiency while remaining area efficient.

The Rocket core and Hwacha accelerator together comprise the variable core voltage domain that is supplied by the integrated voltage regulators and adaptive clock. The core voltage area was also placed under deep n-well to allow forward body bias (FBB) to be applied. Because SRAMs typically limit the minimum operating voltage of digital blocks due to the susceptibility of the small transistors in SRAM bitcells to process variation, all SRAM arrays in the core voltage domain use the same custom 8 KiB 8T-based SRAM macro shown in Figure 6.4. The macro is logically organized as 512 entries of 72 bits (64 bits + 8 possible error-correcting code bits) and physically organized as two arrays of 128 rows by 144 columns with two-to-one physical interleaving. Low-voltage operation is enabled by the 8T bitcell, where each transistor is larger than the equivalent high-density 6T bitcell. While the arrays also implemented a negative-bitline write assist, the assist was not necessary to achieve minimum-voltage operation.

The core voltage is supplied by an integrated SS-SC converter (see Section 3.1.2). 1.0 V and 1.8 V supplies are downconverted to rippling output voltages averaging 0.9 V, 0.67 V, and 0.5 V depending on the reconfigurable converter topology. The 1.0 V supply can also be passed directly to the core in a bypass mode. All 1 V input switches are implemented as low-V_t devices to reduce their ON resistance, while the larger 1.8 V input switches are implemented as regular-V_t devices with FBB applied to further reduce their leakage when active. The converter is subdivided into twenty-four 90 μm × 90 μm unit cells that are located near the core voltage area. To achieve simultaneous switching, the DCDC toggle clock generated by the controller must arrive at each unit cell at the same time. The place-and-route tool was therefore directed to construct a clock tree for this signal, balancing arrival times. The flying capacitor is implemented using MOS capacitors with two layers of MOM capacitors above. Parasitic bottom-plate capacitance is reduced by using a series connection of the box, well, and substrate capacitances [130]. The converter achieves a total capacitance of 2.1 nF and a capacitive density of 11.0 fF/μm^2.

As noted in Section 3.1.2, simultaneous-switching regulators require adaptive clocking

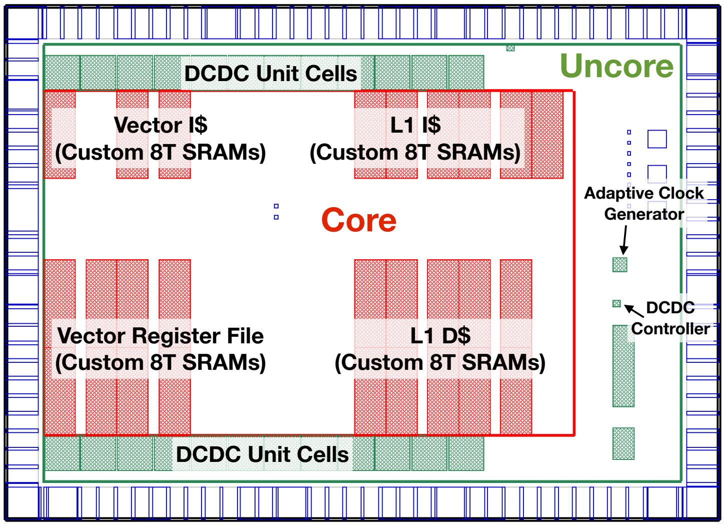


Figure 6.5: Annotated floorplan of the Raven-3 testchip [10] (© 2016 IEEE).

to achieve high system efficiencies. Raven-3 implements an early version of the adaptive clock generator described in Section 3.2. The clock generator selects clock edges from a 16-phase DLL output as determined by the delay through a replica critical path supplied by the core voltage [77]. The replica path is made up only of inverters, with the depth of the path programmable via selectable muxes. The generated clock is supplied to all register and SRAM sinks in the core voltage domain.

The IP blocks and their control registers, IO cells, and HTIF logic comprise the uncore voltage domain, which operates at a fixed 1 V and fixed frequency. Because the uncore and core may be asynchronous depending on the operating mode, standard bisynchrounous FIFOs guard all communication between them (see Section 5.1). Level shifters are inserted on signals crossing the domains to ensure correct operation as the core voltage varies.

A multivoltage and multiclock design flow was used to construct the processor. Figure 6.5 shows the processor floorplan, with the larger core voltage domain in red separated from the smaller uncore voltage domain to the right of the chip. The custom SRAMs were manually placed within the core voltage domain. The SS-SC unit cells surround the core to minimize voltage drop. Two layers of thick upper-layer metal were dedicated to a power grid, where the core voltage and ground each utilize 25% of the chip area in each layer. Outside the core, these core voltage rails are not necessary, so the input voltages to the converters use the majority of the power routing resources to connect power coming from the pad frame to the converters. An annotated die photo of the chip, which was fabricated in 28 nm ultra-thin body and BOX fully depleted silicon-on-insulator (UTTB FD-SOI) technology [131], is shown in Figure 6.6.

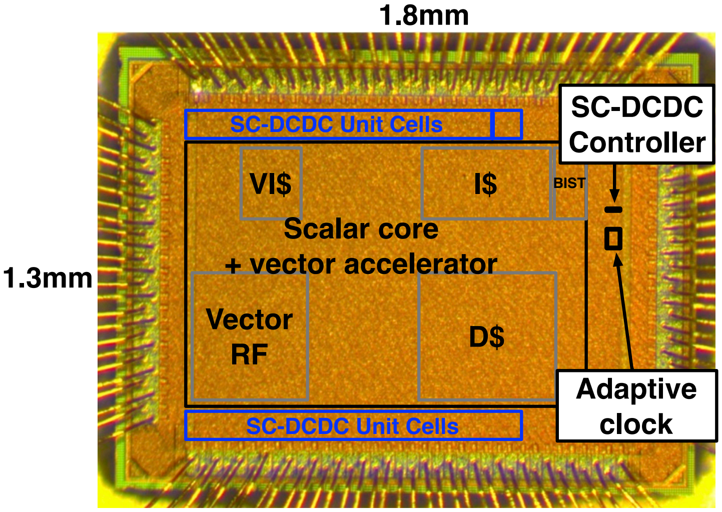


Figure 6.6: Annotated die micrograph of the Raven-3 testchip [10] (© 2016 IEEE).

6.1.2 Measurement Results

The testchip was wirebonded directly to a daughterboard to minimize bond wire inductance. A multimeter or oscilloscope connects to sense points on the daughterboard to measure the output voltage rail supplied by the SS-SC converter. The daughterboard is connected via an FPGA Mezzanine Card (FMC) connector to a motherboard which generates the necessary clocks, supplies, and reference voltages. Additional testpoints on the motherboard connect to a sourcemeter to measure the input power provided to the SS-SC converter. The motherboard connects via a second FMC to a ZedBoard [132] that includes an FPGA and a network-accessible ARM core. The serialized 16-bit digital interface of the testchip is connected to programmable logic in the FPGA, which can route memory traffic to local DRAM and emulate system calls. Software running on the ARM core acts as a host controller for the tethered test chip processor. Figure 6.7 shows a block diagram of the test setup, and Figure 6.8 shows a photograph of the system configured for test. Table 6.1 shows a summary of the chip and key measurement results.

Figure 6.9 shows oscilloscope traces of the rippling core voltage domain for all four SS-SC regulator configurations. As noted in Section 3.1.2, simultaneous-switching converters do not completely eliminate charge sharing because the load itself has a capacitive component. This results in observed average output voltages that are lower than the ideal ratios. Figure 6.10 shows the impact of adjusting the lower-bound reference voltage V_{ref} on conversion efficiency. Changing the reference voltage changes both the average output voltage and the switching rate of the converter, which affects the conversion efficiency. Each switching mode has a particular V_{ref} that achieves maximum efficiency at a given load. Different processor loads correspond to different optimal V_{ref} because the processor load affects the switching frequency of the regulators.

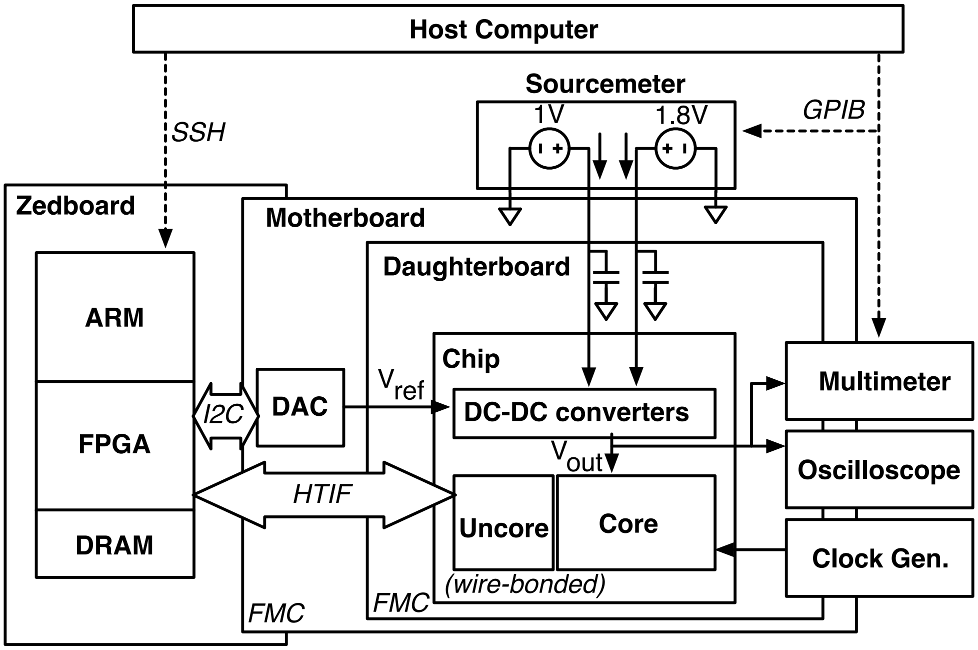


Figure 6.7: A block diagram showing the test setup of the Raven-3 testchip [10] (© 2016 IEEE).

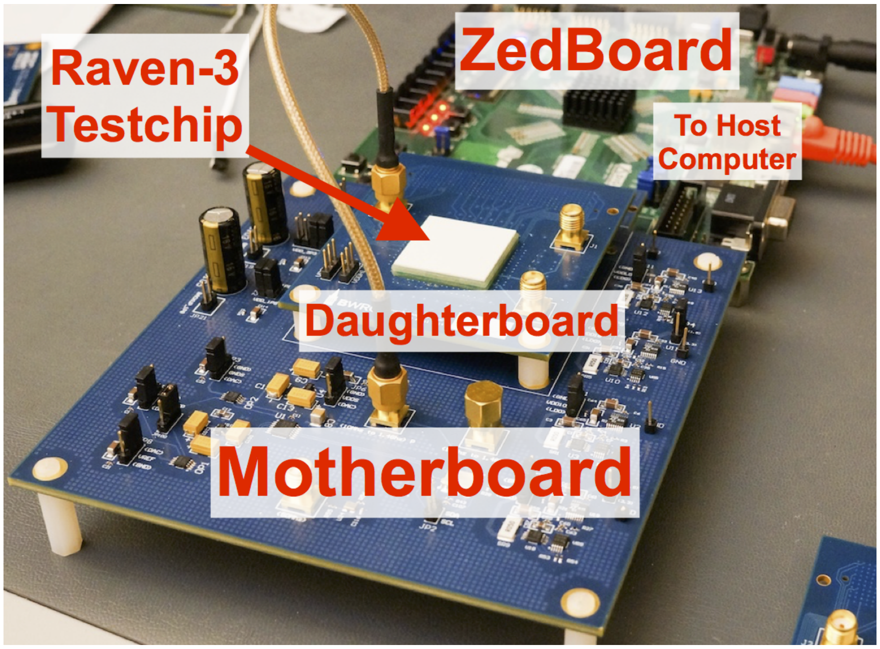


Figure 6.8: The Raven-3 testchip and associated infrastructure. The chip itself is obscured by the white protective covering.

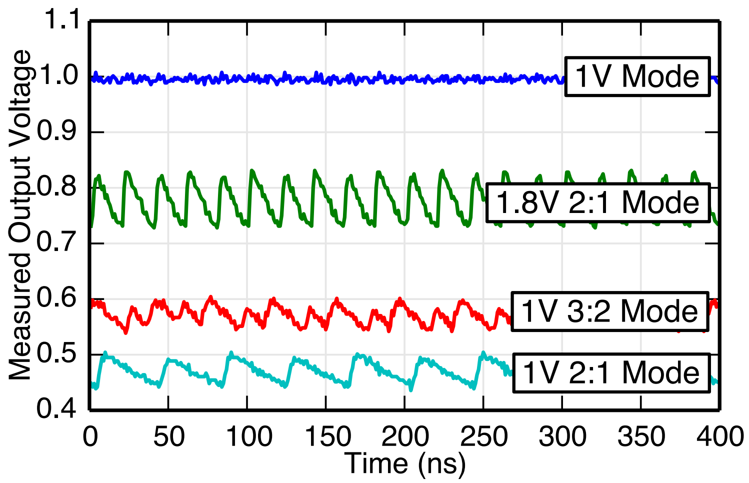
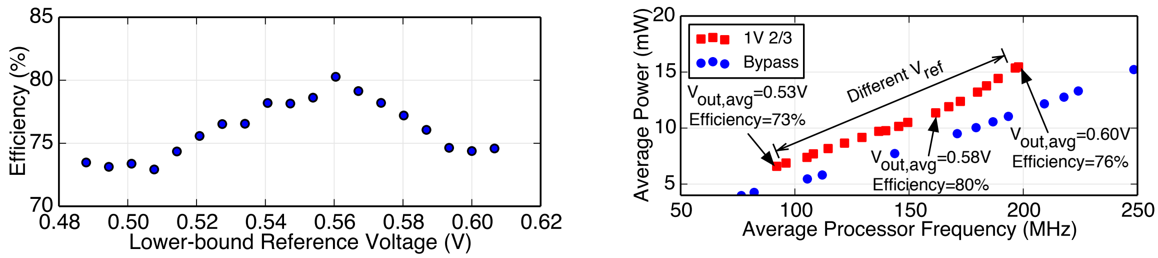


Figure 6.9: Oscilloscope traces of the voltages generated in the four SS-SC operating modes [10] (© 2016 IEEE).



(a)

(b)

Figure 6.10: Figures showing the effect of different lower-bound reference voltages on system operation [10] (© 2016 IEEE). (a) shows the system conversion efficiency as the voltage is swept, and (b) shows the effect of different V_{ref} on average output voltage and frequency.

Technology	28 nm FDSOI
Die Area	1305 µm × 1818 µm (2.37 mm^2)
Core Area	880 µm × 1350 µm (1.19 mm^2)
Converter Area	24 × 90 µm × 90 µm (0.19 mm^2)
Voltage	0.45 V to 1 V (1V FBB)
Frequency	93 MHz to 961 MHz (1V FBB)
Power	8 mW to 173 mW (1V FBB)
SC density	11.0 fF/µm^2
SC power density	0.35 W/mm^2 @ 88% efficiency

Table 6.1: A summary of key results and details from the Raven-3 testchip [10].

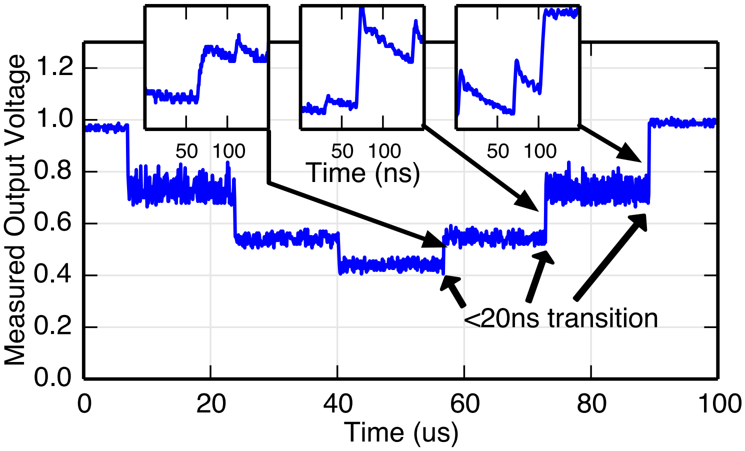


Figure 6.11: An oscilloscope trace showing the generated core voltage as the SS-SC converter rapidly cycles through each of the four operating modes [10] (© 2016 IEEE).

For all possible converter topologies with adaptive clocking, the processor successfully boots Linux and runs user applications, demonstrating that complex digital logic operates reliably with an intentionally rippling supply voltage. Figure 6.11 shows the rapid (<20 ns) transitions between different voltage modes. The application core can continue to operate through these mode transitions, demonstrating the utility of integrated regulators for FG-AVS.

The conversion efficiency of voltage converters is generally computed by measuring the current and voltage on both the input and output of the converter to measure the ratio of power delivered to power supplied. The regulator in the Raven-3 testchip cannot be measured in this way because it is it is difficult to measure on-chip voltage and current since the voltage is rippling very quickly. Furthermore, even if power output of the converter could be measured, this metric would ignore the impact of imperfect adaptive clock tracking, which is an important loss component. Therefore, a different method is required to measure the efficiency of the implemented system.

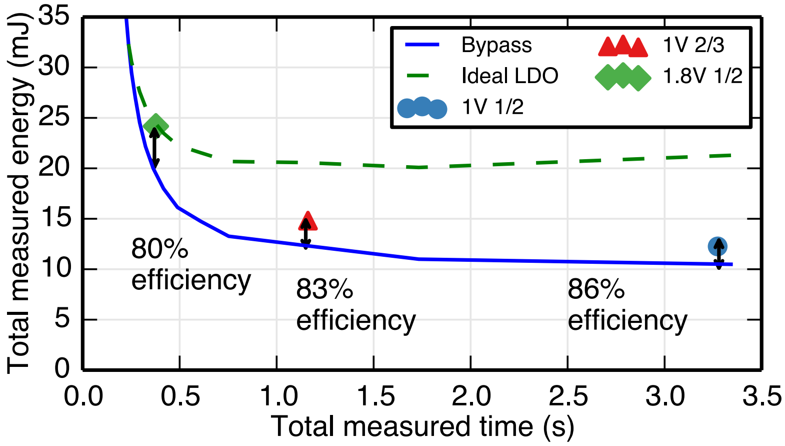


Figure 6.12: A plot demonstrating the system conversion efficiency of the three SS-SC switching modes [10] (© 2016 IEEE).

We define system conversion efficiency as the ratio of energy required to finish the same workload in the same amount of time under the regulated system as compared to a system without voltage conversion losses. In this metric, 100% efficiency represents a lossless regulator supplying the core as it operates at the maximum frequency achievable at that voltage. To determine this 100%-efficiency baseline, the bypass mode is used to directly supply the core with an ideal off-chip voltage source. A self-checking benchmark is run for a fixed number of cycles at different voltages, and a binary search is performed at each voltage point to find the maximum frequency. At the maximum frequency, the total elapsed time and total energy to run the fixed-length benchmark is measured, where the energy is computed by measuring the current drawn from the off-chip supply and the delivered voltage is measured from sense points on the core voltage (to remove the voltage drop across the on-chip bypass-mode power gates from the efficiency calculation). This baseline provides the blue curve in the Figure 6.12 and represents a 100%-efficient off-chip regulator. Then, for each SS-SC mode, the same benchmark is run for the same number of cycles, and the total elapsed time and energy is measured. Due to non-idealities of the converter, it takes more energy to perform the same task in the same amount of time. Therefore, system conversion efficiency is defined as the ratio of energy required to finish the same workload in the same time. This metric includes all sources of overhead, including non-idealities in the adaptive clock. As shown in Figure 6.12, the measured voltage conversion efficiency ranges from 80% to 86% for the different output voltage modes.

Figure 6.13 shows measured average frequency for different delay settings for the tunable replica circuit across a range of operating voltages (supplied using the bypass mode). Annotations above the plot indicate the approximate voltage ranges seen in each SS-SC voltage mode. Because the inverter-based replica path delay characteristics do not match the critical paths of the processor, a single delay setting poorly tracks the processor critical path over the entire voltage range. At higher voltages, a shorter replica path delay best tracks the

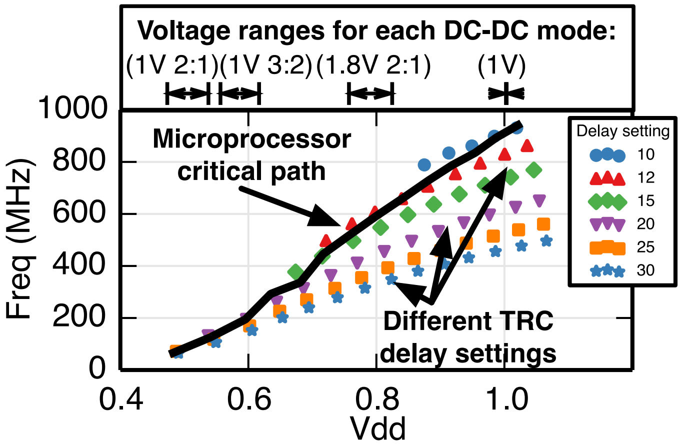
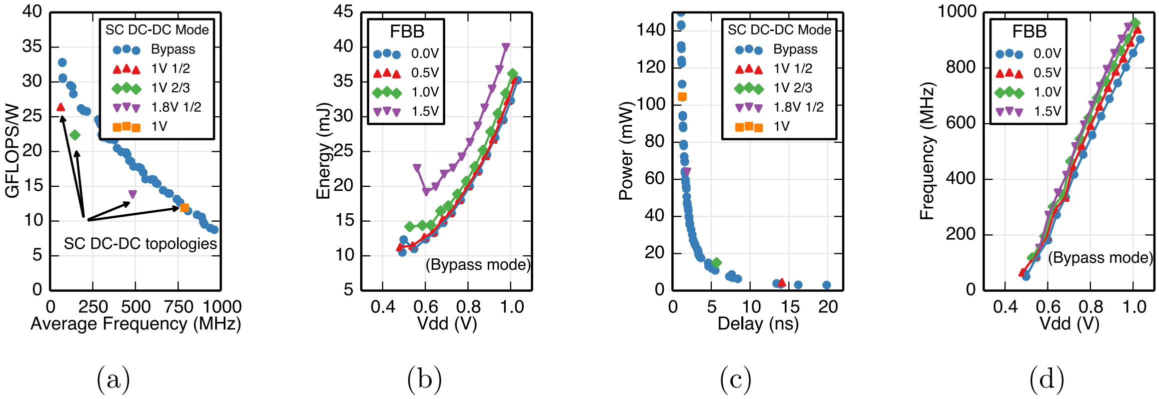


Figure 6.13: Measurements of the adaptive clock generator frequency over different voltages and replica path delay settings [10] (© 2016 IEEE).



(a)

(b)

(c)

(d)

Figure 6.14: Plots showing the measured energy, power, and frequency of the processor core in bypass mode and with the switching regulators enabled [10] (© 2016 IEEE).

processor critical path, while at lower voltages, a longer replica path delay best tracks the processor critical path. However, manual calibration of specific delay settings for each SS-SC voltage mode allows reasonably accurate tracking within the relatively small ripple of that mode.

Figure 6.14 shows various energy-delay curves for the application processor. Energy efficiency is measured by measuring total core energy consumption while executing a fixed-length double-precision floating-point matrix multiplication kernel on the vector accelerator, and is shown both under bypass mode and when accounting for the losses of the switching regulation modes. By using the on-chip converter to generate the lowest output voltage, the system achieves a peak efficiency of 26.2 GFLOPS/W. The FD-SOI technology of the testchip enables up to 1.8 V of FBB to be safely applied, which trades off improved performance for increased leakage [133]. Figures 6.14b and 6.14d show the impact of FBB on frequency and energy. In each of the plots, each point represents the best achievable operating frequency at a given voltage. The integrated system is able to achieve high energy efficiencies under the proposed voltage regulation and clocking scheme, demonstrating the potential utility of FG-AVS.

6.2 Raven-4: Integrated Power Management

The Raven-4 testchip builds upon the Raven-3 design with design changes and additional features [68]. The integrated voltage regulation and adaptive clocking systems are improved and supplemented with an integrated power management unit allowing fast FG-AVS algorithms to be demonstrated on die. An integrated body bias generator, power monitor, and measurement circuits enable more extensive power management and system characterization. A block diagram of the testchip is shown in Figure 6.15.

6.2.1 System Design and Implementation

The core voltage domain contains an updated version of the Rocket core implemented in Raven-3. The Hwacha vector accelerator is improved with the inclusion of a second floating-point functional unit, doubling the peak rate of computation to two fused multiply-adds (FMA) or four floating-point operations per cycle. Additionally, separate long-latency functional units are dedicated to the coprocessor rather than being shared with the Rocket core. The core SRAMs are implemented using the same custom 8T macros to enable low-voltage operation. Level shifters and bisynchronous FIFOs guard the crossings to the uncore clock domain, which contains the PMU, control registers, IP, and IO logic.

The integrated voltage regulator and clocking systems that supply the core are also improved from Raven-3. The flying capacitance of the SS-SC regulators is doubled to forty-eight 90 µm × 90 µm unit cells, enabling improvements in conversion efficiency. Decoupling capacitance was added to improve the integrity of the 1.0 V and 1.8 V inputs to the SS-SC unit cells and help offset power delivery issues caused by the wirebond packaging of the chip. Custom decoupling cells with MOS capacitors and a six-layer MOM 50 nm mesh add 539 pF of capacitance to the 1.8 V supply and 802 pF of capacitance to the 1.0 V supply. The clock generator is implemented as described in Section 3.2 alongside the version employed in Raven-3 for direct measurement comparison. Unlike the Raven-3 design, the replica timing circuit is comprised of multiple types of standard cells, each with an independently selectable depth. As noted in Section 3.3.2, large insertion delays diminish the effectiveness of adaptive clocking, so care was taken during physical design to reduce the insertion delay of the core clock tree compared to Raven-3.

A second, smaller processor serves as the PMU for the design. The PMU processor is based on Z-scale as described in Section 4.2.1. The PMU scratchpad memory is mapped to the upper half of the 4 GiB memory space, with the lower half reserved for the application

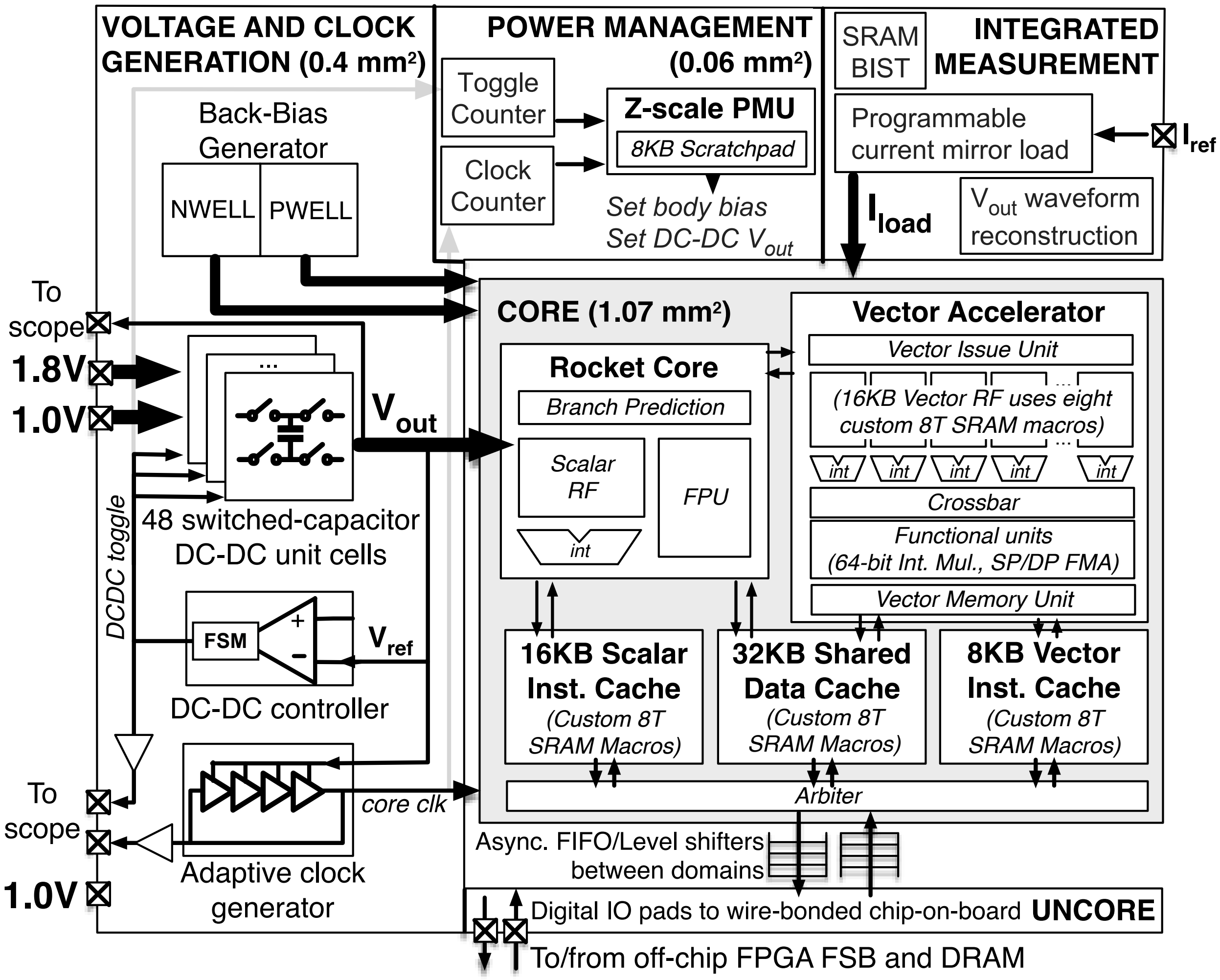

VOLTAGE AND CLOCK GENERATION (0.4 mm²)
Back-Bias Generator
NWELL
PWELL
POWER MANAGEMENT (0.06 mm²)
Toggle Counter
Clock Counter
Z-scale PMU
8KB Scratchpad
Set body bias
Set DC-DC Vout
SRAM BIST
INTEGRATED MEASUREMENT
Programmable current mirror load
Iref
Iload
Vout waveform reconstruction
To scope
1.8V
1.0V
Vout
48 switched-capacitor DC-DC unit cells
DCDC toggle
FSM
Vref
DC-DC controller
To scope
1.0V
core clk
Adaptive clock generator
CORE (1.07 mm²)
Rocket Core
Branch Prediction
Scalar RF
FPU
int
Vector Accelerator
Vector Issue Unit
(16KB Vector RF uses eight custom 8T SRAM macros)
int
Crossbar
Functional units (64-bit Int. Mul., SP/DP FMA)
Vector Memory Unit
16KB Scalar Inst. Cache (Custom 8T SRAM Macros)
32KB Shared Data Cache (Custom 8T SRAM Macros)
8KB Vector Inst. Cache (Custom 8T SRAM Macros)
Arbiter
Async. FIFO/Level shifters between domains
Digital IO pads to wire-bonded chip-on-board UNCORE
To/from off-chip FPGA FSB and DRAM

Figure 6.15: A block diagram of the Raven-4 testchip [68] (© 2017 IEEE).

core. The application core and the PMU can communicate directly via inter-processor interrupts, and each has a register mapped directly into the CSR space of the other system, allowing arbitrary data to be communicated between the two cores. Table 6.2 compares key features of the two processors. The processor maps the control registers for the SS-SC converters, adaptive clock generator, and other IP into its CSR space, which enables programs to directly manipulate the voltage and frequency of the chip. In addition, the core clock and the SS-SC toggle clock are read by counters, and the counter values are synchronized into the uncore domain. As described in Section 4.1.1, successive reads to the second counter enable a rapid estimate of core power consumption for active power management.

Several other circuits further the power management and measurement capabilities of the SoC. The threshold voltage of the logic in the core voltage domain can be manipulated by an integrated body bias generator that can supply up to 1.8 V of FBB [134]. Fine tuning

Feature	**Rocket (excluding coprocessor)**	**Z-scale**
Instruction Set	RISC-V RV64G	RISC-V RV32IM
Pipeline	Five-stage single-issue in-order	Three-stage single-issue in-order
Memories	32KB DCache, 16KB ICache	8KB unified scratchpad
Standard Cell Count	196K	18K
Area (Cells + Memories)	0.461 mm^2	0.053 mm^2

Table 6.2: Comparison of the two Raven-4 processors [68].

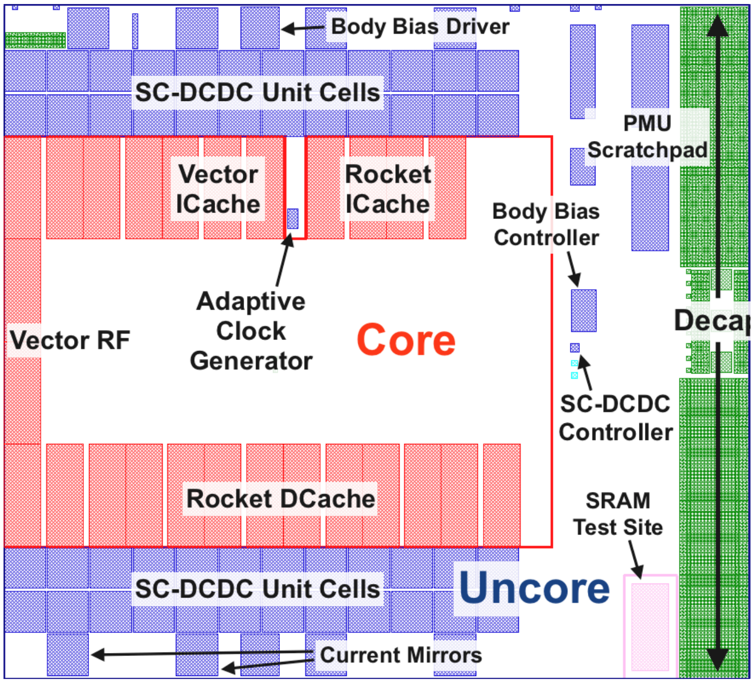


Figure 6.16: Annotated floorplan of the Raven-4 testchip [68] (© 2017 IEEE).

resolution and fast response allow adaptive body bias to be incorporated into power management techniques. Because the core voltage waveform is difficult to accurately observe off-chip due to parasitics on the measurement path, a waveform measurement circuit was implemented that uses a lightweight sampling approach to statistically reconstruct the waveform [82]. This approach allows core power consumption to be measured with high accuracy via numerical integration. As the current load of the application core can vary over time and has a limited range, a programmable current-mirror load connected to the core voltage domain was added to allow straightforward characterization and measurement of the SS-SC converters and power monitoring circuitry. The core clock and SS-SC toggle clock are also connected to output drivers for direct observation.

Figure 6.16 shows the floorplan of the SoC. The design is partitioned into two voltage ar-

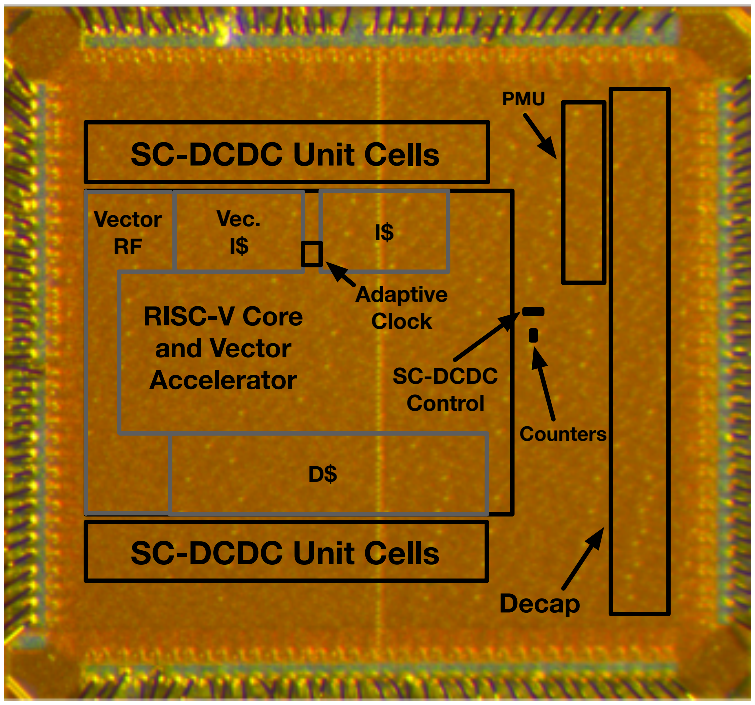


Figure 6.17: Annotated die micrograph of the Raven-4 testchip [135] (© 2016 IEEE).

eas, with the core voltage area supplied by the SS-SC converters placed centrally. Numerous additional voltages and clocks are defined to supply both the core and the various analog and mixed-signal blocks that make up the SoC. To reduce core insertion delay, the clock generator itself was placed near the center of the core area, and a "peninsula" of the uncore voltage domain was extended to allow the routing of control signals from the block to the top level of the design hierarchy. The location of the core clock multiplexer, which allowed the selection of different core clock sources for test, was specified explicitly and placed near the center of the core area. These improvements combined to reduce core insertion delay by several hundred picoseconds. An annotated die photo of the testchip, which was fabricated in 28 nm UTBB FD-SOI, is shown in Figure 6.17.

6.2.2 Measurement Results

The Raven-4 testchip was wirebonded chip-on-board (see Figure 6.18) and connected to an FPGA controller in a similar manner to Raven-3. Table 6.3 summarizes key features and measurement results.

Figure 6.19 shows measured waveforms of the core voltage and core clock during a transition between voltage modes. The core clock was measured via the output driver, which was not present on Raven-3. The core clock adapts automatically to the voltage ripples within each mode and to the large voltage transition between modes. Table 6.4 shows the

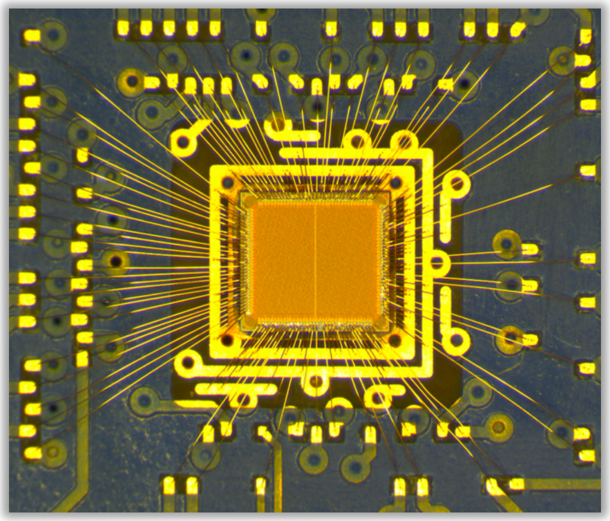


Figure 6.18: The Raven-4 die wirebonded to the daughterboard.

Technology	28 nm FDSOI
Die Area	1665 µm × 1818 µm (3.03 mm^2)
Core Area	895 µm × 1193 µm (1.07 mm^2)
Standard Cells	568K
Converter Area	48 × 90 µm × 90 µm (0.39 mm^2)
Core Voltage	0.48 V to 1 V (bypass mode)
Core Power	1.2 mW to 231 mW (bypass mode)
Core Frequency	20 MHz to 797 MHz (bypass mode)
Peak Energy Efficiency	41.8 GFLOPS/W (1/2 1V mode)
Conversion Efficiency	82-89%
AVS Transition Time	<1 µs
Peak AVS Energy Savings	39.8%

Table 6.3: A summary of key results and details from the Raven-4 testchip [68].

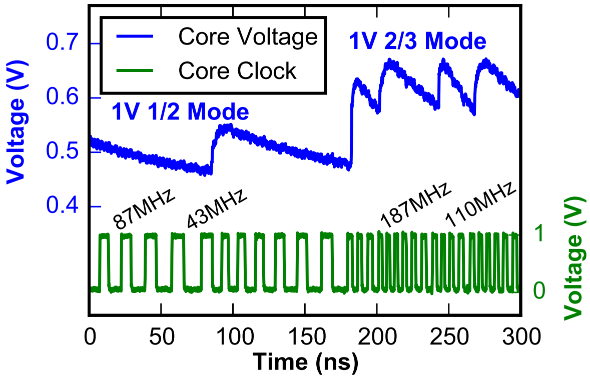


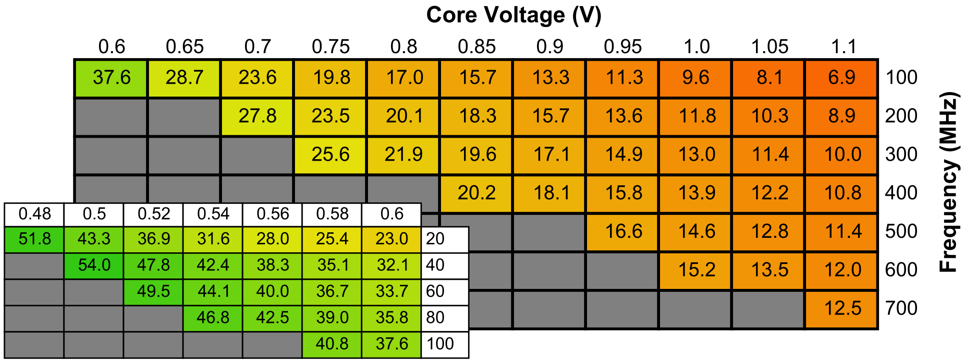
Figure 6.19: Oscilloscope traces of the core voltage and clock during an SS-SC mode transition [68] (© 2017 IEEE).

DC-DC Mode	Efficiency (Adaptive Clock)	Efficiency (Fixed Clock)
1/2 1.8V	88.7%	76.6%
2/3 1.0V	85.0%	75.4%
1/2 1.0V	81.6%	61.9%

Table 6.4: Measured system conversion efficiencies achieved by the Raven-4 system [68].

peak system conversion efficiency of each of the three switching SS-SC modes. The adaptive clock is tuned at each voltage setting by sweeping the settings of its replica delay path and choosing the fastest setting that still results in correct core functionality. The adaptive clocking system provides a large improvement in system conversion efficiency because the core is able to operate at a higher average frequency as the supply voltage ripples, reducing the amount of energy required to complete the same amount of work. When supplied by the SS-SC converter, the processor achieves a peak energy efficiency of 41.8 double-precision GFLOPS/W running an FMA microbenchmark on the vector coprocessor in 1/2 1 V mode. The processor is able to boot Linux and run user programs while powered by the rippling supply voltage and adaptive clock.

Figure 6.20 shows the processor functionality across a wide range of voltages and frequencies. The SS-SC converter is placed into bypass mode for characterization, allowing the measurement of processor performance under fixed voltage and frequency. The best energy efficiency in bypass mode of 54.0 double-precision GFLOPS/W is achieved at 500 mV and 40 MHz. Figure 6.21 shows the best frequency achievable at each operating point and the total energy consumed by a fixed-duration matrix-multiply benchmark at that operating point. The application of FBB increases performance but results in higher leakage power. The FBB voltage that achieves minimum energy depends on the proportion of switching power to leakage power and is therefore benchmark-dependent. Figure 6.22 shows that the energy of more

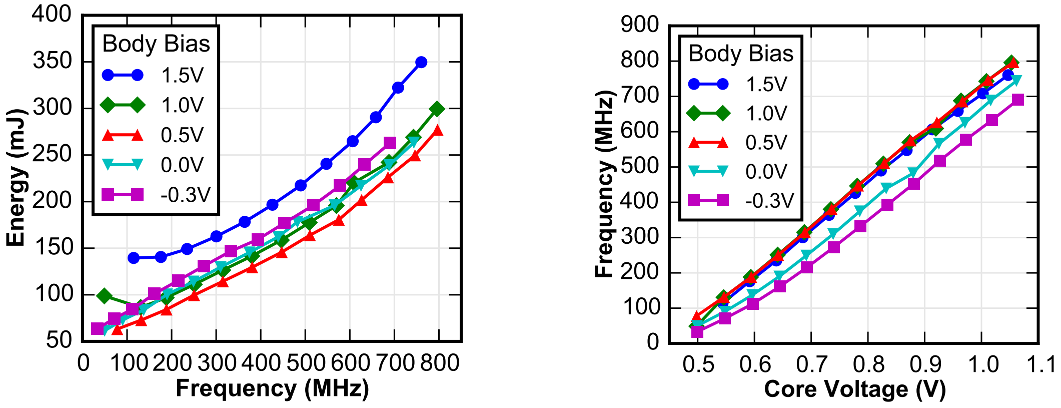


Core Voltage (V)

Frequency (MHz)	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95	1.0	1.05	1.1
100	37.6	28.7	23.6	19.8	17.0	15.7	13.3	11.3	9.6	8.1	6.9
200			27.8	23.5	20.1	18.3	15.7	13.6	11.8	10.3	8.9
300				25.6	21.9	19.6	17.1	14.9	13.0	11.4	10.0
400						20.2	18.1	15.8	13.9	12.2	10.8
500								16.6	14.6	12.8	11.4
600									15.2	13.5	12.0
700											12.5

	0.48	0.5	0.52	0.54	0.56	0.58	0.6
20	51.8	43.3	36.9	31.6	28.0	25.4	23.0
40		54.0	47.8	42.4	38.3	35.1	32.1
60			49.5	44.1	40.0	36.7	33.7
80				46.8	42.5	39.0	35.8
100						40.8	37.6

Figure 6.20: A shmoo chart showing processor performance while executing a matrix-multiply benchmark under a wide range of operating modes [68] (© 2017 IEEE). The number in each box represents the energy efficiency of the application core as measured in double-precision GFLOPS/W.



(a) (b)

Figure 6.21: Plots showing the effects of FBB on core frequency and energy [68] (© 2017 IEEE).

computationally intensive benchmarks is minimized at higher body-bias settings.

Figure 6.23 compares the generated frequency of the free-running adaptive clock generator with the design from Raven-3. Both designs reliably generate a core clock that can supply the core area while a workload is executed on the application core. The two designs track voltage similarly at the slowest delay settings, but the tracking varies at the fastest setting because the free-running design oscillates entirely in the variable-voltage domain, while part of the timing loop in the edge-selecting design is level-shifted to the fixed 1 V supply. The free-running generator achieves the same functionality as the adaptive clocking scheme from Raven-3 while simplifying the design and reducing area and power overhead.

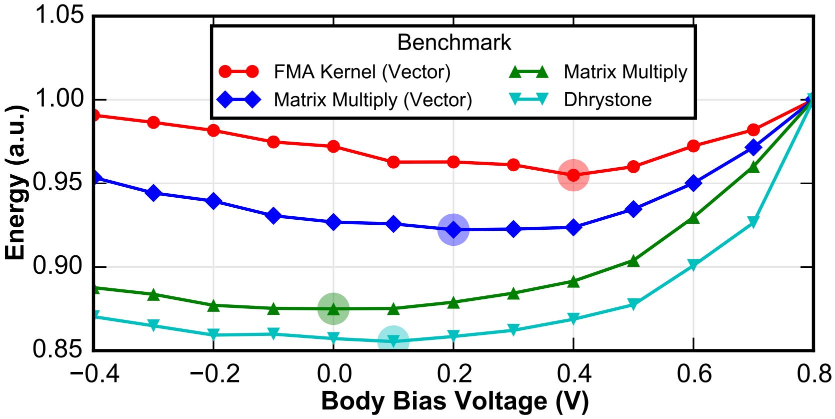


Figure 6.22: The effect of FBB on different benchmarks with a supply voltage of 0.6V (bypass mode) [135] (© 2016 IEEE). The total energy consumed by each benchmark has been normalized so the relative effects of body bias can be compared. The minimum-energy point is highlighted for each benchmark.

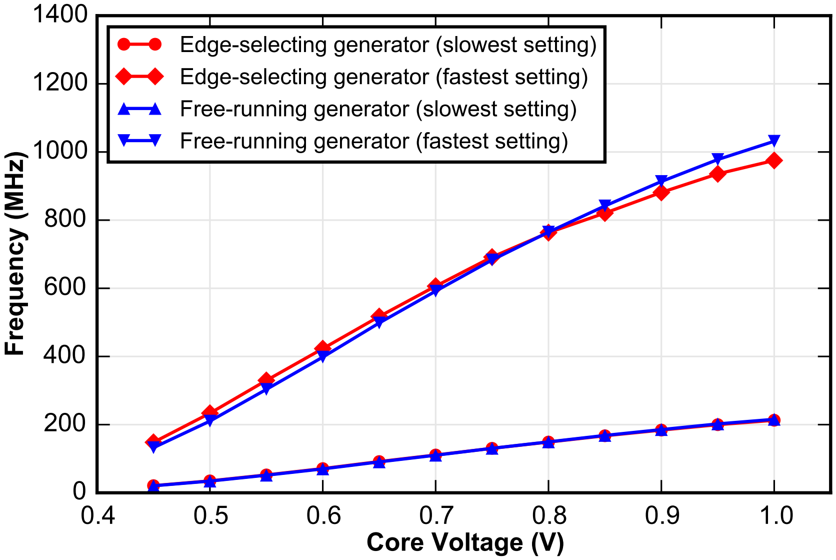


Figure 6.23: Measured comparison of the two clock generators implemented in the Raven-4 testchip [68] (© 2017 IEEE).

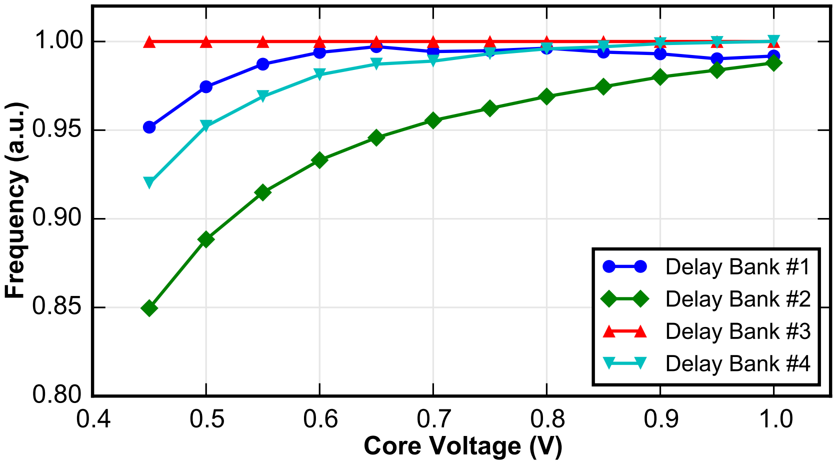


Figure 6.24: Measurement results showing the relative difference in voltage-dependent frequency behavior of the four delay banks in the adaptive clock generator [68] (© 2017 IEEE).

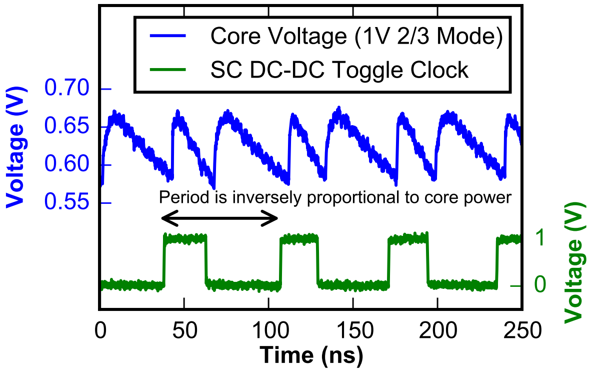


Figure 6.25: Oscilloscope traces showing the rippling core supply voltage and the SS-SC toggle clock [68] (© 2017 IEEE).

Figure 6.24 compares the voltage-dependent delay characteristics of the four delay banks, normalized to the delay of the custom buffer cells in Bank 3. The result for each bank was measured by recording the frequency of the generated clock after selecting the maximum delay through that bank and the minimum delay through the remaining banks. The cells with small pMOS/nMOS ratios and larger gate lengths have larger delays at lower voltages. The wide variation in voltage-dependent delays between the different delay banks (up to 18% at 0.45 V) validates the need for multiple different standard cells to achieve accurate critical path tracking.

Figure 6.25 shows the measured core voltage and SS-SC toggle clock, and Figure 6.26 compares the core power measured by the bench equipment with the SS-SC toggle frequency

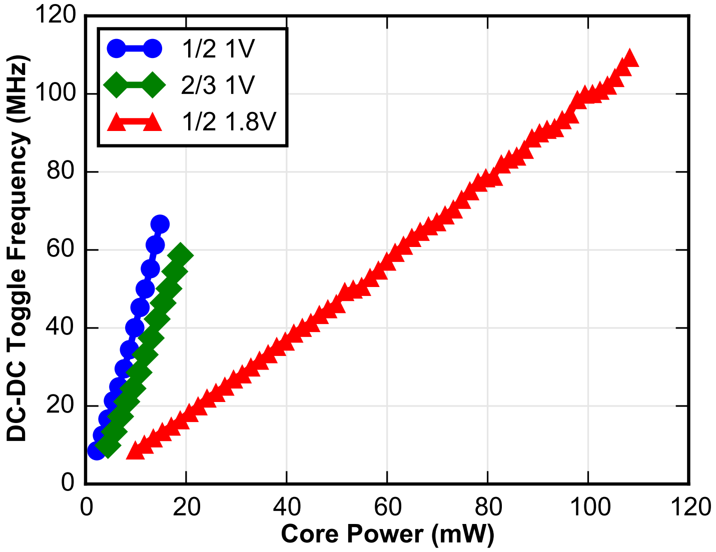


Figure 6.26: Measurement results showing the correlation between SS-SC toggle frequency and measured core power [68] (© 2017 IEEE).

measured by the integrated counter. The correlation is monotonic and approximately linear for each of the three conversion modes, confirming the practicality of using this toggle frequency to estimate core power and system load.

The programmable PMU allows the implementation of a wide variety of power management algorithms to improve energy efficiency. Several different experiments demonstrate the flexibility of the system in implementing common energy-saving techniques. In one experiment, dithering between two voltages to achieve an arbitrary target core frequency is implemented. To implement this algorithm, the PMU first calibrates the system by using the core clock counter to measure the average operating frequency at each voltage mode. Then a target frequency is provided to the PMU, which polls the core clock counter and dithers the voltage setting to achieve the target frequency in aggregate. The results of the experiment are shown in Figure 6.27. Without dithering, the processor would need to operate only in the higher mode to guarantee that the performance target is met, which would consume up to 40% more energy than the dithered approach.

The choice of hopping frequency presents a tradeoff between increased fidelity to the target effective frequency and the more frequent occurrence of transition overheads, which can increase energy consumption. In the testchip SS-SC implementation, the energy cost associated with transitions between voltage modes is small because the processor continues to operate as the clock frequency adjusts during the mode transition. In high-to-low mode transitions, no charge is wasted, but in some low-to-high transitions, the flying capacitance

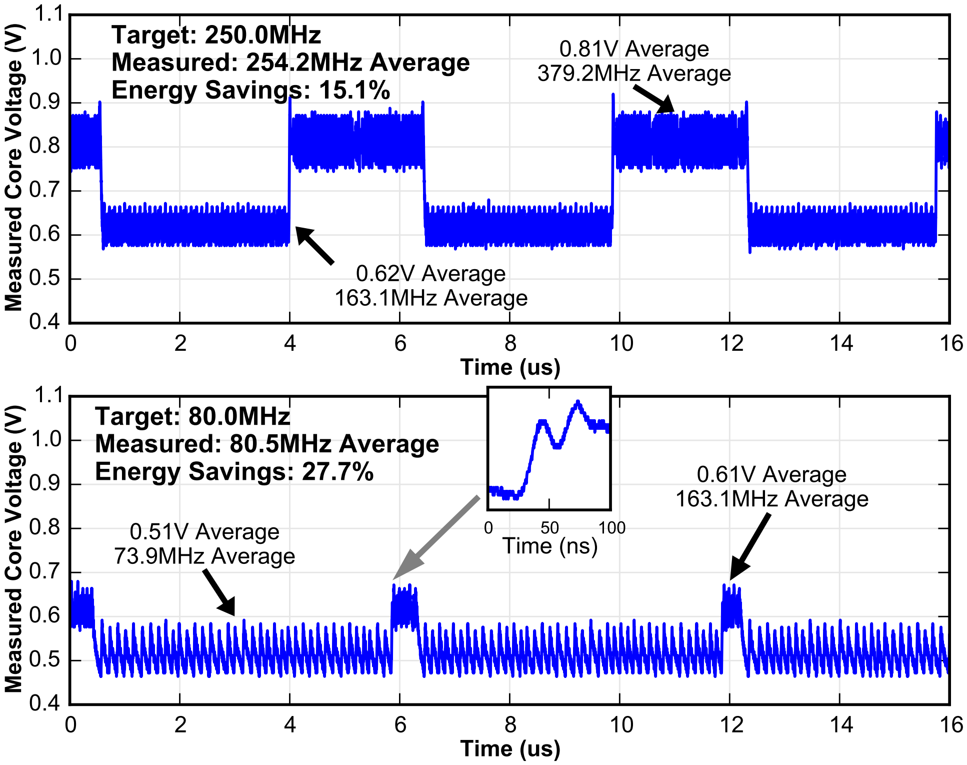


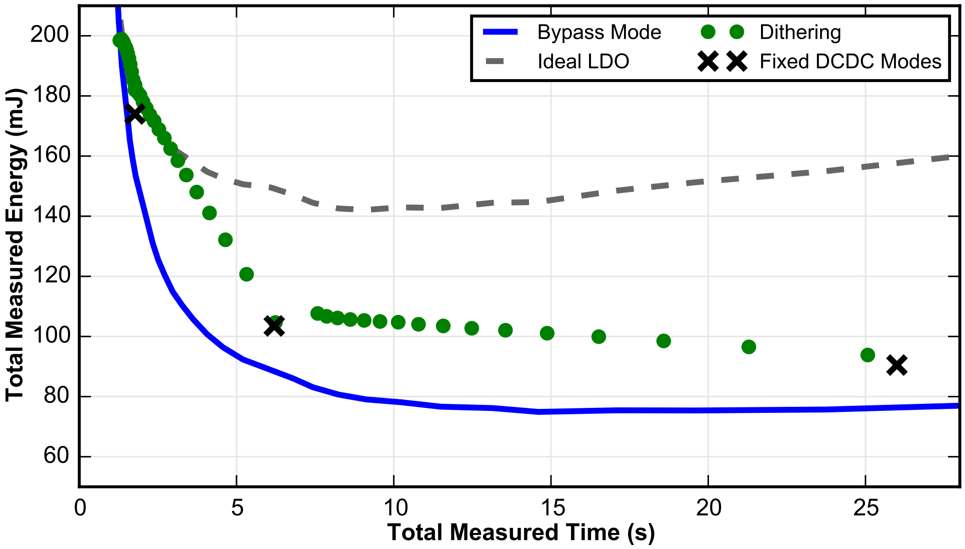
Figure 6.27: Oscilloscope traces showing the core voltage as the frequency hopping algorithm is applied with two different frequency targets [68] (© 2017 IEEE).

is charged to 1 V, consuming wasted energy:

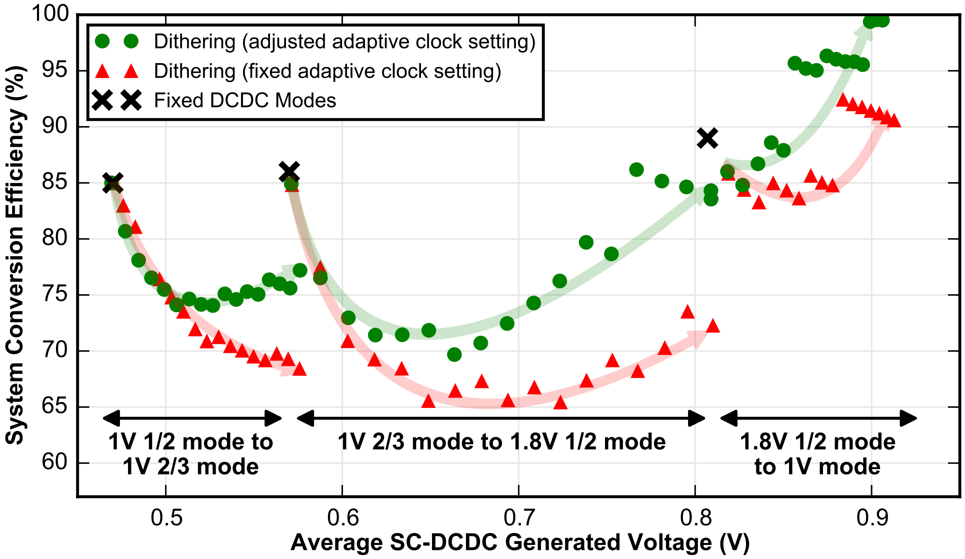
$$E_{loss} = \frac{1}{2}C_{fly}\Delta V^2 \tag{6.1}$$

In the worst-case transition from 1 V 2/3 mode to 1.8 V 1/2 mode, the flying capacitance is charged from roughly 0.33 V, resulting in an E_{loss} of roughly 1 nJ. At a typical core operating power of 50 mW, this loss is equivalent to the energy consumed by just 20 ns of normal operation. Accordingly, a hopping frequency of approximately 6 µs was chosen. This frequency is still quite fast, but is slow enough that transition energy costs can be neglected.

As noted in Section 2.1.3, voltage dithering can suffer reduced conversion efficiency compared to continuous regulation. This effect is explored in Figure 6.28. Each measured point in the figures represents the completion of a fixed-length matrix-multiply benchmark. The dithering algorithm enables a wide operating range for the core, bounded only by the lowest and highest voltage settings of the SS-SC converter. Figure 6.28a shows measurement results of system conversion efficiency, similar to the measured results of Raven-3 as shown in Figure 6.12. Instead of just three conversion points, however, the interpolated time-energy points measured from dithering between each of the neighboring voltages at a fixed ratio are



(a)



(b)

Figure 6.28: Plots showing the effect of voltage dithering on system conversion efficiency [68] (© 2017 IEEE). (a) compares the energy cost of dithering to the bypass mode baseline (in blue), which represents 100% efficient regulation. The dithering operating points linearly interpolate completion time and energy between the fixed SS-SC modes. (b) shows the measured system conversion efficiencies under voltage dithering. The results in green show the benefit of re-tuning the replica timing circuit in the adaptive clock generator after each SS-SC mode transition.

also shown. Figure 6.28b quantifies the effects of dithering on system conversion efficiency. Two different dithering programs were run on the PMU. The first program simply switches the voltage mode setting of the SS-SC converters, without changing any other system settings; the delay settings of the replica paths in the adaptive clock generator were tuned to the best setting that could function across both operating modes. The results are shown by the red points in the figure. Because the best setting of the replica paths changes according to operating mode, the conversion efficiencies of this approach are less than optimal for part of the dithering range. The second program switches both the voltage mode and the delay settings of the replica paths according to a pre-characterization of the best adaptive clock setting associated with each voltage mode. This program is able to speed the generated clock at the higher voltage settings, leading to higher conversion efficiencies. In all, the efficiencies of the second program range from 70% to 100%, depending on the voltage mode and dithering ratio. Conversion efficiencies while dithering are not as high as the efficiencies of the fixed operating modes in Table 6.4 because the external voltage reference used by the comparator cannot be tuned for a particular SS-SC mode. This implies that in some cases it is more efficient to operate at a single mode than to dither, even if the performance target is somewhat exceeded by the average operating frequency at the fixed mode.

A second experiment demonstrates power envelope tracking, a common requirement for systems that must operate within a user-specified power budget. Figure 6.29 shows the results of a power management algorithm executed on the PMU that maximizes core performance within a user-specified power budget. The power management program polls an externally writeable control register that stores the absolute power limit for the program. The PMU core then monitors the SS-SC toggle counter to continuously estimate core power using the quadratic model described in [82] with pre-characterized coefficients. If the estimated core power is above the specified limit, core frequency is decreased, and if it is below the limit, core frequency is increased. In this way, the best possible performance is automatically obtained while the user-specified power budget is respected.

The PMU can also use the integrated counters to coordinate fine-grained adaptive voltage scaling (AVS) on-chip without any explicit guidance from the programs running on the processor. In the algorithm used in this experiment, core power is used as a marker of program phase. When core power is higher, the core is likely executing a compute-intensive program region, and a high voltage is best suited to a race-to-halt strategy. When the core power is lower, the core is likely waiting for off-chip communication in a memory-bound program region, and energy can be saved with minimal performance impact by reducing the voltage. In this experiment, the application core runs a synthetic benchmark that alternates between the compute-intensive and idle phases at a timescale of tens of microseconds. Figure 6.30 shows the core voltage measured during the execution of the benchmark and the AVS power-management algorithm. The algorithm switches the core voltage between the 1.8 V 1/2 mode and the 1 V 2/3 mode, actuated by core power estimates determined by continuously polling the SS-SC toggle counter. When the core voltage is high and the toggle rate drops below a threshold, this corresponds to an idle program period, so the PMU reduces the core voltage to save energy. When the core voltage is low and the toggle rate

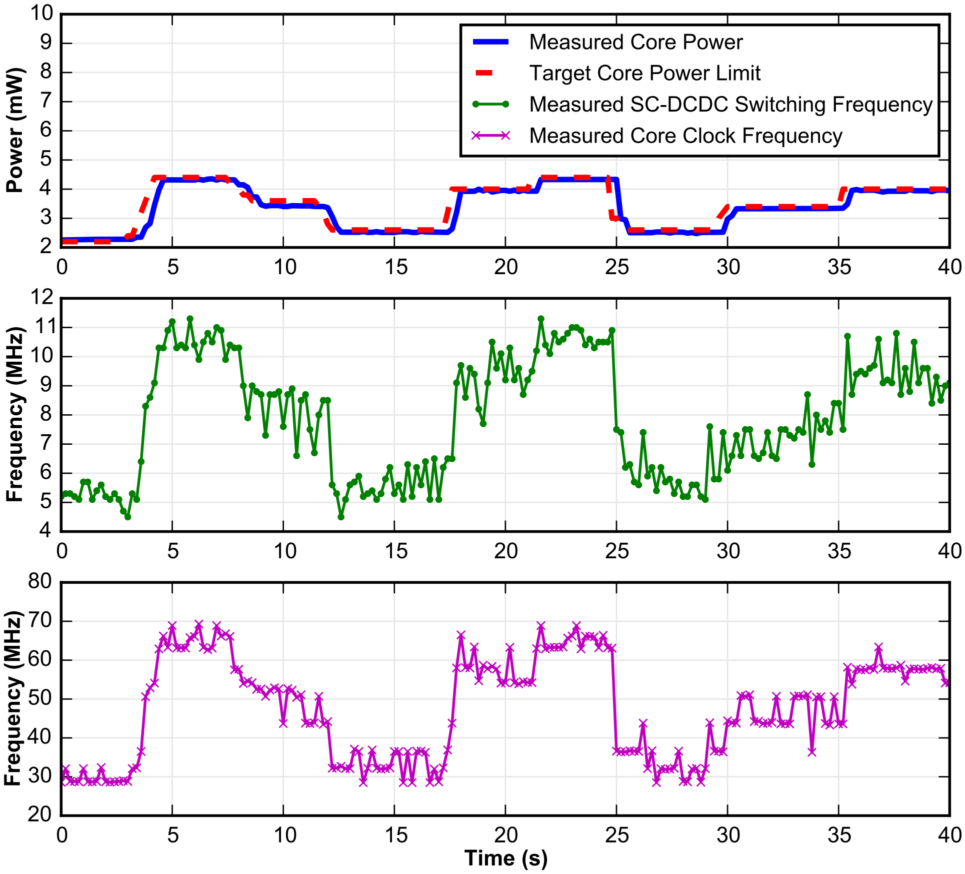


Figure 6.29: Measurement results showing a power envelope tracking program executing on the PMU [68] (© 2017 IEEE). The frequency of the core is adjusted in response to measured changes in core power so that the core operates as quickly as possible without exceeding a time-varying, user-specified power budget.

exceeds a threshold, the workload has increased and the PMU increases the core voltage. The system is able to detect changes in workload in less than 1 µs and adjust the core voltage in response. Without integrated voltage regulators and power management, the system would not be able to respond within the timescales of the workload variation. The results of the power-management algorithm are therefore compared against continuous operation in the higher voltage mode, which would otherwise be required to meet the same performance target. The power-management algorithm reduces the energy consumed by 39.8%, and the fast response incurs negligible (<0.2%) performance penalty compared with this baseline because the fast response minimizes the time spent in the lower voltage mode during a compute-bound region. This experiment demonstrates the efficacy of fine-grained AVS at improving energy efficiency with fast workload tracking.

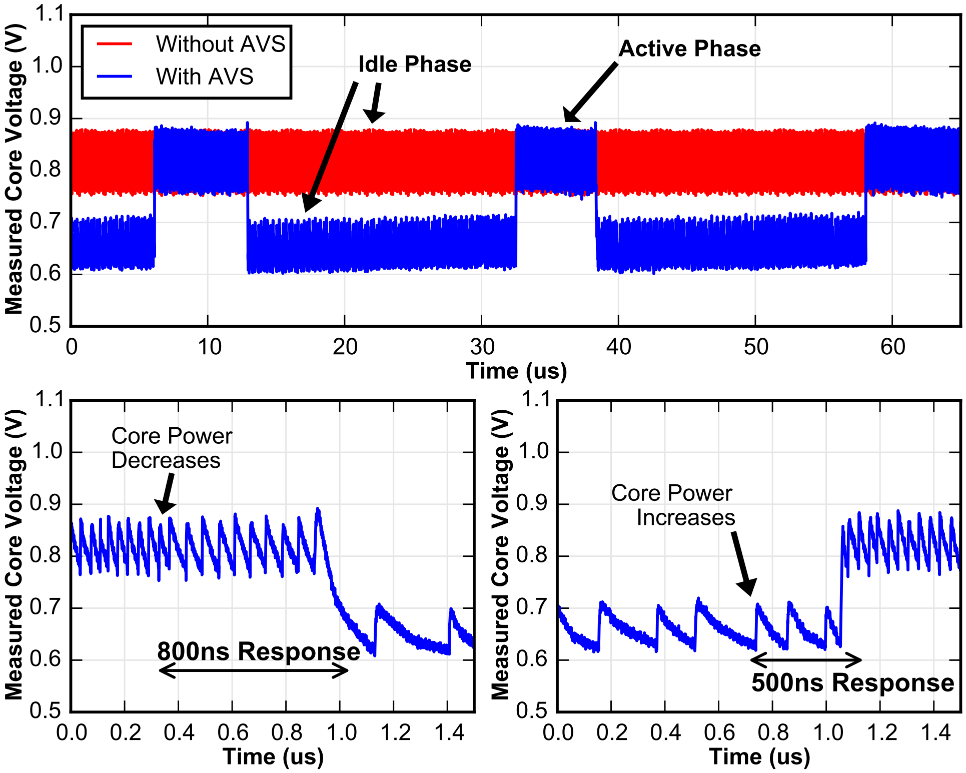


Figure 6.30: Oscilloscope traces showing an FG-AVS algorithm running on the PMU [68] (© 2017 IEEE).

6.3 Hurricane-1: AVS At Scale

The Hurricane-1 testchip scales up many of the design concepts of the Raven systems to demonstrate FG-AVS in a larger, more realistic SoC. The system is comprised of two homogenous application processors with a shared, coherent memory system. Each application core has its own integrated voltage regulator and clock generator and operates as an independent voltage domain. The memory system is improved with the addition of high-speed serial links for off-chip communication, and distributed thermal sensors make possible temperature-based power management. A block diagram of the Hurricane-1 system is shown in Figure 6.31.

6.3.1 System Design and Implementation

Each Hurricane-1 core implements an updated version of the five-stage Rocket processor targeting a draft of version 2.1 of the RV64G user specification and version 1.7 of the privileged specification. The size of the Rocket instruction cache is increased to 32 KiB, and the data cache is dedicated to Rocket instead of being shared with the vector coprocessor.

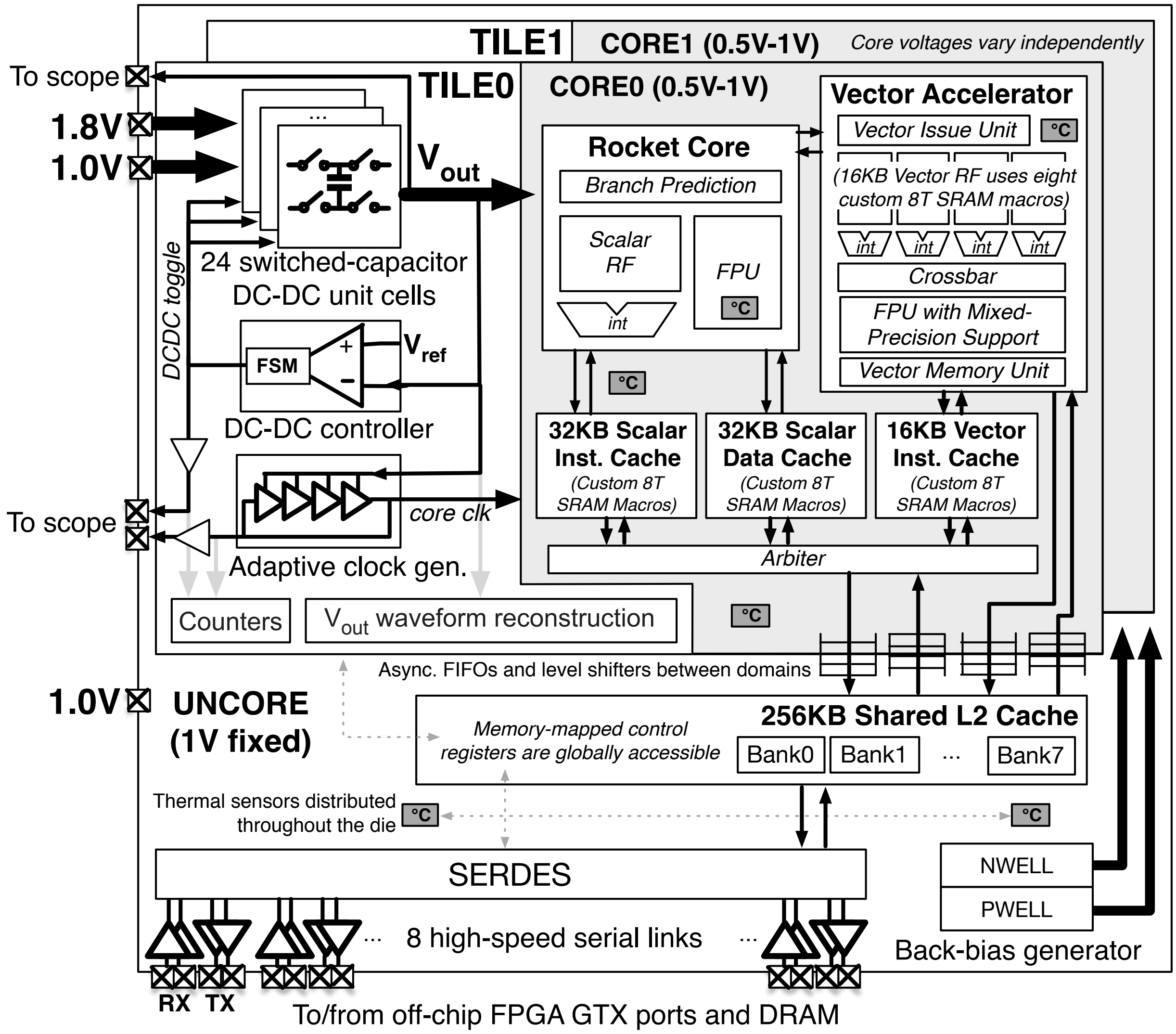

TILE1
CORE1 (0.5V-1V)
Core voltages vary independently
TILE0
CORE0 (0.5V-1V)
To scope
1.8V
1.0V
Vout
24 switched-capacitor DC-DC unit cells
DCDC toggle
FSM
Vref
DC-DC controller
core clk
Adaptive clock gen.
To scope
Counters
Vout waveform reconstruction
Rocket Core
Branch Prediction
Scalar RF
FPU
int
°C
Vector Accelerator
Vector Issue Unit
(16KB Vector RF uses eight custom 8T SRAM macros)
int
Crossbar
FPU with Mixed-Precision Support
Vector Memory Unit
32KB Scalar Inst. Cache (Custom 8T SRAM Macros)
32KB Scalar Data Cache (Custom 8T SRAM Macros)
16KB Vector Inst. Cache (Custom 8T SRAM Macros)
Arbiter
Async. FIFOs and level shifters between domains
1.0V
UNCORE (1V fixed)
Memory-mapped control registers are globally accessible
256KB Shared L2 Cache
Bank0
Bank1
Bank7
Thermal sensors distributed throughout the die
SERDES
NWELL
PWELL
Back-bias generator
8 high-speed serial links
RX
TX
To/from off-chip FPGA GTX ports and DRAM

Figure 6.31: A block diagram of the Hurricane-1 testchip.

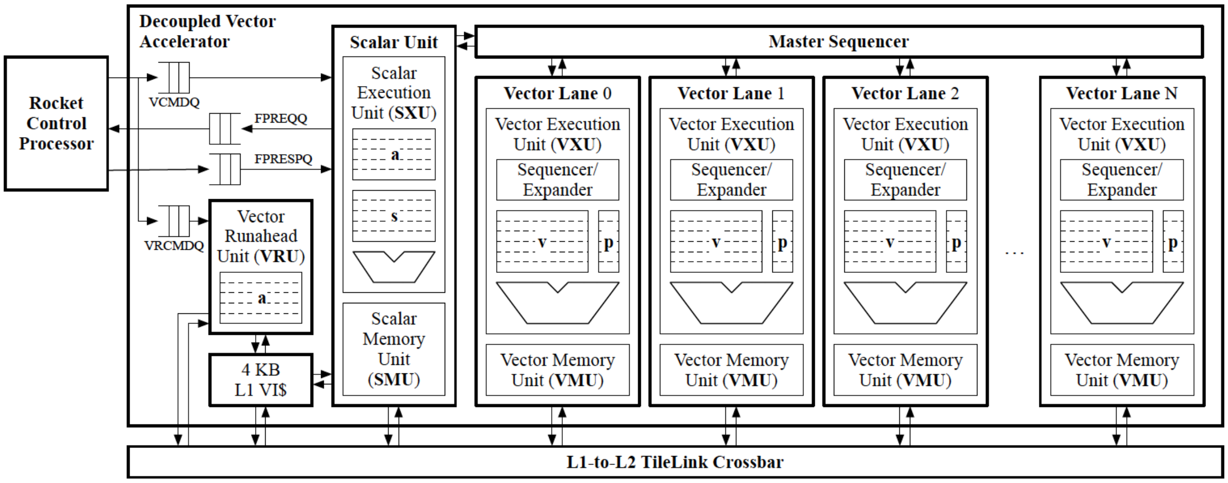


Figure 6.32: A block diagram of the reimplemented Hwacha microarchitecture [137].

The floating-point pipeline depth is increased to four to better balance the timing with the memory pipeline stages. Several improvements have been made to the Hwacha vector accelerator that is also part of each core. The machine has been redesigned to target the Hwacha v4 ISA [136] as shown in Figure 6.32. The new implementation supports optimized mixed-precision data packing and instructions that increase efficiency for certain workloads. Full predication supports arbitrary control flow, allowing Hwacha to be targeted as a backend of an OpenCL compiler for improved programmability. The private vector instruction cache has been increased in size to 16 KiB. Vector loads and stores are serviced by a dedicated port to the outer memory system, increasing memory bandwidth. Additional floating-point hardware resources have been added, increasing the peak performance of the system to 8 double-precision or 16 single-precision FLOPs per cycle, and the floating point units in both Rocket and Hwacha are updated to support floating-point division and square roots. All core SRAMs are implemented using the custom 8T macros first deployed in Raven-3, but the aspect ratio of the L1 cache memories has been adjusted to improve the utilization of these macros, which are only available in a single size. Each core is supplied via the integrated voltage regulator first implemented in Raven-3 with 24 SS-SC unit cells per core. Clocks are generated per-core by adaptive clock generators as implemented in Raven-4.

The two variable-voltage core domains communicate with the fixed-voltage uncore domain via level shifters and standard bisynchronous FIFOs. The uncore contains a 256 KiB, 8-bank L2 cache that serves as a shared, coherent backing store for both cores. Counters track load and store misses from the L2 cache, providing useful information for FG-AVS feedback. The system memory map has also been unified such that all system control registers and counters can be read by either core via a simple memory access. This allows the two cores to communicate, execute shared-memory programs, and perform per-core or global power management. While a dedicated power management controller is not included in the design,

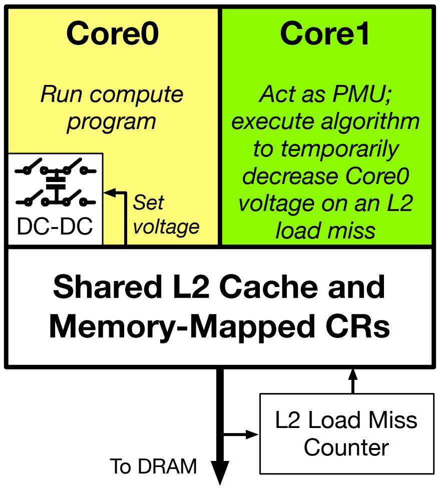

Core0
Run compute program
DC-DC
Set voltage
Core1
Act as PMU; execute algorithm to temporarily decrease Core0 voltage on an L2 load miss
Shared L2 Cache and Memory-Mapped CRs
L2 Load Miss Counter
To DRAM

Figure 6.33: A feedback diagram showing how FG-AVS algorithms can be implemented on the Hurricane-1 testchip.

either core can act as a PMU while application programs execute on the other core as shown in Figure 6.33.

In addition to sending memory requests over the slow HTIF link, the system also implements eight high-speed serial lanes so that higher bandwidth can be achieved by the memory system. The links implement a tunneled AXI-over-serial interface that can communicate with a deserializer on an FPGA so that requests can be handled by the FPGA memory system. The physical implementation of the TX and RX blocks is compatible with the GTX interface used by the FPGA. Each link is designed to operate at up to 10 Gbps DDR, for a peak bandwidth of 80 Gbps. The integrated memory system can be configured at boot time to direct memory traffic over the slow HTIF link, a single SERDES lane, or all eight lanes. In the latter case, each of the eight L2 bank sends requests to a dedicated serial link.

The uncore domain also contains additional IP blocks to round out system functionality. The integrated body bias generator, power monitors, and waveform reconstruction sensors are reimplemented from the Raven-4 design. (Both cores share the same deep n-well and so their body bias voltage is adjusted together.) An all-digital bang-bang PLL can generate a wide range of clock frequencies from a fixed reference. Because temperature can have a dramatic effect on device leakage and total system power, distributed thermal sensors were implemented that can measure the temperature at different locations on the die. The temperature probes consist of a simple nMOS-only ring oscillator for which both the supply and back bias voltage are provided by an LDO (see Figure 6.34). The frequency of the oscillator varies predictably with temperature, and after a one-point calibration procedure the sensors can measure local temperature for use in power management algorithms. Eleven of these sensors are implemented in total (four in each core and three in the uncore), but their simple design makes the area overhead negligible ($225\,\mu\text{m}^2$). Digital control signals

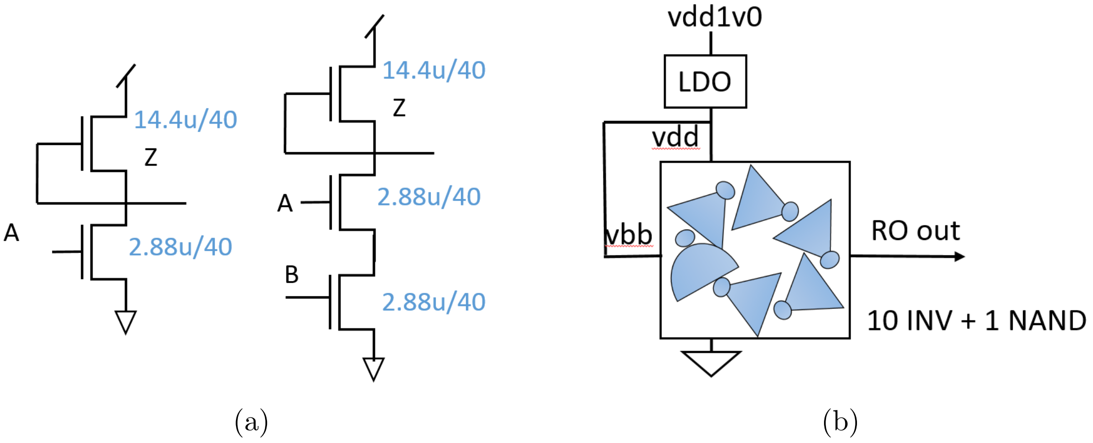


Figure 6.34: The distributed thermal sensor circuit. (a) shows the nMOS-only standard cells that make up the ring oscillator shown in (b).

and readout are distributed to each sensor from a central logic block that aggregates the data from each sensor and applies the calibration based on a lookup table. A much larger vendor-supplied temperature sensor IP block based on a more traditional band-gap reference is placed near one of the uncore sensors for a baseline comparison.

Figure 6.35 shows the floorplan of the SoC. The two cores are tiled in the upper left and upper right along with the associated per-core IP. The shared memory system and serial links are in the lower part of the chip, with the serial links distributed because each macro requires four data and two supply pads. The large chip necessitated the use of a hierarchical, multiply-instantiated-module place-and-route flow in which some parts of the design were hardened before integration at the top level. The core design was placed and routed only once, and then two copies were placed side-by-side in the top-level floorplan. The chip was fabricated in 28 nm UTBB FD-SOI. An annotated die micrograph is shown in Figure 6.36. The chip dimensions total 7.84 mm^2, with each core occupying 1.49 mm^2.

6.3.2 Measurement Results

The Hurricane-1 testchip was wirebonded chip-on-board in a similar manner to the Raven testchips. The FPGA system was upgraded to a Xilinx ZC706 development board. This board supports high-speed GTX connections for the integrated serial links and a high-pin-count FMC connection that allows all eight links to be transmitted through FMC to the FPGA. The daughterboard allows redirection of one set of TX and RX pairs to SMA connectors via 0 Ω resistors, to allow for better signal integrity in a single lane. The system is able to boot SMP Linux and run independent threads or a multithreaded program across both cores; it can also run each core at an independent voltage and frequency, demonstrating the ability of integrated voltage regulation to achieve fine-grained spatial AVS.

To explore the capabilities of fine-grained AVS in time, one core must act as the power

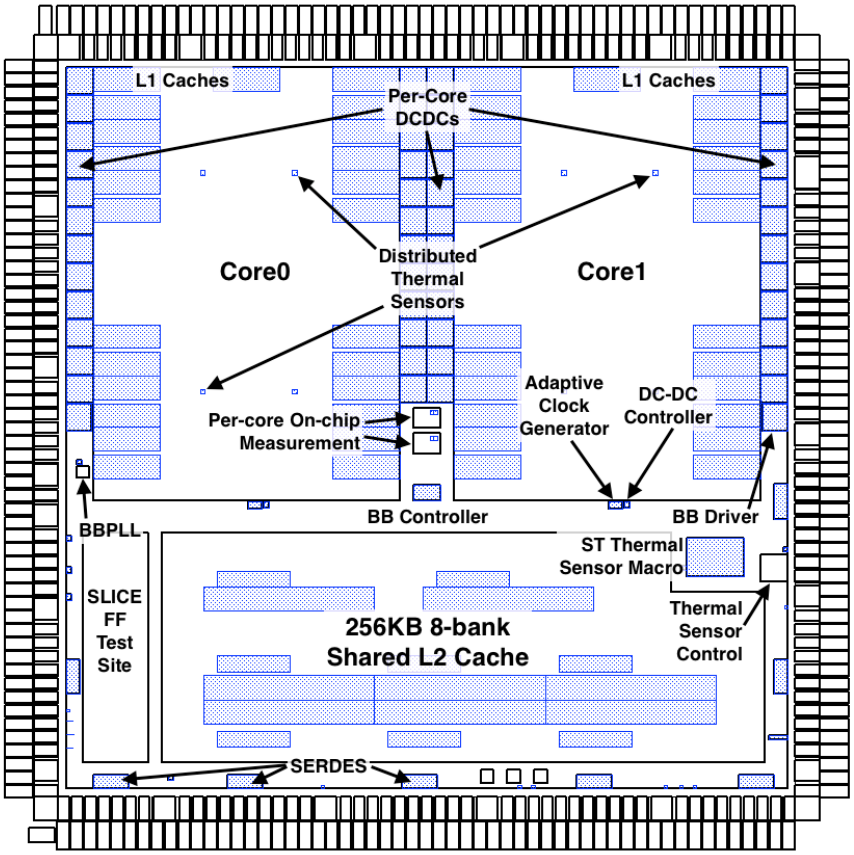


Figure 6.35: Annotated floorplan of the Hurricane-1 testchip.

management unit as described in Figure 6.33. To perform AVS experiments, Core 1 operates in fixed 1V mode and is clocked synchronously with the fixed-voltage uncore. Acting as a PMU, Core 1 polls the L2 load miss counters integrated into the uncore and responds by actuating a change in the Core 0 voltage via writes to the appropriate memory-mapped register. The software for both cores is compiled from C++ using the RISC-V toolchain managed by the Linux operating system. After booting Linux, the PMU program is scheduled onto Core 1 via the `taskset` command, while application programs are executed on Core 0. The memory system is configured to use the HTIF interface, with the uncore operating at 170MHz.

Unfortunately, several issues restrict the possible range of AVS experimentation in the Hurricane-1 testchip. Due to substantial IR drop in the DCDC power supplies, the cores do not operate robustly in the 1V 2/3 and 1V 1/2 modes, leaving only the 1.8V 1/2 mode and the 1V fixed mode as reliable targets for AVS. Furthermore, synchronization issues between Core 0 and the memory system prevent robust system operation when using the adaptive clock generator. As a workaround, frequency scaling is achieved by switching the Core 0 clock mux during operation. When Core 0 is operating in the 1V fixed mode, its clock mux

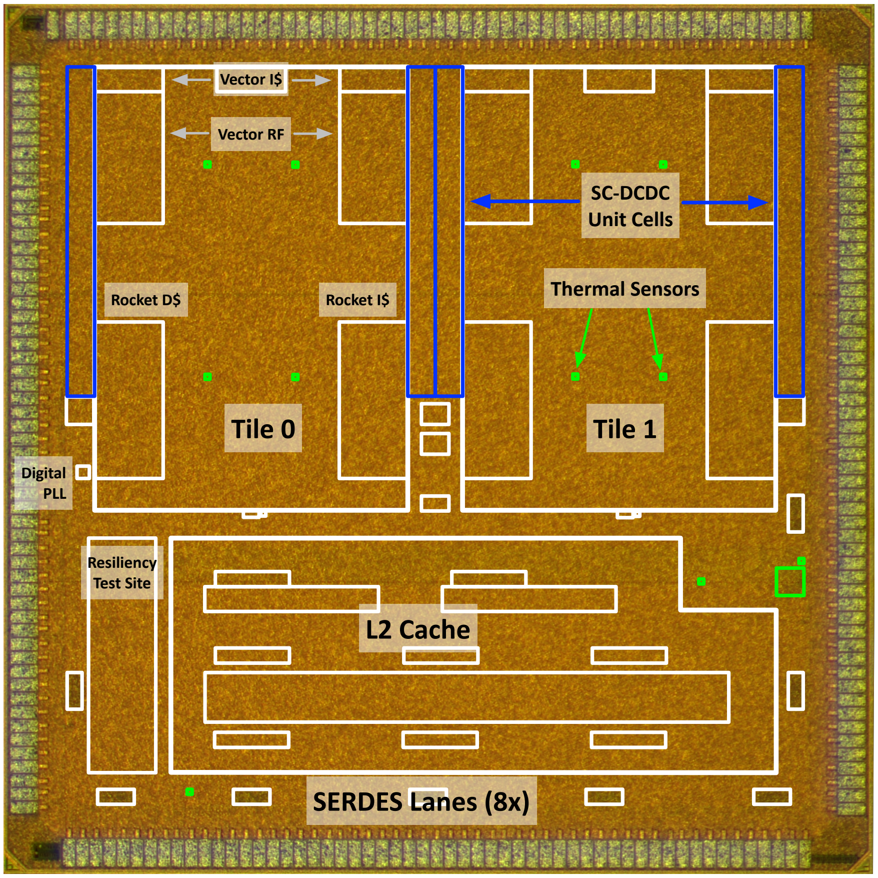


Figure 6.36: Annotated die micrograph of the Hurricane-1 testchip.

is set so that it operates synchronously with the uncore. When the voltage mode changes to the 1.8V 1/2 mode, the clock mux select is adjusted to use the divided-by-two version of the uncore clock. Because these two clock sources have a known, fixed phase relation, switching dynamically between them cannot result in short edges that could cause timing violations. These limitations and workarounds greatly reduce the possible energy savings of AVS in the system.

Figure 6.37 shows pseudocode representing two algorithms run on Core 1. Each algorithm polls the system L2 load miss counter and sets the operating condition of the application core based on whether L2 misses have been detected. Since Core 1 is operating only this small inner loop, L2 misses are assumed to originate from Core 0, and so indicate a period of idleness during which the voltage and frequency can be decreased to save energy with minimal impact on runtime. Algorithm A is event-driven, scaling voltage as soon as possible after an L2 miss is detected and then holding it low for some fixed duration. Algorithm B instead polls the counters, waits for a fixed interval, and then polls the counters again to determine if an L2 miss occurred during this interval. Each algorithm was run while Core0 executed the Perlbench program from the SPEC2006 benchmark suite [138].

```
for (;;) { // Algorithm A
  poll_new = read_scr(L2_LOAD_MISS_COUNTER);
  load_misses_in_interval = poll_new - poll_old;
  poll_old = poll_new;
  if ( load_misses_in_interval > 0 ) {
    set_operating_mode("0.9V");
    wait(n_cycles);
  } else {
    set_operating_mode("1V");
  }
}
```

```
for (;;) { // Algorithm B
  poll_old = read_scr(L2_LOAD_MISS_COUNTER);
  wait(n_cycles);
  poll_new = read_scr(L2_LOAD_MISS_COUNTER);
  load_misses_in_interval = poll_new - poll_old;
  if ( load_misses_in_interval > 0 ) {
    set_operating_mode("0.9V");
    wait(150);
  } else {
    set_operating_mode("1V");
  }
}
```

Figure 6.37: Pseudocode for two power-management algorithms run on Hurricane-1. The n_{cycles} variable, which specifies different interval durations in each algorithm, was swept to find the greatest energy savings for each algorithm.

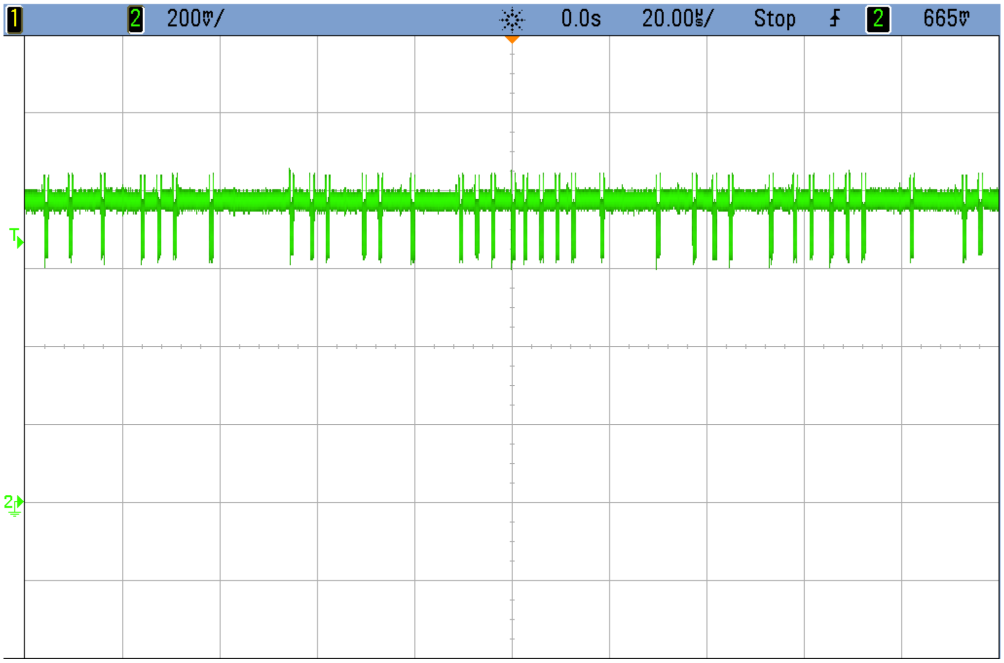


Figure 6.38: Measurement of the Core0 voltage during AVS algorithm execution.

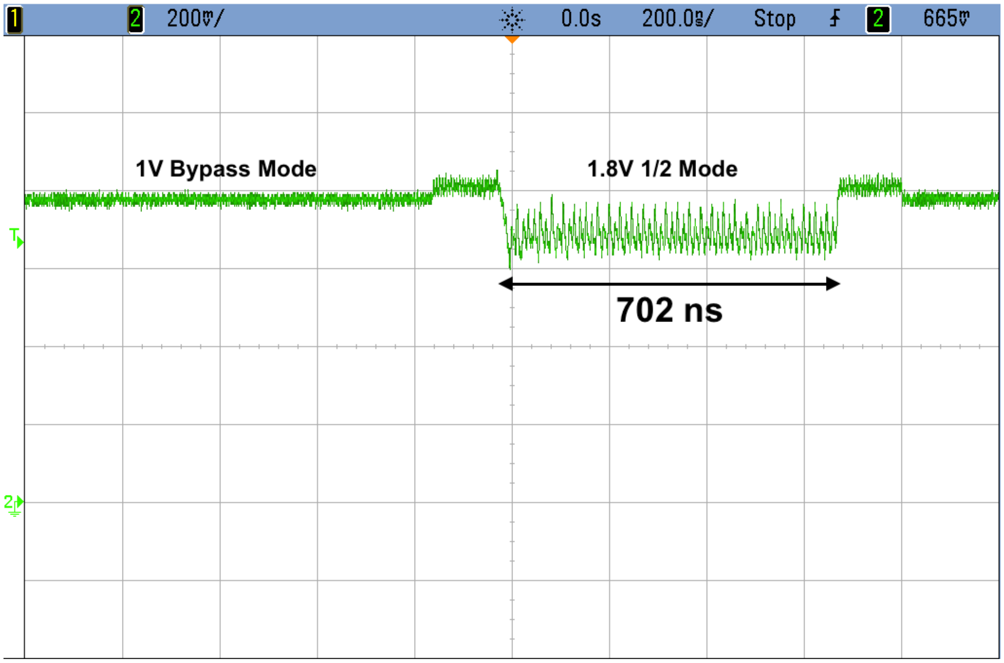


Figure 6.39: Measurement of the Core0 voltage after an L2 cache miss triggers voltage scaling.

Figures 6.38 and 6.39 show the measured Core0 voltage while executing Algorithm A with $n_{cycles} = 0$ so that the shortest possible response could be observed. In Figure 6.38, the PMU responds to multiple sequential L2 load misses by repeatedly lowering the operating voltage for a short time. Figure 6.39 shows a single voltage-scaling event. The overall response time of the feedback loop cannot be known, because the exact time of the L2 cache miss that triggers Core 0 idleness cannot be measured externally. However, the overall scaling event duration is just 702 ns, showing the ability of the PMU to execute multiple changes in operating mode in rapid succession.

Figure 6.40 and Tables 6.6 and 6.6 show the measured results of the AVS algorithms. Application core energy was measured by measuring program runtime and the voltage and current of the input DCDC supplies and subtracting the estimated power consumed by the PMU core. (Core 1 power cannot be measured directly because all of the voltage regulators on the chip share input supply rails.) Overall, Algorithm A is better able to save energy with minimal impact on runtime, while the energy savings of Algorithm B are achieved only as runtime increases. In the best case of Algorithm A compiled with $n_{cycles} = 300$, AVS is able to reduce application core energy by 6% relative to the baseline with a small (0.5%) impact on performance.

The energy savings of these AVS algorithms could be greatly improved with more robust chip functionality. Faster PMU and uncore operating frequencies would allow more responsive execution of the algorithms. Operating the core at lower voltage modes during idle

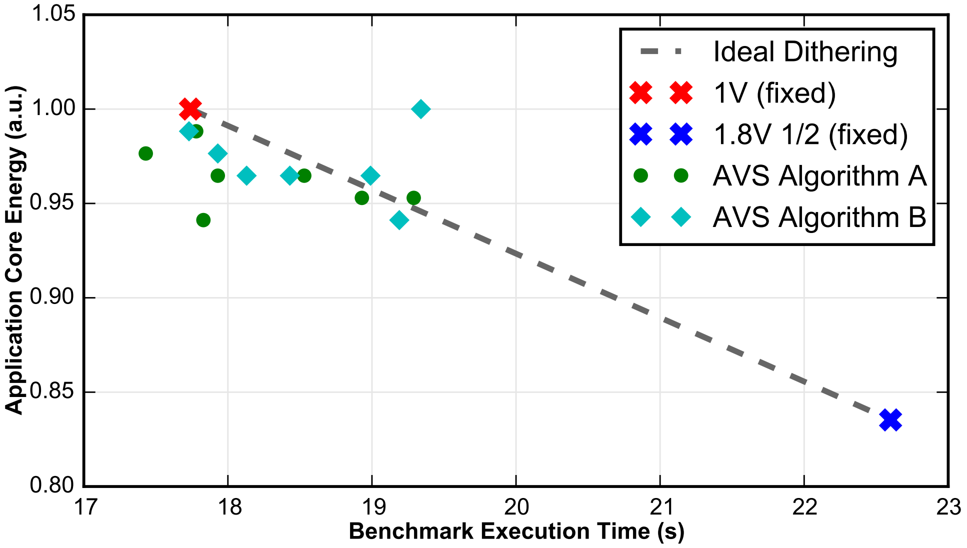


Figure 6.40: Measurement of a Core 0 voltage after an L2 cache miss triggers voltage scaling. Energy measurements are normalized to the fixed-1V operating mode.

n_{cycles}	Runtime (s)	Application Core Energy (a.u.)	Proportion of Runtime in Low-Voltage Mode
1V (no AVS)	17.74	1.00	0.0
0	17.43	0.98	0.026
30	17.78	0.99	0.041
150	17.93	0.97	0.101
300	17.83	0.94	0.159
600	18.53	0.97	0.207
1500	18.93	0.95	0.264
3000	19.29	0.95	0.340
1.8V 1/2 (no AVS)	22.60	0.84	1.0

Table 6.5: The effects of AVS Algorithm A on runtime and energy, as well as the proportion of runtime spent in the lower-voltage 1.8V 1/2 mode. Data from the fixed 1V and 1.8V 1/2 modes are included for comparison.

periods would save additional energy. The use of adaptive clock generation would greatly improve the conversion efficiency of the SS-SC converters, reducing the effective conversion losses that mitigate energy savings at lower voltage modes. Furthermore, the relative completion time of the benchmark at each fixed operating mode suggests that the core is idle for additional periods of time not captured by the L2 load miss counters. More robust counter instrumentation could allow for more sophisticated algorithms that better estimate

n_{cycles}	Runtime (s)	Application Core Energy (a.u.)	Proportion of Runtime in Low-Voltage Mode
1V (no AVS)	17.74	1.00	0.0
0	17.73	0.99	0.026
30	17.93	0.98	0.800
150	18.13	0.98	0.239
300	18.43	0.96	0.400
600	19.34	1.00	0.404
1500	18.99	0.96	0.550
3000	19.19	0.94	0.747
1.8V 1/2 (no AVS)	22.60	0.84	1.0

Table 6.6: The effects of AVS Algorithm B on runtime and energy, as well as the proportion of runtime spent in the lower-voltage 1.8V 1/2 mode.

core idleness. Nonetheless, Hurricane-1 demonstrates the potential utility of fine-grained AVS in time to save energy on realistic workloads without sacrificing performance.

6.4 Hurricane-2: Realizing Fine-Grained AVS

The Hurricane-2 testchip makes further improvements to compute and memory system capability while adding features to implement FG-AVS in time and space. Unlike previous designs, Hurricane-2 splits the application core itself into two independent voltage domains, one containing the Rocket core and one containing the vector accelerator. A DRAM controller and PHY are included for more realistic memory system performance. A dedicated PMU and a range of counters implemented throughout the core and memory system enable robust feedback and fast integrated power management. A block diagram of the Hurricane-2 system is shown in Figure 6.41.

6.4.1 System Design and Implementation

The Rocket voltage domain contains an updated version of the five-stage Rocket core that implements a draft of version 2.1 of the RISC-V RV64G user specification and version 1.9 of the RISC-V privileged specification. The Rocket processor interfaces with two different accelerators via the Rocket Custom Coprocessor (RoCC) interface. One accelerator, the Hwacha vector coprocessor, is located in its own voltage domain, allowing for its voltage and frequency to be adjusted independently of those of the Rocket domain when workloads vary between them. This RoCC interface therefore uses deeper bisynchronous queues and bidirectional level shifters. The second accelerator is a memory copy engine that has a dedicated port to the memory system in order to move data from one memory location to

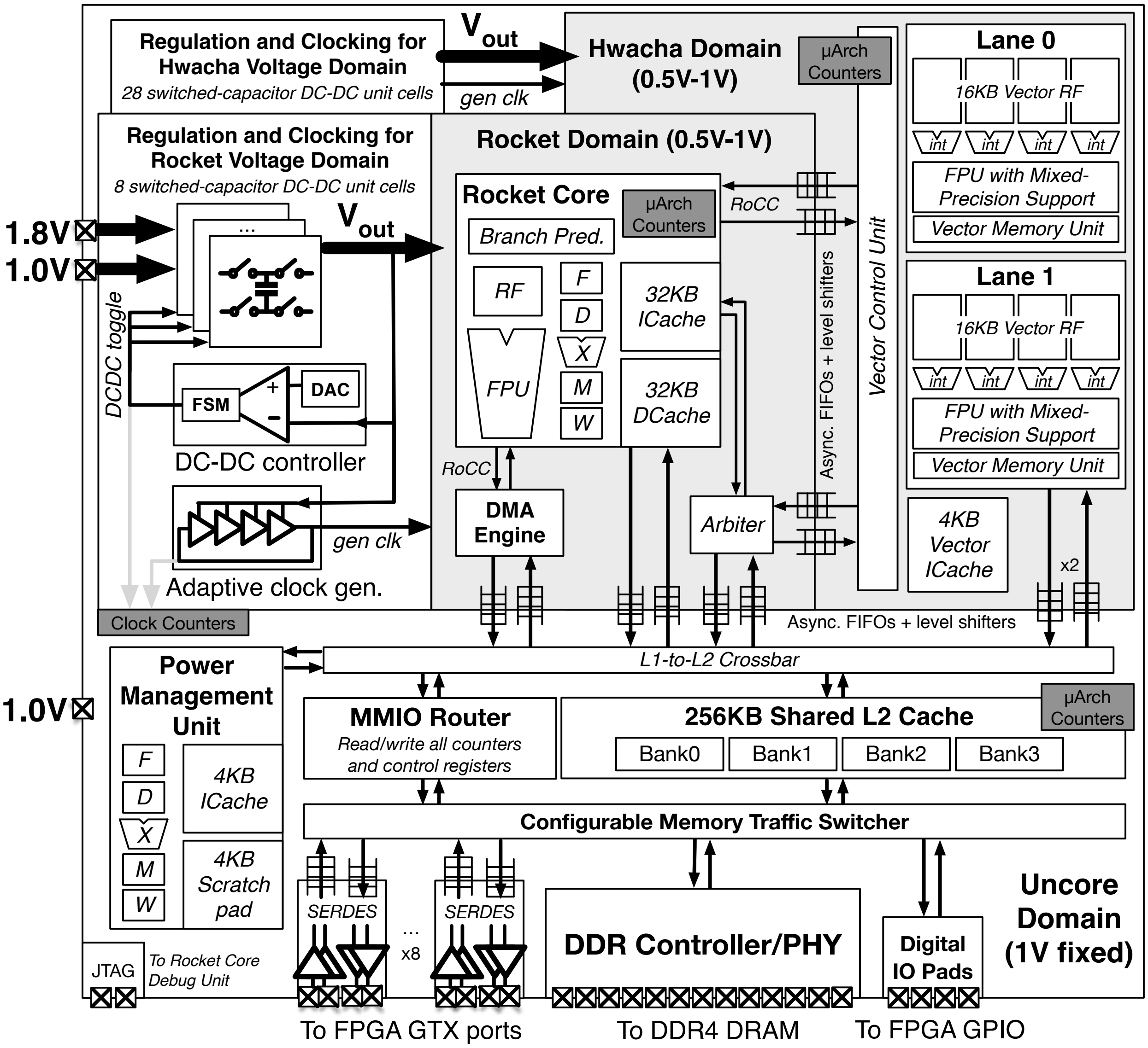


Figure 6.41: A block diagram of the Hurricane-2 testchip.

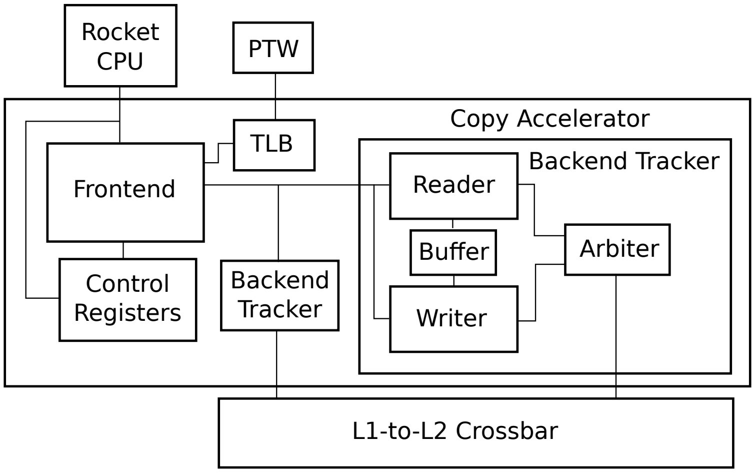


Figure 6.42: A block diagram of the memory copy accelerator in Hurricane-2.

another at high bandwidth. The copy accelerator improves on traditional designs because it can perform its own virtual address translation and can access arbitrary memory addresses, rather than being restricted to the range associated with a particular device. Figure 6.42 shows a block diagram of the memory copy accelerator.

The Hwacha voltage domain contains a version of the vector accelerator. The most significant implementation change from Hurricane-1 is the implementation of a second vector lane, which allows increased data-level parallelism without requiring code recompilation. The second lane doubles the available floating-point resources, allowing a peak throughput of 16 double-precision FLOPs per cycle for the entire vector unit. Each lane has its own port into the shared memory system, increasing available bandwidth of data into and out of the accelerator so that data movement can keep pace with its arithmetic capacity. The vector instruction cache was reduced in size to 4 KiB as simulations showed no performance penalty from this reduction. Unlike in previous designs, Rocket and Hwacha use vendor-supplied SRAM macros, reducing area but possibly limiting functionality at low voltages.

The Rocket and Hwacha domains are supplied by the integrated voltage regulators first implemented in Raven-3, and their clocks are provided by the adaptive clock generator first implemented in Raven-4. The flying capacitance was divided between the two varying-voltage domains according to their area, so 8 unit cells were assigned to the Rocket domain and 28 unit cells were assigned to the Hwacha domain. As noted in Section 6.2.2, the necessity of supplying a single external reference voltage limited efficiency in the case of rapid toggling between different voltage modes. The V_{ref} that achieves peak energy efficiency also varies with workload as shown by the results in Section 6.1.2. The SS-SC controller in Hurricane-2 implements an integrated digital-to-analog converter (DAC) that can generate the necessary reference voltage for use by the comparator. The DAC can rapidly switch between different output voltages following SS-SC mode changes, and can also be adjusted by the PMU to optimize converter efficiency as the workloads of the application cores vary over time.

The fixed-voltage uncore domain contains the shared memory system, PMU processor, and other IP blocks. The L2 cache consists of four banks with a total capacity of 256 KiB that is shared and coherent between the two processors via a broadcast-based MESI protocol. Each bank can have up to four non-conflicting transactions in flight at once using a hardware construct called a tracker. A shared memory-mapped IO router directs memory-mapped requests throughout the system. In contrast to the PMU in the Raven-4 design, the Hurricane-2 PMU is a minimal version of the five-stage Rocket processor. The PMU processor implements the RV64IM ISA, and includes a slow integer multiplier, a 4 KiB instruction cache, and a 4 KiB data scratchpad. A JTAG debug interface is also implemented to access a debug port in the Rocket core.

The HTIF interface was redesigned in Hurricane-2 to provide more flexibility and robustness in the operation of the memory system. The digital interface to the FPGA now tunnels serialized memory traffic using the TileLink protocol [139]. The interface is implemented so that memory requests are serviced but the FPGA can still issue its own memory requests to read and write memory-mapped registers on chip. Eight high-speed SERDES lanes are rearchitected from Hurricane-1 to improve performance and to implement the TileLink protocol. A fully configurable memory traffic switcher can direct memory traffic to this slow digital interface, over one or more high-speed serial links, or to the DDR controller for eventual consumption by dedicated DRAM.

The DDR controller and PHY are third-party IP implemented to interface with a 512 MB DDR4 SDRAM to provide a realistic memory system for large workloads. The TileLink connection from the memory switcher is converted to AXI for use by the memory controller, and an AHB port is used to configure the controller and PHY for operation. Because the DDR PHY and companion DRAM must operate at specific frequencies, the AXI connection implements bisynchronous queues, allowing the DDR controller and PHY to operate in their own clock domain.

The Hurricane-2 testchip implements many of the counters described in Section 4.1.2 for use in integrated power management. The Rocket voltage domain includes L1 data cache hit counters, load miss counters, store miss counters, and AMO miss counters, as well as L1 instruction cache miss counters and a wait-for-interrupt counter. The Hwacha voltage domain has counters that track the number of memory, ALU, and predication operations active in the master sequencer, the number of outstanding memory operations in the vector register unit, and the depth of each of the RoCC request and response queues. The uncore counters track the hit and miss counts of each client for each L2 bank, as well as the number of active trackers in each bank. Counters also track the frequency of each clock in the design, including the SS-SC toggle clocks that can be used to estimate power for each voltage domain. All of these counters are memory-mapped and accessible by the PMU in just a few cycles.

The floorplan of the Hurricane-2 testchip is shown in Figure 6.43. The top-level layout was completely redesigned to accommodate the large DDR PHY IP, which has a large size and irregular aspect ratio. The Rocket and Hwacha layouts also had to be changed to fit within the new floorplan and to place the heterogeneous SRAM macros which now form their caches. The uncore voltage domain is overprovisioned in area, partly due to the irregular

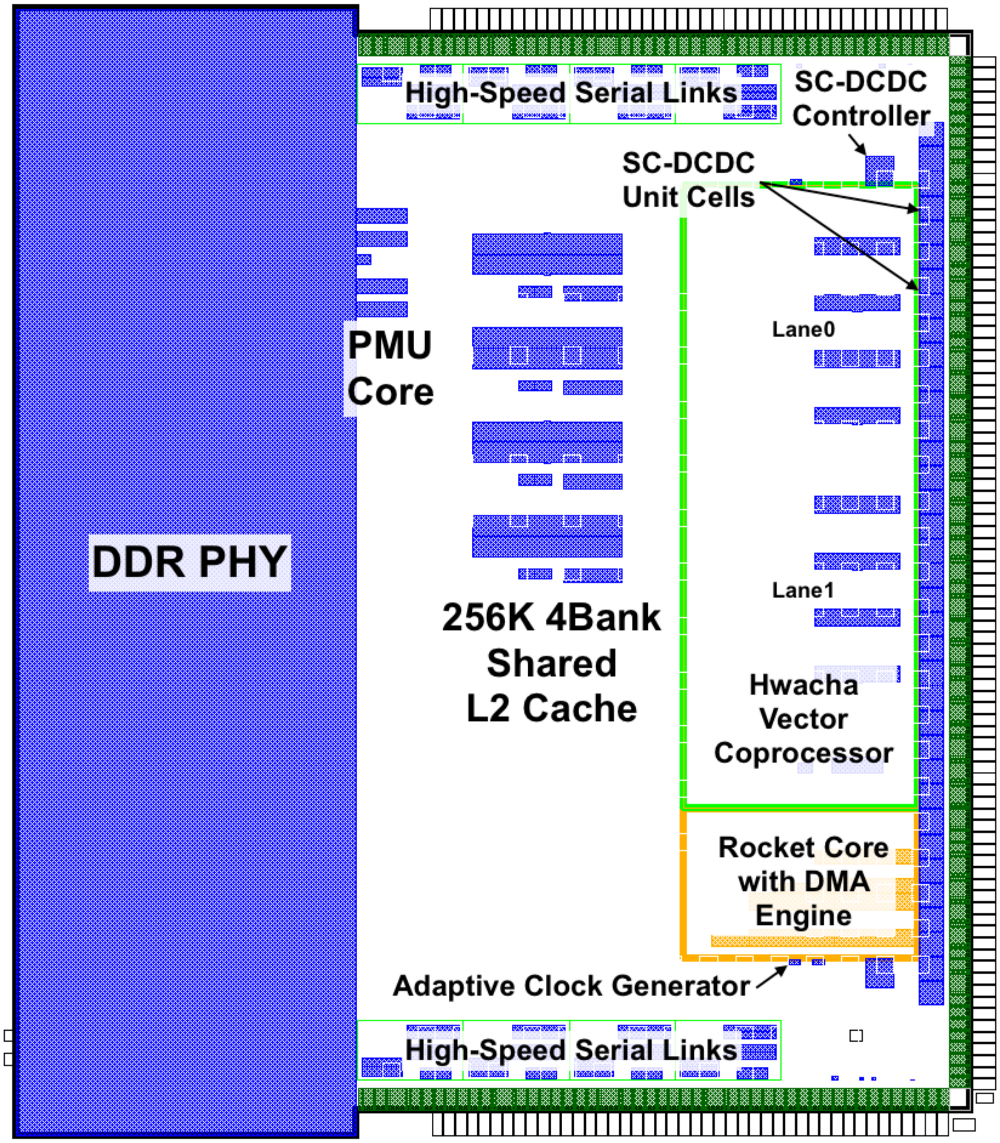


Figure 6.43: Annotated floorplan of the Hurricane-2 testchip.

aspect ratio of the PHY that made cross-chip routing difficult. The PHY macro includes its own power and signal bumps, and the remaining area implements its own bumps in a three-sided ring. Wirebond pads are also included as a backup packaging method, although the DDR PHY will not function if the chip is packaged in this way. The chip is being fabricated in 28 nm UTBB FD-SOI. The chip dimensions total 17.3 mm^2, with the Rocket domain occupying 0.51 mm^2, the Hwacha domain taking up 2.14 mm^2, and the DDR PHY using 5.61 mm^2.

6.4.2 Simulating Fine-Grained AVS

As the Hurricane-2 testchip is still being fabricated at the time of publication, the system was instead simulated to determine the potential energy savings from FG-AVS. The Hurricane-2 RTL corresponding to the design that was taped out was simulated, with black-box behavioral models standing in for the IP blocks in the design, including the DDR PHY. The backing memory was simulated using a Denali memory model [140] configured with the timing behavior matching the DDR4 chip that will be used on the test board. The memory traffic switcher was configured to send all memory traffic over the DDR interface (the SERDES links were not utilized). While the slow speed of RTL simulation limits the duration of programs that can be simulated, the simulations can nonetheless provide useful guidance on the feasibility of FG-AVS in the corresponding testchip.

The behavioral models of the SS-SC converters and the adaptive clock generator are not sophisticated enough to simulate accurate system timing behavior at the four operating modes. Instead, adaptive voltage scaling is simulated by changing the settings of the clock muxes that provide clocks to the variable-voltage domains. The clocks for the Rocket and Hwacha domains can therefore be toggled between several pre-set frequencies. The clock mux settings are controlled by memory-mapped SCRs, the same mechanism used to control SS-SC mode settings, so the PMU program code is similar to programs that would be run in silicon. However, the clock mux setting changes take effect the cycle after they are actuated, unlike SS-SC mode changes, which take several cycles to complete. This results in simulated AVS that demonstrates marginally more responsiveness than would be expected in silicon.

Fine-Grained AVS In Time

Figure 6.44 shows pseudocode representing a PMU program that exercises FG-AVS in time. The algorithm repeatedly polls the counter tracking L2 misses sourced from the Rocket data cache. If a change in the value of this counter is detected, the Rocket clock frequency is reduced to one-fourth the uncore frequency, simulating the performance effects of a reduction in voltage. If the counter is unchanged, the Rocket clock is set to match the faster uncore clock frequency. This algorithm executes on the PMU processor while the application core executes short test programs. The application programs do not have vector instructions, so control of the Hwacha domain is not considered in this experiment.

```
for (;;) { // PMU algorithm
  poll_new = read_scr(L2_MISS_COUNTER);
  if ( poll_old != poll_new ) {
    set_rocket_clock_mux(FAST_CLOCK_DIV4);
    poll_old = poll_new;
  } else {
    set_rocket_clock_mux(FAST_CLOCK);
  }
}
```

Figure 6.44: Pseudocode for a PMU program to demonstrate FG-AVS in time on Hurricane-2.

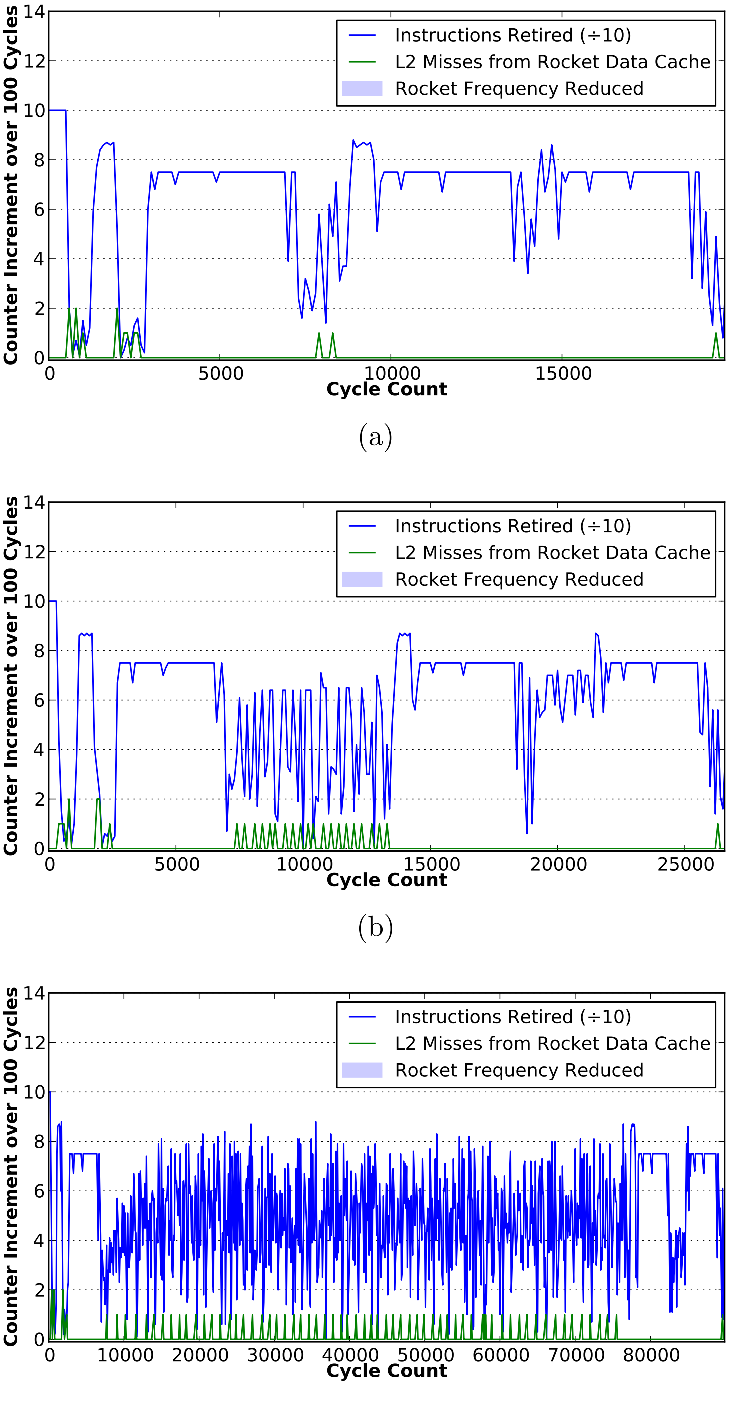
Figure 6.45 shows time-series results of the AVS program as the application core executes three short benchmarks. The instructions retired counter indicates forward progress by the Rocket core; there is strong correlation between an increment of the L2 miss counter and a decrease in the rate of instructions retired. The PMU is able to sense the L2 miss counter increment and actuate a change in the Rocket operating mode in tens of cycles or less, demonstrating the ability of the Hurricane-2 system to achieve fine temporal granularity for AVS. The algorithm saves energy in each benchmark by setting the Rocket domain to the simulated low-voltage operating mode for part of the operation while negligibly impacting benchmark runtime.

The energy model presented in [141] can be used to estimate the energy savings of the PMU algorithm. This model assumes that voltage can be reduced during periods of inactivity without impacting runtime, as is the case here, and accounts for leakage and dynamic energy savings as well as regulator conversion efficiencies. Table 6.7 shows the parameter values used to inform the model, many of which are based on measurements from prior testchips. The energy savings for each benchmark are proportional to the length of time spent in the low-voltage operating mode. Table 6.8 shows the energy savings for the Rocket domain calculated from the model for each benchmark. The benchmarks attain an average energy savings of 10.1%.

Fine-Grained AVS With Multiple Levels

The algorithm presented in the previous section used just a single low-voltage mode to achieve energy savings, but the SC-DCDC converters are capable of generating multiple output voltages and toggling rapidly between them. Accordingly, it is worth investigating whether the algorithmic application of multiple low-voltage levels can achieve additional voltage savings at fine-grained timescales.

Figure 6.46 shows pseudocode of two modifications of the previous algorithm that incorporate multi-level voltage mode switching. In these algorithms, the processor operates either at the uncore clock frequency (representing a high voltage), one-fourth the uncore clock frequency (representing a low voltage), or one-half the uncore clock frequency (representing an



(a)

(b)

(c)

Figure 6.45: Execution traces with counter values for median (a), vector-vector add (b), and sparse matrix-vector multiplication (c) benchmarks executing on the application core while running the AVS algorithm on the PMU core. The cycle count is measured in uncore cycles, which are invariant to core clock frequency changes.

Model Parameter	Value	Source
Nominal supply voltage	1V	Chip implementation
Scaled supply voltage	0.67V	Regulator implementation
Frequency ratio at low voltage	0.25	Experimental condition
Process threshold voltage	450mV	Simulation estimate
Ratio of static to dynamic power at nominal voltage	0.2	Measured result from Raven-4, ICC power report
Clock gating efficiency	0	Chip implementation (Hurricane-2 does not implement clock gating)
Voltage regulator efficiency	0.86	Measured result from Raven-4

Table 6.7: Parameters used in the energy model from [141] to calculate simulated energy savings.

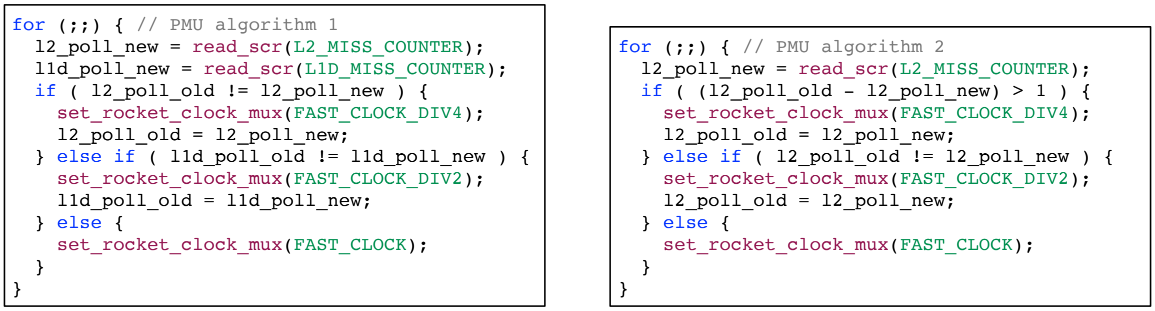
Program	Ratio at Low Voltage	Energy Savings
Median	8.8%	7.2%
Vector-Vector Add	12.0%	9.8%
Sparse Matrix-Vector Multiply	16.3%	13.3%

Table 6.8: Energy savings for the Rocket voltage domain resulting from the implementation of FG-AVS.

intermediate voltage). The first algorithm polls both the L2 cache misses sourced from the Rocket data cache and L1 data cache misses directly. The latter may indicate a forthcoming L2 miss, and it likely indicates a brief period of processor idleness even in the case of an L2 hit. The core clock frequency is reduced to one-fourth the uncore clock frequency when an L2 miss is detected and one-half the base frequency if an L1 data cache miss is detected but an L2 miss is not. If neither event occurs within the polling interval, the frequency is increased to match the uncore clock rate. This algorithm is intended to behave similarly to the algorithm shown in Figure 6.44 when an L2 miss is detected, but it will spend additional time in the medium-voltage state that may save additional energy.

The second algorithm to take advantage of multiple voltage levels is actuated only by L2 cache misses sourced from the Rocket data cache. It modifies the algorithm shown in Figure 6.44 by only reducing the core frequency to one-fourth the uncore frequency when multiple L2 misses are detected within the polling interval. If a single cache miss is detected, the core frequency is instead reduced to one-half the uncore frequency. This algorithm will spend less time in the low-voltage state, but it may better track the performance impact of a single L2 cache miss, which may not warrant the full reduction to the lower frequency.

Figure 6.47 and Table 6.9 show the results of these algorithms running while the Rocket core executes the SPMV microbenchmark, including the energy savings of the programs



```
for (;;) { // PMU algorithm 1
  l2_poll_new = read_scr(L2_MISS_COUNTER);
  l1d_poll_new = read_scr(L1D_MISS_COUNTER);
  if ( l2_poll_old != l2_poll_new ) {
    set_rocket_clock_mux(FAST_CLOCK_DIV4);
    l2_poll_old = l2_poll_new;
  } else if ( l1d_poll_old != l1d_poll_new ) {
    set_rocket_clock_mux(FAST_CLOCK_DIV2);
    l1d_poll_old = l1d_poll_new;
  } else {
    set_rocket_clock_mux(FAST_CLOCK);
  }
}
```

```
for (;;) { // PMU algorithm 2
  l2_poll_new = read_scr(L2_MISS_COUNTER);
  if ( (l2_poll_old - l2_poll_new) > 1 ) {
    set_rocket_clock_mux(FAST_CLOCK_DIV4);
    l2_poll_old = l2_poll_new;
  } else if ( l2_poll_old != l2_poll_new ) {
    set_rocket_clock_mux(FAST_CLOCK_DIV2);
    l2_poll_old = l2_poll_new;
  } else {
    set_rocket_clock_mux(FAST_CLOCK);
  }
}
```

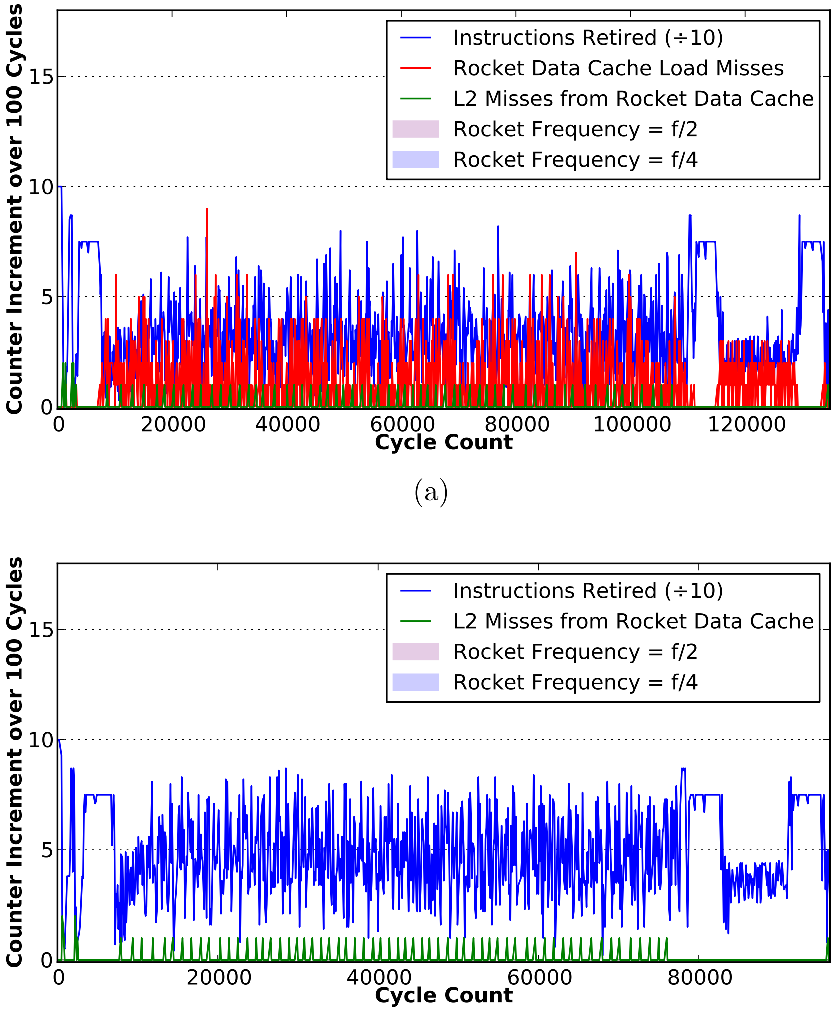
Figure 6.46: Pseudocode for two power-management algorithms that take advantage of multiple AVS levels.

Multi-Level Algorithm	Ratio at Low Voltage	Ratio at Medium Voltage	Increase in Execution Time	Energy Savings
1 (DCache Misses)	10.6%	61.3%	47.9%	9.2%
2 (Multiple L2 Misses)	0.4%	16.9%	-	8.6%

Table 6.9: Energy savings for the Rocket voltage domain resulting from the implementation of FG-AVS with multiple simulated voltage levels.

as estimated by the model in [141]. Neither three-level algorithm is able to outperform the two-level algorithm presented in the previous section, which was able to reduce energy consumption by 13.3% without impacting performance. The first algorithm dramatically increases the execution time of the benchmark; the reduced frequency on a data cache miss impacts performance because the idleness caused by the event is shorter than the polling interval. Furthermore, the proportion of execution time spent in the low-voltage mode is reduced compared to the two-level algorithm, perhaps because the lengthened polling interval due to the additional conditional clause in the PMU loop causes fewer distinct L2 misses to be detected. These two effects obviate almost all of the energy savings of the algorithm relative to the fixed-voltage baseline, and the energy-performance tradeoff is worse than that of simple coarse-grained voltage scaling. The second algorithm detects very few instances of multiple L2 misses, and therefore reduces energy consumption almost entirely via the medium-voltage state, which necessarily saves less energy than the low-voltage state during these periods of processor idleness. Accordingly, this algorithm sees less energy savings than the two-level approach.

There are many possible algorithms that could take advantage of various combinations of counter information and voltage level responses, so the ineffectiveness of the algorithms presented here does not rule out the possibility that the use of multiple voltage levels could save additional energy for FG-AVS in time. However, given the poor performance of these two candidate algorithms, the need for multiple voltage levels in FG-AVS systems is evidently less important than the ability for fast switching in time and fine spatial granularity. Systems



(a)

(b)

Figure 6.47: Execution traces with counter values for the two multi-level PMU algorithms. The first algorithm is actuated by L1 data cache load misses (a), and the second by counting multiple L2 cache misses (b).

```
for (;;) { // PMU algorithm
  poll_rocc_cmd = read_scr(ROCC_CMD_COUNTER);
  poll_rocc_rd = read_scr(ROCC_RD_COUNTER);
  if ( poll_rocc_cmd > 8 || poll_rocc_rd > 8 ) {
    set_rocket_clock_mux(FAST_CLOCK_DIV2);
    set_hwacha_clock_mux(FAST_CLOCK);
  } else if ( poll_rocc_cmd == 0 || poll_rocc_rd == 0 ){
    set_rocket_clock_mux(FAST_CLOCK);
    set_hwacha_clock_mux(FAST_CLOCK_DIV4);
  } else {
    set_rocket_clock_mux(FAST_CLOCK);
    set_hwacha_clock_mux(FAST_CLOCK);
  }
}
```

Figure 6.48: Pseudocode for a PMU program to demonstrate FG-AVS in space on Hurricane-2.

that implement many voltage levels (or a continuous voltage range) for FG-AVS are likely overdesigned.

Fine-Grained AVS In Space

The decreased spatial granularity of Hurricane-2 can also be used to save energy. Figure 6.48 shows pseudocde for an algorithm that actuate changes in the operating modes of both the Rocket and Hwacha domains. The algorithm polls counters that track the current occupancy of queues making up part of the RoCC interface between Rocket and Hwacha. Each queue has a depth of 32. If the queues are empty, then Rocket has not issued any outstanding vector instructions to Hwacha, so the algorithm slows the Hwacha domain. If the queues are occupied, then there is outstanding work for Hwacha to complete and the Hwacha clock is sped up. If one of the queues has occupancy greater than eight, then Rocket is slowed, as it has issued enough work to Hwacha that it does not need to run at full rate (vector instructions take longer to execute than standard RISC-V instructions because each instruction encodes many operations). However, experimentation revealed that if Rocket was slowed to one-fourth the uncore clock in this condition, the queues would empty and Hwacha would stall before Rocket could be sped up to issue more instructions. Accordingly, the algorithm only slows Rocket to one-half the clock rate when the queues are full, allowing Rocket to keep Hwacha fed and preserving performance. (This algorithm therefore does take advantage of three different voltage levels, although each voltage domain switches between just two levels.) If the queue occupancy drops below eight, Rocket is sped up to ensure that Hwacha does not run out of instructions.

Figure 6.49 shows the simulated trace of the system while executing a matrix-multiply benchmark. The program has three well-defined phases: two in which Hwacha is inactive,

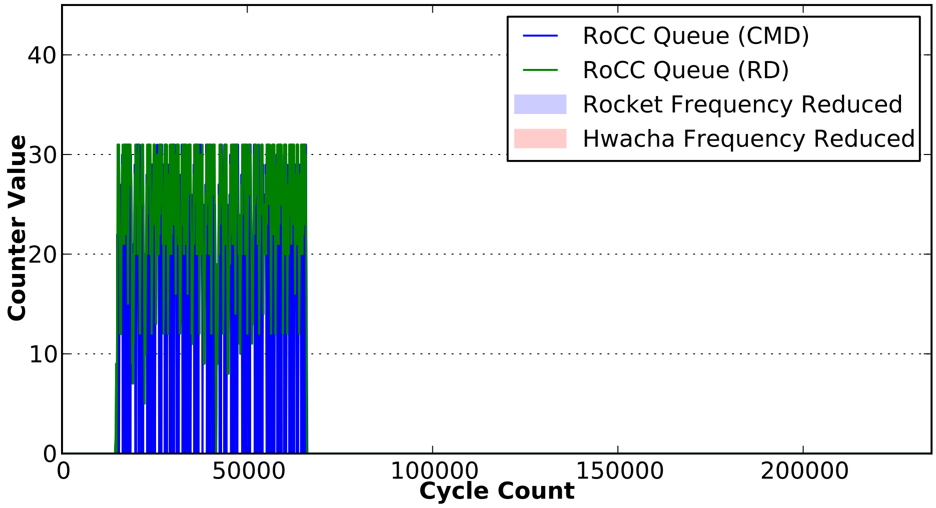


Figure 6.49: An execution trace showing the RoCC queue counters used for feedback in the AVS algorithm.

Voltage Area	Ratio at Low Voltage	Energy Savings
Rocket	22.0%	10.8%
Hwacha	78.0%	63.5%

Table 6.10: Energy savings for each voltage algorithm resulting from FG-AVS as the application core executes a matrix-multiply benchmark.

and one in which Hwacha is active and Rocket is only feeding instructions to Hwacha. The PMU algorithm correctly detects these phases and performs the appropriate scaling behavior with very fast response time of tens of cycles, resulting in negligible performance penalty. Table 6.10 shows the energy savings of the PMU program as estimated by the model in [141]. The relative power of the two domains as reported by the place-and-route tool can be used to estimate an overall energy savings of 46.0% for the full system. Most of these savings result from reducing the operating voltage and frequency of Hwacha when it is not in use. These results demonstrate the utility of fine-grained spatial AVS in saving energy when different parts of the SoC transition between different levels of activity.

Chapter 7

Conclusion

Fine-grained adaptive voltage scaling is key to improving energy efficiency in modern SoCs, but numerous technical hurdles have impeded the widespread adoption of this technique. Integrated voltage regulators, necessary to supply the many voltages required for FG-AVS, are challenging to implement with high efficiency and power density. Clock generation and data synchronization must be carefully considered to ensure functional correctness and operating efficiency. All of these design decisions impose overheads that mitigate the energy savings gained by the approach.

This work has presented circuit designs and system implementations that prove the feasibility of FG-AVS. By building fully-featured integrated systems, all overheads of FG-AVS are necessarily accounted for. The 28 nm SoC testchips presented nonetheless show that FG-AVS is possible to implement and can save energy in silicon.

7.1 Summary of Contributions

The major contributions of this work include:

- A thorough overview of the challenges of adaptive voltage scaling, with particular focus on barriers to widespread industry adoption (Chapter 2).
- An analysis of the system implications of a proposed voltage regulation and clock generation scheme to enable FG-AVS (Section 3.3).
- A survey of integrated power management options for FG-AVS, including proposals for counter-based power and activity measurement, an integrated control processor, and power-management algorithms to perform adaptive feedback (Chapter 4).
- A proposal for a novel low-latency clock-domain-crossing circuit using pausible clocks, including detailed analytical timing analysis and simulation results (Section 5.3).

- The Raven-4 testchip implementation, which demonstrated practical FG-AVS with energy savings of up to 39.6% with minimal performance overhead (Section 6.2).
- The Hurricane-1 testchip implementation (Section 6.3), which proved the ability to supply multiple independent voltage domains via integrated regulation and showed the feasibility of counter-based power management.
- The Hurricane-2 testchip implementation (Section 6.4), as well as simulation results demonstrating FG-AVS algorithms in space and in time executing on the system that can save up to 46.0% energy with no impact on performance.

7.2 Future Work

While technology trends suggest favorable conditions for the use of FG-AVS, further research is required to ensure that FG-AVS is widely adopted in industrial designs.

- The Hurricane-2 testchip is still in fabrication at the time of writing. Once the chip is fabricated and packaged, the algorithms for FG-AVS simulated in Section 6.4.2 should be executed in silicon to measure the actual performance and energy implications. FG-AVS can also be evaluated in the context of more realistic workloads such as convolutional neural nets and graph traversal.
- This work does not consider the alternative power-reduction approach of power gating, in which the voltage of blocks is reduced to zero when they are totally inactive. Power gating requires additional hardware support, such as isolation cells and state retention logic, to safely implement. Silicon implementations could compare the energy savings and implementation costs of power gating to those of FG-AVS. Circuit designs capable of both power gating and FG-AVS, as well as the appropriate algorithms to take advantage of them, could also be explored.
- The testchips presented in this work do not methodically evaluate the di/dt implications of integrated voltage regulation. The digital loads supplied by the regulators are likely too small to exhibit significant di/dt noise, and additional circuitry would be required to measure the di/dt effects across the power grid. Dedicated "di/dt virus" circuitry and the appropriate instrumentation could be added to a future testchip to determine the extent to which integrated regulators improve resilience to di/dt droop.
- As noted in Section 2.1.1, integrated voltage regulators generally must over-provision their power delivery capacity for the worst-case current demand of the load in each voltage domain, resulting in more power delivery capacity on the die than is likely to be used at any one time. Reconfigurable voltage regulator networks could ameliorate this problem by sharing parts of the passive elements used in regulators between multiple domains, apportioning the total capacitance according to load. Reconfigurable power

delivery has been proposed in the literature and evaluated in simulation [112], but silicon evaluation could prove the feasibility of the concept.

- The pausible bisynchronous FIFO presented in Section 5.3 was evaluated only in simulation. A silicon implementation would confirm the utility of the approach in reducing asynchronous interface latencies.

- Most of the synchronization circuits presented in Chapter 5 assume fully asynchronous clocks on either side of the clock crossing. In the case in which neighboring clocks are sometimes synchronous or ratiochronous, the constraints on the clock crossing could be temporarily loosened to reduce interface latency. A clock crossing could be designed that dynamically switches between asynchronous and synchronous modes, depending on the operating conditions of the neighboring domains.

- This work does not consider the use of fully asynchronous logic, such as bundled-data or quasi-delay-insensitive logic styles. However, in many ways, asynchronous logic is ideally suited to FG-AVS. With no clock, the challenges of clock generation and synchronization are implicitly solved; without these constraints, the number of independent voltage domains can be increased arbitrarily, limited only by the number of independent voltage regulators that can be implemented. Asynchronous logic imposes its own design overheads, so a future silicon implementation could compare the GALS approach considered in this work with a fully asynchronous design.

- The simulations presented in Section 6.4.2 are limited in duration because of the slow progression of RTL simulation. Faster simulation can be accomplished with the use of FPGAs, but care must be taken to design the simulation environment so that the performance tradeoffs of FG-AVS are correctly simulated. FPGA-based architecture simulators such as RAMP Gold [142] could be extended to model FG-AVS, allowing longer, more realistic benchmarking of FG-AVS algorithms.

The study of FG-AVS and its associated enabling technologies will continue to drive significant circuit and architecture innovation in the years to come.

Bibliography

[1] J. A. Laitner, C. P. Knight, V. L. McKinney, and K. Ehrhardt-Martinez, “Semiconductor technologies: The potential to revolutionize U.S. energy productivity (Part III),” *Environmental Quality Management*, vol. 19, no. 4, pp. 29–50, 2010.

[2] Y. Sverdlik. (Jun. 2016). Here’s how much energy all us data centers consume, Data Center Knowledge, [Online]. Available: `http://www.datacenterknowledge.com/archives/2016/06/27/heres-how-much-energy-all-us-data-centers-consume/` (visited on 06/09/2017).

[3] P. P. Gelsinger, “Microprocessors for the new millennium: Challenges, opportunities, and new frontiers,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2001, pp. 22–25.

[4] J. Karidis, J. E. Moreira, and J. Moreno, “True value: Assessing and optimizing the cost of computing at the data center level,” in *Proceedings of the ACM International Conference on Computing Frontiers*, May 2009, pp. 185–192.

[5] J. Koomey, *A simple model for determining true total cost of ownership for data centers*, White Paper, Uptime Institute, Oct. 2007.

[6] B. Grot, D. Hardy, P. Lotfi-Kamran, B. Falsafi, C. Nicopoulos, and Y. Sazeides, “Optimizing data-center TCO with scale-out processors,” *IEEE Micro*, vol. 32, no. 5, pp. 52–63, Sep. 2012.

[7] A. Pilon. (Jan. 2016). Smartphone battery survey: Battery life considered important, AYTM, [Online]. Available: `https://aytm.com/blog/daily-survey-results/smartphone-battery-survey/` (visited on 06/09/2017).

[8] M. Alioto, Ed., *Enabling the Internet of Things: From Integrated Circuits to Integrated Systems*. Cham, Switzerland: Springer International Publishing, 2017.

[9] S. Jain *et al.*, “A 280mV-to-1.2V wide-operating-range IA-32 processor in 32nm CMOS,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2012, pp. 66–67.

[10] B. Zimmer *et al.*, “A RISC-V vector processor with simultaneous-switching switched-capacitor DC-DC converters in 28 nm FDSOI,” *IEEE Journal of Solid-State Circuits*, vol. 51, no. 4, pp. 930–942, Apr. 2016.

[11] 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.

[12] D. M. Chapiro, “Globally-asynchronous locally-synchronous systems,” PhD thesis, Department of Computer Science, Stanford University, Oct. 1984.

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

[14] S. S. Sapatnekar, “Overcoming variations in nanometer-scale technologies,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 1, pp. 5–18, May 2011.

[15] C. Hou, “A smart design paradigm for smart chips,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 8–13.

[16] N. Tega *et al.*, “Increasing threshold voltage variation due to random telegraph noise in FETs as gate lengths scale to 20 nm,” in *Proceedings of the Symposium on VLSI Technology*, Apr. 2009, pp. 50–51.

[17] A. Shye, B. Scholbrock, and G. Memik, “Into the wild: Studying real user activity patterns to guide power optimizations for mobile architectures,” in *Proceedings of the IEEE/ACM International Symposium on Microarchitecture*, Dec. 2009, pp. 168–178.

[18] W. Kim and M. Gupta, “System level analysis of fast, per-core DVFS using on-chip switching regulators,” *Proceedings of the IEEE International Symposium on High Performance Computer Architecture*, pp. 123–134, Feb. 2008.

[19] K. Rangan, G. Wei, and D. Brooks, “Thread motion: fine-grained power management for multi-core systems,” *Proceedings of the ACM/IEEE International Symposium on Computer Architecture*, pp. 302–313, Jun. 2009.

[20] S. Eyerman and L. Eeckhout, “Fine-grained DVFS using on-chip regulators,” *ACM Transactions on Architecture and Code Optimization*, vol. 8, no. 1, pp. 1–24, Apr. 2011.

[21] E. Sperling. (Mar. 2014). How much will that chip cost? Semiconductor Engineering, [Online]. Available: `http://semiengineering.com/how-much-will-that-chip-cost/` (visited on 06/12/2017).

[22] R. Smith. (May 2017). NVIDIA Volta unveiled: GV100 GPU and Tesla V100 accelerator announced, AnandTech, [Online]. Available: `http://www.anandtech.com/show/11367/nvidia-volta-unveiled-gv100-gpu-and-tesla-v100-accelerator-announced` (visited on 06/12/2017).

[23] W. J. Dally, C. Malachowsky, and S. W. Keckler, “21st century digital design tools,” in *Proceedings of the Design Automation Conference*, Jun. 2013.

[24] N. Kurd *et al.*, “Haswell: A family of IA 22nm processors,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2014, pp. 112–113.

[25] E. Burton *et al.*, “FIVR — Fully integrated voltage regulators on 4th generation Intel Core SoCs,” in *Proceedings of the Applied Power Electronics Conference*, Mar. 2014, pp. 432–439.

[26] A. Nalamalpu *et al.*, “Broadwell: A family of IA 14nm processors,” in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2015, pp. C314–C315.

[27] A. Grenat *et al.*, “Increasing the performance of a 28nm x86-64 microprocessor through system power management,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 74–75.

[28] C. Gonzalez *et al.*, “Power9: A processor family optimized for cognitive computing with 25Gb/s accelerator links and 16Gb/s PCIe Gen4,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 50–51.

[29] T. Singh *et al.*, “Zen: A next-generation high-performance x86 core,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 52–53.

[30] Z. Toprak-Deniz *et al.*, “Distributed system of digitally controlled microregulators enabling per-core DVFS for the POWER8 microprocessor,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2014, pp. 98–99.

[31] “Switching regulator fundamentals,” Texas Instruments, Tech. Rep. SNVA559A, Sep. 2016.

[32] D. S. Gardner *et al.*, “Integrated on-chip inductors with magnetic films,” in *Proceedings of the International Electron Devices Meeting*, Dec. 2006.

[33] J. Lee, G. Hatcher, L. Vandenberghe, and C. K. K. Yang, “Evaluation of fully-integrated switching regulators for CMOS process technologies,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 9, pp. 1017–1027, Sep. 2007.

[34] H. Krishnamurthy *et al.*, “A 500 MHz, 68% efficient, fully on-die digitally controlled buck voltage regulator on 22nm tri-gate CMOS,” in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2014, pp. 167–168.

[35] 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 *Proceedings of the IEEE Workshop on Control and Modeling for Power Electronics*, Jun. 2010.

[36] M. D. Seeman, “A design methodology for switched-capacitor DC-DC converters,” PhD thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, May 2009.

[37] L. G. Salem and P. P. Mercier, “A battery-connected 24-ratio switched capacitor PMIC achieving 95.5%-efficiency,” in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2015, pp. C340–C341.

[38] W. Jung, D. Sylvester, and D. Blaauw, “A rational-conversion-ratio switched-capacitor DC-DC converter using negative-output feedback,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 218–219.

[39] 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, Sep. 2011.

[40] S. Clerc *et al.*, “A 0.33V/-40C process/temperature closed-loop compensation SoC embedding all-digital clock multiplier and DC-DC converter exploiting FDSOI 28nm back-gate biasing,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2015, pp. 150–151.

[41] S. Kim *et al.*, “Enabling wide autonomous DVFS in a 22nm graphics execution core using a digitally controlled hybrid LDO/switched-capacitor VR with fast droop mitigation,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2015, pp. 154–155.

[42] M. Turnquist *et al.*, “Fully integrated DC-DC converter and a 0.4V 32-bit CPU with timing-error prevention supplied from a prototype 1.55V Li-ion battery,” in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2015, pp. 320–321.

[43] D. Bol *et al.*, “SleepWalker: A 25-MHz 0.4-V sub-mm^2 7-uW/MHz microcontroller in 65-nm LP/GP CMOS for low-carbon wireless sensor nodes,” *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 20–32, Jan. 2013.

[44] D. N. Truong *et al.*, “A 167-processor computational platform in 65 nm CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, pp. 1130–1144, Apr. 2009.

[45] N. Pinckney, M. Fojtik, B. Giridhar, D. Sylvester, and D. Blaauw, “Shortstop: An on-chip fast supply boosting technique,” in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2013, pp. C290–C291.

[46] N. Pinckney, D. Sylvester, and D. Blaauw, “Supply boosting for high-performance processors in flip-chip packages,” in *Proceedings of the European Solid-State Circuits Conference*, Sep. 2016, pp. 473–476.

[47] E. Beigne *et al.*, “An asynchronous power aware and adaptive NoC based circuit,” *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, pp. 1167–1177, Mar. 2009.

[48] 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, Jan. 2006.

[49] C. R. Lefurgy *et al.*, “Active management of timing guardband to save energy in POWER7,” in *Proceedings of the IEEE/ACM International Symposium on Microarchitecture*, Dec. 2011, pp. 1–11.

[50] M. Abdelfattah *et al.*, “A novel digital loop filter architecture for bang-bang ADPLL,” in *Proceedings of the IEEE System-on-Chip (SoC) Conference*, Sep. 2012, pp. 45–50.

[51] A. Sai, S. Kondo, T. T. Ta, H. Okuni, M. Furuta, and T. Itakura, “A 65nm CMOS ADPLL with 360uW 1.6ps-INL SS-ADC-based period-detection-free TDC,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 336–337.

[52] T. Jang, S. Jeong, D. Jeon, K. D. Choo, D. Sylvester, and D. Blaauw, “A 2.5ps 0.8-to-3.2GHz bang-bang phase- and frequency-detector-based all-digital PLL with noise self-adjustment,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 148–149.

[53] H. Cho *et al.*, “A 0.0047mm^2 highly synthesizable TDC- and DCO-less fractional-N PLL with a seamless lock range of $\mathrm{f_{REF}}$ to 1GHz,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 154–155.

[54] A. Elkholy, A. Elmallah, M. Elzeftawi, K. Chang, and P. K. Hanumolu, “A 6.75-to-8.25GHz, 250fs$_{\mathrm{rms}}$-integrated-jitter 3.25mW rapid on/off PVT-insensitive fractional-N injection-locked clock multiplier in 65nm CMOS,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 192–193.

[55] S. Choi, S. Yoo, and J. Choi, “A 185fs$_{\mathrm{rms}}$-integrated-jitter and -245dB FOM PVT-robust ring-VCO-based injection-locked clock multiplier with a continuous frequency-tracking loop using a replica-delay cell and a dual-edge phase detector,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 194–195.

[56] S. Kundu, B. Kim, and C. H. Kim, “A 0.2-to-1.45GHz subsampling fractional-N all-digital MDLL with zero-offset aperture PD-based spur cancellation and in-situ timing mismatch detection,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 326–327.

[57] H. Kim, Y. Kim, T. Kim, H. Park, and S. Cho, “A 2.4GHz 1.5mW digital MDLL using pulse-width comparator and double injection technique in 28nm CMOS,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 328–329.

[58] H. C. Ngo, K. Nakata, T. Yoshioka, Y. Terashima, K. Okada, and A. Matsuzawa, “A 0.42ps-jitter -241.7dB-FOM synthesizable injection-locked PLL with noise-isolation LDO,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 150–151.

[59] D. Coombs, A. Elkholy, R. K. Nandwana, A. Elmallah, and P. K. Hanumolu, “A 2.5-to-5.75GHz 5mW 0.3ps$_{\mathrm{rms}}$-jitter cascaded ring-based digital injection-locked clock multiplier in 65nm CMOS,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 152–153.

[60] B. Stackhouse *et al.*, “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.

[61] K. Bowman *et al.*, "A 16nm auto-calibrating dynamically adaptive clock distribution for maximizing supply-voltage-droop tolerance across a wide operating range," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2015, pp. 152–154.

[62] M. S. Floyd *et al.*, "Adaptive clocking in the POWER9 processor for voltage droop protection," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 444–445.

[63] P. Li *et al.*, "A 20nm 32-core 64MB L3 cache SPARC M7 processor," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2015, pp. 72–74.

[64] K. Wilcox *et al.*, "A 28nm x86 APU optimized for power and area efficiency," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2015, pp. 84–86.

[65] D. A. Kamakshi, M. Fojtik, B. Khailany, S. Kudva, Y. Zhou, and B. H. Calhoun, "Modeling and analysis of power supply noise tolerance with fine-grained GALS adaptive clocks," in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, May 2016, pp. 75–82.

[66] S. Felix, *Digital frequency locked loop*, US Patent 8,723,571, May 2014.

[67] T. J. Chaney and F. U. Rosenberger, "Characterization and scaling of MOS flip flop performance in synchronizer applications," in *Proceedings of the Caltech Conference On Very Large Scale Integration*, California Institute of Technology, Jan. 1979.

[68] B. Keller *et al.*, "A RISC-V processor SoC with integrated power management at submicrosecond timescales in 28 nm FD-SOI," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 7, pp. 1863–1875, Jul. 2017.

[69] R. Jain, S. Kim, V. Vaidya, J. Tschanz, K. Ravichandran, and V. De, "Conductance modulation techniques in switched-capacitor DC-DC converter for maximum-efficiency tracking and ripple mitigation in 22nm tri-gate CMOS," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, Sep. 2014.

[70] J. Jiang, Y. Lu, W. H. Ki, S. P. U, and R. P. Martins, "A dual-symmetrical-output switched-capacitor converter with dynamic power cells and minimized cross regulation for application processors in 28nm CMOS," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2017, pp. 344–345.

[71] M. K. Song, L. Chen, J. Sankman, S. Terry, and D. Ma, "A 20V 8.4W 20MHz four-phase GaN DC-DC converter with fully on-chip dual-SR bootstrapped GaN FET driver achieving 4ns constant propagation delay and 1ns switching rise time," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2015, pp. 302–304.

[72] C. K. Teh and A. Suzuki, “A 2-output step-up/step-down switched-capacitor DC-DC converter with 95.8% peak efficiency and 0.85-to-3.6V input voltage range,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Jan. 2016, pp. 222–223.

[73] T. M. Andersen *et al.*, “A sub-ns response on-chip switched-capacitor DC-DC voltage regulator delivering 3.7W/mm2 at 90% efficiency using deep-trench capacitors in 32nm SOI CMOS,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2014, pp. 90–91.

[74] G. V. Piqué, “A 41-phase switched-capacitor power converter with 3.8mV output ripple and 81% efficiency in baseline 90nm CMOS,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2012, pp. 98–100.

[75] R. Jevtić *et al.*, “Per-core DVFS with switched-capacitor converters for energy efficiency in manycore processors,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 4, pp. 723–730, Apr. 2015.

[76] D. Jiao, J. Gu, and C. H. Kim, “Circuit design and modeling techniques for enhancing the clock-data compensation effect under resonant supply noise,” *IEEE Journal of Solid-State Circuits*, vol. 45, no. 10, pp. 2130–2141, Oct. 2010.

[77] J. Kwak and B. Nikolić, “A self-adjustable clock generator with wide dynamic range in 28 nm FDSOI,” *IEEE Journal of Solid-State Circuits*, vol. 51, no. 10, pp. 2368–2379, Oct. 2016.

[78] K. Wong, T. Rahal-arabi, M. Ma, and G. Taylor, “Enhancing microprocessor immunity to power supply noise with clock-data compensation,” *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 749–758, Apr. 2006.

[79] X. Jiang, P. Dutta, D. Culler, and I. Stoica, “Micro power meter for energy monitoring of wireless sensor networks at scale,” in *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks*, Apr. 2007, pp. 186–195.

[80] R. McGowen *et al.*, “Power and temperature control on a 90-nm Itanium family processor,” *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 229–237, Jan. 2006.

[81] S. Bhagavatula and B. Jung, “A power sensor with 80ns response time for power management in microprocessors,” in *Proceedings of the IEEE Custom Integrated Circuits Conference*, Sep. 2013.

[82] M. Cochet *et al.*, “On-chip supply power measurement and waveform reconstruction in a 28nm FD-SOI processor SoC,” in *Proceedings of the IEEE Asian Solid-State Circuits Conference*, Nov. 2016, pp. 125–128.

[83] P. Dutta, M. Feldmeier, J. Paradiso, and D. Culler, “Energy metering for free: Augmenting switching regulators for real-time monitoring,” in *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks*, Apr. 2008, pp. 283–294.

[84] Y. Sinangil *et al.*, "A self-aware processor SoC using energy monitors integrated into power converters for self-adaptation," in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2014, pp. 139–140.

[85] 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*, Aug. 2011.

[86] D. C. Snowdon, S. M. Petters, and G. Heiser, "Accurate on-line prediction of processor and memory energy usage under voltage scaling," *Proceedings of the IEEE International Conference on Embedded Software*, pp. 84–93, Sep. 2007.

[87] T. Webel *et al.*, "Robust power management in the IBM z13," *IBM Journal of Research and Development*, vol. 59, no. 4/5, 16:1–16:12, Jul. 2015.

[88] S. Bird, "Software knows best: A case for hardware transparency and measurability," Master's thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, May 2010.

[89] D. Horner, *RISC V - low power instructions*, RISC-V Hardware Development Mailing List, Feb. 2017. [Online]. Available: `https://groups.google.com/a/groups.riscv.org/d/msg/hw-dev/fmn3ux_XLs0/kkPkJnStAwAJ`.

[90] P. Juang, Qiang Wu, Li-Shiuan Peh, M. Martonosi, and D. Clark, "Coordinated, distributed, formal energy management of chip multiprocessors," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, Aug. 2005, pp. 127–130.

[91] J. Pouwelse, K. Langendoen, and H. Sips, "Dynamic voltage scaling on a low-power microprocessor," in *Proceedings of the International Conference on Mobile Computing and Networking*, Jul. 2001, pp. 251–259.

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

[93] C.-H. Hsu and U. Kremer, "The design, implementation, and evaluation of a compiler algorithm for CPU energy reduction," in *Proceedings of the ACM Conference on Programming Language Design and Implementation*, Jun. 2003, pp. 38–48.

[94] Q. Wu *et al.*, "A dynamic compilation framework for controlling microprocessor energy and performance," in *Proceedings of the IEEE/ACM International Symposium on Microarchitecture*, Nov. 2005, pp. 271–282.

[95] Y. Lee, "Z-scale: Tiny 32-bit RISC-V systems," in *OpenRISC Conference*, Oct. 2015.

[96] J. Sharkey, A. Buyuktosunoglu, and P. Bose, "Evaluating design tradeoffs in on-chip power management for CMPs," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, Aug. 2007, pp. 44–49.

[97] Q. Wu, P. Juang, M. Martonosi, and D. W. Clark, "Voltage and frequency control with adaptive reaction time in multiple-clock-domain processors," in *Proceedings of the IEEE Symposium on High-Performance Computer Architecture*, Feb. 2005, pp. 178–189.

[98] R. Efraim, R. Ginosar, C. Weiser, and A. Mendelson, "Energy aware race to halt: A down to EARtH approach for platform energy management," *IEEE Computer Architecture Letters*, vol. 13, no. 1, pp. 25–28, Jan. 2014.

[99] T. Yuki and S. Rajopadhye, "Folklore confirmed: Compiling for speed = compiling for energy," in *Proceedings of the International Workshop on Languages and Compilers for Parallel Computing*, Sep. 2014, pp. 169–184.

[100] D. Snowdon, S. Ruocco, and G. Heiser, *Power management and dynamic voltage scaling: Myths and facts*, 2005.

[101] G. Dhiman, K. K. Pusukuri, and T. Rosing, "Analysis of dynamic voltage scaling for system level energy management," in *Proceedings of the Workshop on Power Aware Computing and Systems*, Dec. 2008.

[102] A. Bhattacharjee and M. Martonosi, "Thread criticality predictors for dynamic performance, power, and resource management in chip multiprocessors," in *Proceedings of the ACM/IEEE International Symposium on Computer Architecture*, Jun. 2009, pp. 290–301.

[103] G. Keramidas, V. Spiliopoulos, and S. Kaxiras, "Interval-based models for run-time DVFS orchestration in superscalar processors," in *Proceedings of the ACM International Conference on Computing Frontiers*, May 2010, pp. 287–296.

[104] E. Talpes and D. Marculescu, "Toward a multiple clock/voltage island design style for power-aware processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 5, pp. 591–603, May 2005.

[105] G. Dhiman and T. S. Rosing, "Dynamic voltage frequency scaling for multi-tasking systems using online learning," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, Aug. 2007, pp. 207–212.

[106] K. Rajamani, H. Hanson, J. Rubio, S. Ghiasi, and F. Rawson, "Application-aware power management," in *Proceedings of the IEEE International Symposium on Workload Characterization*, Oct. 2006, pp. 39–48.

[107] 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, Jan. 2005.

[108] L. Guang, E. Nigussie, L. Koskinen, and H. Tenhunen, "Autonomous DVFS on supply islands for energy-constrained NoC communication," in *Proceedings of the International Conference on Architecture of Computing Systems*, Mar. 2009, pp. 183–194.

[109] Q. Wu, P. Juang, M. Martonosi, and D. W. Clark, "Formal online methods for voltage/frequency control in multiple clock domain microprocessors," in *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*, Oct. 2004, pp. 248–259.

[110] D. Marculescu, "On the use of microarchitecture-driven dynamic voltage scaling," in *Proceedings of the Workshop on Complexity-Effective Design*, Jun. 2000.

[111] 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, May 2005.

[112] W. Godycki, C. Torng, I. Bukreyev, A. Apsel, and C. Batten, "Enabling realistic fine-grain voltage scaling with reconfigurable power distribution networks," in *Proceedings of the IEEE/ACM International Symposium on Microarchitecture*, Dec. 2014, pp. 381–393.

[113] P. Stanley-Marbell, M. S. Hsiao, and U. Kremer, "A hardware architecture for dynamic performance and energy adaptation," in *Proceedings of the International Conference on Power-aware Computer Systems*, Feb. 2003, pp. 33–52.

[114] R. Ginosar, "Fourteen ways to fool your synchronizer," in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, Mar. 2003, pp. 89–96.

[115] C. E. Cummings, "Simulation and synthesis techniques for asynchronous FIFO design," in *Synopsys Users Group Conference User Papers*, 2002.

[116] A. Chakraborty and M. Greenstreet, "Efficient self-timed interfaces for crossing clock domains," in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, Mar. 2003, pp. 78–88.

[117] W. J. Dally and S. G. Tell, "The even/odd synchronizer: A fast, all-digital, periodic synchronizer," in *Proceedings of the IEEE Symposium on Asynchronous Circuits and Systems*, May 2010, pp. 75–84.

[118] K. Yun and R. Donohue, "Pausible clocking: A first step toward heterogeneous systems," in *Proceedings of the IEEE International Conference on Computer Design*, Oct. 1996, pp. 118–123.

[119] R. Mullins and S. Moore, "Demystifying data-driven and pausible clocking schemes," in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, Mar. 2007, pp. 175–185.

[120] B. Keller, M. Fojtik, and B. Khailany, "A pausible bisynchronous FIFO for GALS systems," in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, May 2015, pp. 1–8.

[121] S. Moore, G. Taylor, R. Mullins, and P. Robinson, “Point to point GALS interconnect,” in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, Apr. 2002, pp. 69–75.

[122] I. E. Sutherland, “Micropipelines,” *Communications of the ACM*, vol. 32, no. 6, pp. 720–738, Jun. 1989.

[123] I. Sutherland and S. Fairbanks, “GasP: A minimal FIFO control,” in *Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems*, Mar. 2001, pp. 46–53.

[124] M. Singh and S. Nowick, “MOUSETRAP: High-speed transition-signaling asynchronous pipelines,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 6, pp. 684–698, Jun. 2007.

[125] A. E. Sjogren and C. J. Myers, “Interfacing synchronous and asynchronous modules within a high-speed pipeline,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 5, pp. 573–583, Oct. 2000.

[126] X. Fan, M. Krstić, and E. Grass, “Analysis and optimization of pausible clocking based GALS design,” in *Proceedings of the IEEE International Conference on Computer Design*, Oct. 2009, pp. 358–365.

[127] K. Asanović *et al.*, “The Rocket Chip Generator,” Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17, Apr. 2016.

[128] Y. Lee *et al.*, “A 45nm 1.3GHz 16.7 double-precision GFLOPS/W RISC-V processor with vector accelerators,” in *Proceedings of the European Solid-State Circuits Conference*, Sep. 2014, pp. 199–202.

[129] R. M. Russell, “The CRAY-1 computer system,” *Communications of the ACM*, vol. 21, no. 1, pp. 63–72, Jan. 1978.

[130] H. P. Le, J. Crossley, S. R. Sanders, and E. Alon, “A sub-ns response fully integrated battery-connected switched-capacitor voltage regulator delivering 0.19W/mm2 at 73% efficiency,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2013, pp. 372–373.

[131] P. Flatresse *et al.*, “Ultra-wide body-bias range LDPC decoder in 28nm UTBB FD-SOI technology,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 2013, pp. 424–425.

[132] *Zedboard*. [Online]. Available: `http://zedboard.org/product/zedboard` (visited on 08/10/2017).

[133] D. Jacquet *et al.*, “A 3 GHz dual core processor ARM Cortex-A9 in 28 nm UTBB FD-SOI CMOS with ultra-wide voltage range and energy efficiency optimization,” *IEEE Journal of Solid-State Circuits*, vol. 49, no. 4, pp. 812–826, Apr. 2014.

[134] M. Blagojević, M. Cochet, B. Keller, P. Flatresse, A. Vladimirescu, and B. Nikolić, "A fast, flexible, positive and negative adaptive body-bias generator in 28nm FDSOI," in *Proceedings of the Symposium on VLSI Circuits*, Jun. 2016, pp. 61–62.

[135] B. Keller *et al.*, "Sub-microsecond adaptive voltage scaling in a 28nm FD-SOI processor SoC," in *Proceedings of the European Solid-State Circuits Conference*, Sep. 2016, pp. 269–272.

[136] Y. Lee, C. Schmidt, A. Ou, A. Waterman, and K. Asanović, "The Hwacha vector-fetch architecture manual, version 3.8.1," Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS-2015-262, Dec. 2015.

[137] Y. Lee, A. Ou, C. Schmidt, S. Karandikar, H. Mao, and K. Asanović, "The Hwacha microarchitecture manual, version 3.8.1," Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS-2015-263, Dec. 2015.

[138] J. L. Henning, "SPEC CPU2006 benchmark descriptions," *SIGARCH Computer Architecture News*, vol. 34, no. 4, pp. 1–17, Sep. 2006.

[139] *Tilelink 0.3.3 specification*. [Online]. Available: `https://docs.google.com/document/d/1Iczcjigc-LUi8QmDPwnAu1kH4Rrt6Kqi1_EUaCrfrk8/pub` (visited on 07/14/2017).

[140] *Denali memory interface IP*, Cadence. [Online]. Available: `https://ip.cadence.com/ipportfolio/ip-portfolio-overview/memory-ip` (visited on 08/17/2017).

[141] B. Keller, "Opportunities for fine-grained adaptive voltage scaling to improve system-level energy efficiency," Master's thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Dec. 2015.

[142] Z. Tan *et al.*, "RAMP gold: An FPGA-based architecture simulator for multiprocessors," in *Proceedings of the Design Automation Conference*, Jun. 2010, pp. 463–468.