Near-Threshold Computing: Past, Present, and Future

by

Nathaniel Ross Pinckney

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2015

Doctoral Committee:

Professor David Blaauw, Chair
Professor Trevor N. Mudge
Associate Professor Kevin P. Pipe
Professor Dennis M. Sylvester

Dedicated to family and friends,

young and old,

from home school

through camp

to college.

TABLE OF CONTENTS

DEDICATION ii

LIST OF FIGURES vi

LIST OF TABLES xiii

CHAPTERS

1 Introduction 1

2 Defining Near-Threshold 11
- 2.1 Motivation 11
- 2.2 Scaling Limiters 13
  - 2.2.1 Leakage 14
  - 2.2.2 Amdahl 17
  - 2.2.3 Architectural 19
- 2.3 Impact of Technology and Circuit Features on NTC 21
  - 2.3.1 Technology 21
  - 2.3.2 Process Variation 24
  - 2.3.3 Transistor Threshold Voltage 27
- 2.4 Related Work: Energy-Delay Product 28
- 2.5 Conclusions 31

3 NTC with Voltage Boosting 32
- 3.1 Motivation 32
- 3.2 Reexamining the Scaling Limiters 33
  - 3.2.1 Amdahl 33
  - 3.2.2 Architectural 35
- 3.3 Reexamining the Impact of Technology on NTC 36
- 3.4 Conclusions 39

4 Fast Boosting: Reclaiming Idle Cycles 41
- 4.1 Motivation 41
  - 4.1.1 Voltage Scaling 42
  - 4.1.2 Heterogenous Architectures 43

4.2 Methodology . . . 44
4.2.1 GEM5 Simulation . . . 44
4.2.2 Benchmarks . . . 44
4.3 Results . . . 45
4.3.1 Out-of-Order Core . . . 45
4.3.2 In-Order Core . . . 47
4.3.3 Fast Boosting . . . 50
4.4 Conclusions . . . 51

5 Shortstop: An On-Chip Fast Supply Boosting Technique . . . 53
5.1 Motivation . . . 53
5.2 Technique . . . 54
5.3 Measured Results . . . 61

6 Shortstop FC: A Fast Boosting Flip-Chip Implementation . . . 69
6.1 Motivation . . . 69
6.2 Improved Architecture . . . 69
6.3 Automatic Tuning Algorithm . . . 70
6.4 Physical Implementation . . . 76

7 Near-Threshold in FinFET Technologies: Impact of Process on Voltage Scalability . . . 78
7.1 Motivation . . . 78
7.2 Near-Threshold in 7nm FinFET . . . 81
7.2.1 Background . . . 81
7.2.2 Methodology . . . 83
7.2.3 7nm FinFET Results . . . 85
7.3 Analytical Model . . . 87
7.4 Device Characteristics . . . 94
7.4.1 Basic Device Characteristics . . . 96
7.4.2 Additional Process Characteristics . . . 100
7.5 Technology Trends . . . 109
7.5.1 Co-Optimized 7nm FinFET Device . . . 109
7.5.2 Comparison to Planar . . . 111
7.6 Additional Observations . . . 113
7.6.1 Variability . . . 114
7.6.2 Area Analysis . . . 118
7.7 Conclusions . . . 121

8 Near-Threshold Computing in FinFET Technologies: An Architectural Study 125
8.1 Motivation . . . 125
8.2 WAMI: An Example Application . . . 126
8.3 Circuit Analysis . . . 128
8.3.1 Methodology . . . 128
8.3.2 Power and Area Scaling . . . 129

8.3.3 Circuit Delay Scaling . . . 131
8.3.4 Voltage Scaling Scenarios . . . 133
8.3.5 Sensitivity to Parallelism Overheads . . . 137
8.4 System Analysis . . . 143
8.4.1 Compute Units . . . 143
8.4.2 Results . . . 146
8.5 Related Work . . . 148
8.6 Conclusions . . . 150

9 Summary and Future Work . . . 153
9.1 Contributions . . . 153
9.2 Future Work . . . 154
9.3 Related Publications . . . 156

BIBLIOGRAPHY . . . 158

LIST OF FIGURES

Figure

1.1 **Nominal supply voltage and energy density from 250 nm to 22 nm.** Supply voltage has not scaled past 130nm, consequently energy and power density has risen for each successive technology node. 2

2.1 **Clock frequency of a logic chain versus operating voltage.** Data is from a simulation of a chain of inverters in 32nm. As Vdd is further reduced and nears Vt, frequency scales exponentially with Vdd and performance degrades significantly. 15

2.2 **Total, static, and dynamic energy across Vdd for a 32nm process.** Energy is minimized when the slope of static and dynamic energy is equal. Below the minimum energy point, energy efficiency is not improved through power consumption may decrease. 16

2.3 **Speedup versus amount of parallelism demonstrating application-dependent Amdahl Overheads.** An ideal speedup corresponds to the black diagonal line, where the number of cores is exactly equal to the speedup. Increased parallelization overheads degenerate the line, so that most cores are needed to achievable comparable speedup. Three of the SPLASH-2 benchmarks are labeled for demonstration, with `Barnes` being close to ideal and `LUNC` very non-ideal. 18

2.4 **Vopt vs. Amdahl coefficient for all SPLASH-2 benchmarks (three labeled) in 32nm.** Higher Amdahl coefficient, *e.g.*less parallelizable workload, increases V_{opt}. A coefficient of 100% corresponds to a V_{opt} of max Vdd. 19

2.5 **Vopt in 32nm with leakage, Amdahl, and Architectural overheads.** Architectural and Amdahl overheads increase Vopt by no more than 200 mV. 21

2.6 **Vopt across technologies when including all three overheads.** Vopt has been trending downward with each generation and is 200-400mV above Vt for most benchmarks. 22

2.7 **Median energy gains and optimal number of cores, Nopt, when operating at Vopt as compared to nominal voltage for SPLASH-2 benchmarks.** Energy gains and optimal number of cores has trended downward from 180nm to 32nm as leakage has increased and the dynamic range available for voltage scaling has narrowed. 23

2.8 **Theoretical lowest Vopt across six technology nodes with leakage overheads only.** Vopt has been trending upward with each generation and becomes super-threshold in 90nm. 25
2.9 **Theoretical maximum energy-efficient parallelism Nopt and energy gains across six technology nodes with leakage overheads only.** Energy gain has reduced from 52× in 180nm to 6× in 32nm. 25
2.10 **Change of delay for total and local process variation of a single gate and a logic chain of 31 gates.** Longer gate chains average out mismatch. 26
2.11 **Increase in Vopt because of 3-sigma variation across generations.** Variation increases Vopt by 30mV-60mV on average for SPLASH-2 benchmarks. 27
2.12 **Vopt vs. Vt in 32nm with Amdahl and architecture overheads (top) and leakage only (bottom).** With ideal workloads, higher threshold voltage increases energy efficiency until Vopt is below Vt. With non-ideal workloads, a lower Vt improves energy efficiency. 29
2.13 **Energy and performance tradeoff in 32nm and marginal cost.** Tradeoff between performance and energy (top) in 32nm as Vdd is swept showing the energy minimum. Energy decreases towards Vopt / Eopt before static energy dominates. Marginal gains in energy for marginal decreases in frequency (bottom) show that energy gains diminish as voltage is scaled. The energy-delay optimal point is when the marginal gain is 1 while the energy-minimal Vopt is at 0. 30
3.1 **Comparison of Vopt vs. Amdahl serial coefficient with and without boosting.** Vopt vs. Amdahl coefficient for all SPLASH-2 benchmarks (three labeled) in 32nm. If parallel and serial portions of code are separated, the parallel portions operate at the lowest possible Vopt. Meanwhile, the serial portion will run at its optimal voltage of max Vdd. 34
3.2 **Comparison of NTC energy gain vs. Amdahl serial coefficient with and without boosting.** Boosting can increase energy efficiency, especially with a moderate Amdahl coefficient. Without boosting, no energy gain can be achieved above 50% serial coefficient. 36
3.3 **Vopt across SPLASH-2 benchmarks with and without boosting.** Architectural and Amdahl overheads increase Vopt by no more than 100 mV with boosting and 200 mV without boosting. However, this varies by benchmark. 37
3.4 **Vopt across technologies when including all three overheads without boosting.** 37
3.5 **Vopt across technologies when including all three overheads with boosting.** Boosting tightens the range of Vopt as Amdahl overheads are removed. 38
3.6 **Median energy gains and optimal number of cores, Nopt, when operating at Vopt as compared to nominal voltage for SPLASH-2 benchmarks.** Boosting regains $\sim 0.5\times$ additional energy on average for SPLASH-2 benchmarks. 40

4.1 **Traditional DVFS increases energy efficiency but requires hundreds of thousands of cycles to switch.** Show here from an industrial 65nm technology, energy consumption is reduced by 60% when operating at a low voltage. 42
4.2 **Big.LITTLE further reduces energy consumption at 50% performance.** Since Big.LITTLE includes architectural improvements, energy consumption improvements exceed straight voltage scaling for a fixed workload. 43
4.3 **Percent windows are idle for varying window sizes. GCC benchmark on out-of-order core.** Approximately 30% of windows are idle more than not (idle cycles 50%+) with window size of 100. 46
4.4 **Percent windows are idle for varying window sizes. Vortex benchmark on out-of-order core.** Approximately 25% of windows are idle more than not (idle cycles 50%+) with 100-cycle windows. 47
4.5 **Percent windows are idle for varying window sizes. Matrix multiplication benchmark on out-of-order core.** Approximately 3% of windows are idle more than not (idle cycles 50%+) with 100-cycle windows. 48
4.6 **Percent windows are idle for varying window sizes. GCC benchmark on in-order core.** Approximately 55% of windows are idle more than not (idle cycles 50%+) with window size of 100. 49
4.7 **Percent windows are idle for varying window sizes. Vortex benchmark on in-order core.** Approximately 55% of windows are idle more than not (idle cycles 50%+) with window size of 100. 49
4.8 **Ideal energy improvement using fast boosting with idleness $> 50\%$ for varying window sizes and averaged across four SPEC2000 benchmarks.** Both in-order and out-of-order results are shown. 51
5.1 **Shortstop high-level concept compared to other boost approaches.** Traditional DVFS uses off-chip regulators to adjust core voltage (top left), but takes 100s - 1000s of cycles to adjust. A PMOS header approach (bottom left) is very fast, requiring a handful of cycles, but destabilizes the high power supply through droop and ringing. Shortstop (right) uses PMOS headers, but adds a Vdirty power supply acting as a boost converter and an on-chip boost capacitor to boost a core in several cycles without destabilizing the Vhigh supply. 55
5.2 **Steps of Shortstop to boost core supply rail.** The five steps include: 1) The core is initially connected to the low voltage supply; 2) the boost cycle begins by charge sharing between a core and on-chip capacitor for a fast initial boost, while a dirty supply is shorted to ground to energize the wirebond/C4 parasitic inductance; 3) after charge sharing completes, the core is boosted the remaining amount to the high voltage by shorting to the Vdirty supply rail; 4) when the core reaches the target high voltage, it is switched to the stable Vhigh supply and the on-chip capacitor is connected to the Vdirty rail to quickly recharge; 5) once the on-chip capacitor is charged to its peak value, the Vdirty supply is disconnected and clamped to the Vhigh supply to prevent ringing. 56

5.3 **Complete system of delay generators, XOR trees, force blocks.** A shared ring oscillator, enabled with the *boost_go* signal is used to provide coarse delay adjustment between the delay generators. Since the delay generators count edges of the ring oscillator, all of the delay generators are synchronized for coarse adjustment. Within each delay generator is a tunable delay chain that provides fine adjustment. Configurable XOR trees are used to generate pulses from the delay generator step outputs. Force blocks directly before the switches are used to prevent glitching and safely reset the timing system. . . 58
5.4 **Principle of delay chain + XOR tree operation.** Three delay chains select points in time T1, T2, and T3 (top of figure). The XOR tree combine the three points to create a pulsed output (bottom of figure). 58
5.5 **Delay generator element.** Shortstop includes 16 of these delay elements. . 59
5.6 **Maskable XOR trees to select a subset of delay elements to control each switch.** An additional input controls polarity, whether the switch is active high or low. 60
5.7 **Force block to prevent glitching on switches.** The scan signal *force_bit* configures the value of the bit, while *force_en* actuates forcing the output to the switch to the programmed value. 61
5.8 **Test chip architecture for Shortstop.** The Shortstop test chip architecture includes on-chip variable boost and core capacitors to mimic large cores. An ARM Cortex-M3 and on-chip samplers are also included on the test chip. The on chip samplers are based on [61] can observe power supply transients by 20 point averaging a sampled supply voltage provided by an analog mux. The sampler was sized to hold values for roughly 1 ms with minimal leakage so that boosting experiments can be repeated and observed. 62
5.9 **Photomicrograph of 28nm test chip and chip specifications.** The chip measured 3.9 mm^2 and was wirebonded to CPGA and QFN packages. Variable core and boost capacitors dominate chip area, to emulate different boosting scenarios. 63
5.10 **Measured rail voltages for 3 nF core (QFN package).** For the M3 core, boost latency is improved by 1.7× and and droop reduced by 6×. 64
5.11 **Measured latency/droop improvement for varying core cap. (QFN package).** As core size decreases, Shortstop exhibits slightly increased gains against baselines. Supply droop is relatively constant with core size. 66
5.12 **Measured droop and latency for varying boost cap. (QFN package).** A boost cap sized for 30-40% of intrinsic core capacitance is sufficient to achieve most of Shortstop's performance gains. 67
5.13 **Measured performance improvements with varying packages and wirebond lengths.** Shortstop is fairly insensitive to these wirebond lengths. 68
6.1 **Shortstop FC improved topology.** Core areas include three instead of four switches, at the cost of one switch in the shared boost block, to reduce area overhead. 71
6.2 **Shortstop FC boost steps with improved topology.** 72

6.3 **Shortstop FC tuning concept.** A comparator is clocked on the switch of the core supply from Vdirty (Vboost with the improved architecture) to Vhigh. The comparators returns a '0' or '1' depending on if the core supply was above a droop threshold or not. 73
6.4 **Steps of the automatic tuning algorithm.** The four steps are used to minimize boost latency subject to a droop constraint on Vhigh. 74
6.5 **Top-level test chip layout.** Colored circles indicate flip-chip bumps. I/O signal bumps are around the chip perimeter, while power is supplied on all inner bumps. 77
7.1 **Power and area scaling normalized to 40nm.** 79
7.2 **Comparison of energy gain for performance sensitive and insensitive workloads, from previous planar study [10] to predicted 7nm FinFET device.** 86
7.3 **Total energy of a circuit is composed of dynamic and leakage energies. Total energy is minimized when dynamic and leakage slopes match.** 87
7.4 **Task completion time and leakage energy.** Task completion time increases as voltage is lowered, top. A larger serial coefficient requires more cores to maintain fixed latency. Leakage energy increases at low voltages since leakage power is integrated over longer periods of time (bottom). . . . 89
7.5 **Total energy of a task increases with a higher serial coefficient (P_s) since parallelism overheads limit voltage scalability as task latency is fixed.** Aggregate throughput for a fixed power budget is improved as energy per task is reduced. 95
7.6 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) for varying threshold voltage in 7nm FinFET.** 97
7.7 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) as DIBL coefficient increases in 7nm FinFET.** 99
7.8 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) as subthreshold slope becomes less steep in 7nm FinFET.** 101
7.9 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) with parasitic source/drain resistance decreasing in 7nm FinFET.** 102
7.10 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) with gate capacitance decreasing from baseline in 7nm FinFET.** 103
7.11 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) for within-cell parasitics and back-end-of-line parasitics from varying wire lengths in 7nm FinFET.** Throughput is normalized for each BEOL to 0.7V. 105
7.12 **Circuit delay scaling (top) and energy efficiency gain (bottom) for varying fin heights and different wire lengths in 7nm FinFET.** . . . 106

7.13 **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) with varying channel lengths in a 40nm planar technology.** . . . 108
7.14 **Comparison of 7nm FinFET predicted device before and after NTC optimizations.** . . . 110
7.15 **Energy gain and throughput across technology.** . . . 112
7.16 **Vopt across technology.** . . . 113
7.17 V_{opt} **and maximum aggregate throughput across technology nodes when considering global variation and binning is available.** Within-cell and 300TR wire parasitics included. . . . 115
7.18 V_{opt} **and maximum aggregate throughput across technology nodes when considering global variation and binning is not available.** Within-cell and 300TR wire parasitics included. . . . 116
7.19 **Relative increase in circuit delay variation normalized to nominal voltage in each node.** . . . 117
7.20 **Energy efficiency gain and optimal near-threshold voltage when including hold and setup time margining along with within-cell parasitics and 300TR wire loads.** . . . 119
7.21 **Aggregate throughput across technology when power constrained (solid line) and area constrained (dashed line) with varying supply voltage.** Within-cell and BEOL parasitics of a 300TR wire are included. . . . 120
7.22 **Energy efficiency gain and** V_{opt} **with and without an area constraint across technologies.** Within-cell and 300TR wire load parasitics included. 122
7.23 **Comparison of aggregate throughput at nominal and near-threshold supply voltages across technologies.** Within-cell and 300TR wire load parasitics included. . . . 123
8.1 **Workload pyramid for our wide-area motion imaging (WAMI) application.** Low levels (pixels) process lots of data in parallel, while higher levels are single tasked. . . . 127
8.2 **Power and area scaling normalized to 40nm.** Area scaling has outpaced power scaling, leading to a rise in power density. . . . 130
8.3 **Voltage scaling's impact on clock frequency of a circuit.** FinFET scales better than planar. . . . 132
8.4 **Throughput of single-task scenario (minimizing latency) across six technologies when power and area are constrained.** FinFET scaling and area density allow for improved single-task performance, which was not possible in older CMOS technologies. . . . 135
8.5 **Throughput of many tasks when latency constrained, without an area budget (top) and with an area budget (bottom).** FinFET is drastically better than planar for both energy gains and absolute throughput. An area constraint limits achievable gains, but newer technologies scale better because of higher packing densities. . . . 138

8.6 **Throughput of many tasks when task latency is unconstrained, without an area budget (top) and with an area budget (bottom).** When area constrained results are similar to fixed latency results, since needed parallelism was small. Without an area constraint higher energy efficiency can be achieved. 139
8.7 **Throughput gain and optimal operating voltage of a single task (minimize latency) in 7nm FinFET and 20nm Planar as percent serial is swept.** Nominal voltage is optimal above 33% serial. 142
8.8 **Proposed high-performance compute unit to handle heterogeneous WAMI workloads.** 144
8.9 **Summary of energy efficiency gains of WAMI kernels in futuristic 7nm FinFET compared to 20nm planar.** 152

LIST OF TABLES

Table

2.1 Gem5 simulation parameters used to measure architectural overheads. 20
2.2 Energy gain and optimal number of cores Nopt (in parenthesis) across SPLASH-2 benchmarks and technologies when including the three voltage scaling overheads. 24
4.1 Architectural Parameters for GEM5. 44
4.2 GEM5 Benchmarks Simulated. 45
4.3 Percentage of windows with idleness > 50% for varying window sizes. Both in-order and out-of-order are included. 50
4.4 Ideal energy improvement using fast boosting with idleness > 50% for varying window sizes. Both in-order and out-of-order are included. 50
5.1 Shortstop Test Chip Specifications. 65
7.1 **Voltage scaling operating scenarios, from ultra-low supply voltages to traditional nominal-voltage operation.** 81
8.1 Voltage scaling scenarios when latency is minimized, fixed, or unconstrained. Traditional computing is scenario one, while ultra-low voltage is scenario three and recent near-threshold definitions scenario two. 134
8.2 Summary of throughput (energy efficiency) gains from voltage scaling in 7nm FinFET and 20nm planar. Relative throughput and number of cores are also listed to compare across scenario or technology. 140
8.3 WAMI workload mapping to voltage scaling scenarios. 147
8.4 Pre-scale estimates of compute unit components running in 40nm at nominal voltage. 147
8.5 Components of compute unit in futuristic 7nm FinFET technology at nominal operating voltage. 148
8.6 7nm estimates after voltage scaling and increasing number of units until power matches nominal Vdd. 149

CHAPTER 1

Introduction

Moore's law [1] has historically enabled increased microprocessor performance with each technology node while maintaining constant power density. However, conventional voltage scaling has slowed in recent years, limiting processor frequency to meet power-density constraints. As transistors have become leakier, it has become more difficult reduce the threshold voltage and hence supply voltage scaling has stagnated as well in order to maintain sufficient overdrive and performance [2]. This deviation from constant-field scaling theory [3], combined with continuing scaling of transistor density has resulted in an increase in power density (W/mm^2) beyond the 130nm node, as shown in Figure 1.1.

Consequently, processor designs have added more cores without significantly increasing frequency, leading to a prevalence of chip multiprocessors (CMP) [4] in contemporary commercial architectures. However, because the number of cores has been increasing geometrically with each process node while die area has remained fixed, the total chip power has again started to increase, despite relatively flat core frequencies. In practice, the maximum allowable power dissipation of a single die is constrained by thermal cooling limits. Hence, the consequence of supply-voltage stagnation is a limit on the number of cores that can be active simultaneously on a die and, thus, a limit on the attainable performance of a modern CMP.

Recent work has observed that we are at a point where not all cores can be simultaneously active at full voltage and clock frequency without exceeding the thermal design budget [5]. Consequently, at any given time large sections of a chip will remain inactive in order to not

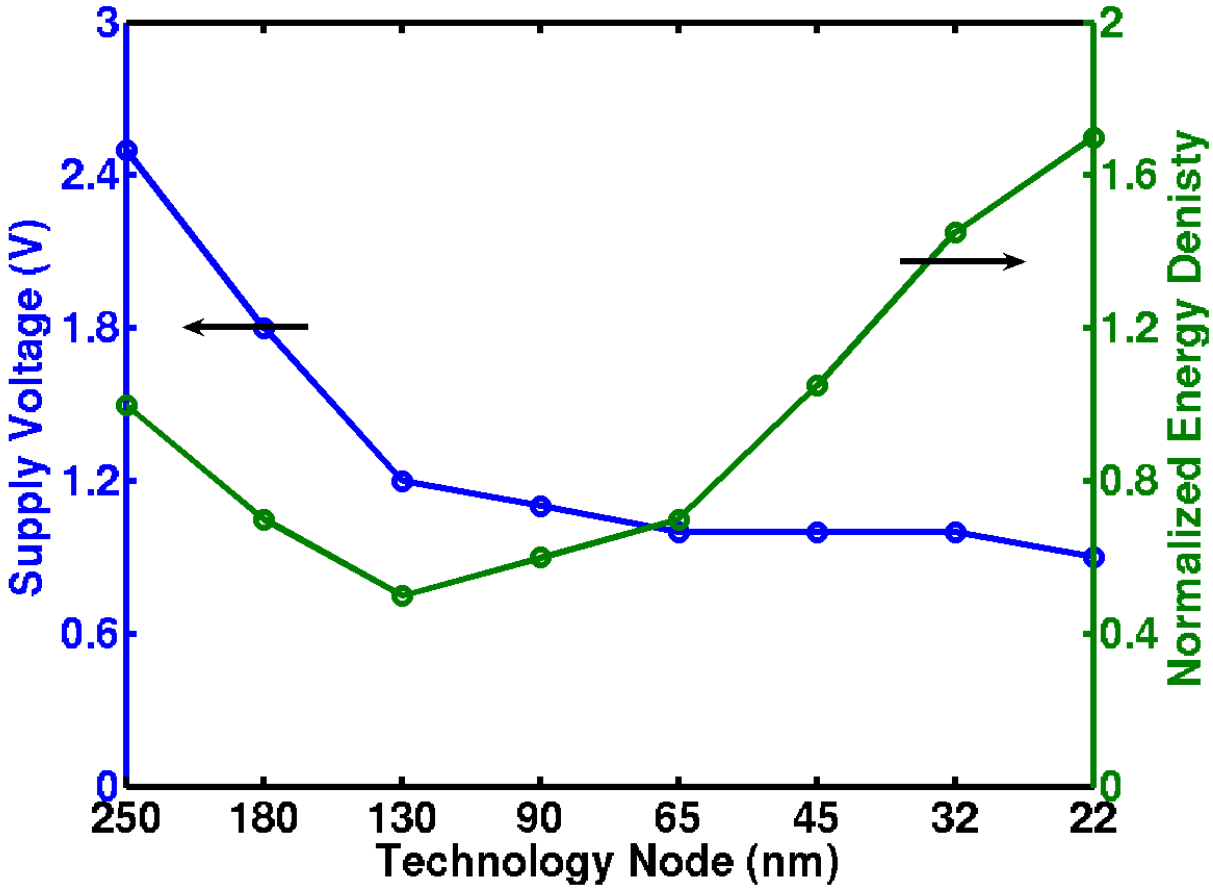


Figure 1.1: **Nominal supply voltage and energy density from 250 nm to 22 nm.** Supply voltage has not scaled past 130nm, consequently energy and power density has risen for each successive technology node.

exceed thermal limits of the package and cooling system. This scenario, dubbed *dark silicon*, has shown that the percent of chip inactivity is increasing each generation and the majority of chip area in a CPU could be dark by as soon as 2016 [5]. As a result, the most recent server-class CMPs incorporate extensive power-gating methods to turn off idle cores to free the thermal budget for active cores [1]. Because modern CMP performance is now limited by power and not die area, a paradigm shift is needed in CMP design: cores are plentiful, but power for them is not.

To overcome the resulting energy and power dissipation barriers, energy efficiency can be improved through aggressive voltage scaling, and there has been increased interest in operating at *near-threshold computing* (NTC) supply voltages [6–14]. In this region sizable energy gains are achieved with moderate performance loss. Lost performance, due to reduced clock frequency from increased logic delay, can be regained by parallelizing across cores [5, 10, 11, 15]. Thus, dark silicon becomes *dim silicon* [15] through lowering the supply voltage to near-threshold and trading off single-core performance for many cores operating in parallel at a low voltage. Near-threshold has extended beyond academic research and is being developed by industry leaders, such as Intel [16, 17] and Qualcomm [18, 19].

Near-threshold computing has a wide impact on design, from algorithmic optimizations and high-level systems architecture to circuit techniques and device-level tuning. A foundational near-threshold work published in a 2010 volume of Proceedings of the IEEE [6] has been cited by 233 publications in 6 years. These citations include numerous architectural papers [15, 20–22], circuit papers [23, 24], OS/compiler papers [25, 26], and device papers [27, 28].

Understanding sources of energy consumption is a circuit is critical for understanding how near-threshold improves energy efficiency. To start, energy is categorized into two main types within a circuit: static and dynamic. Static is continuous current consumption, regardless of circuit switching speed or frequency, due to leakage through transistors. In most CMOS logic circuits the primary leakage path is through the source-drain nodes, though gate leakage can also be substantial. Static energy can be expressed as

$$E_{static} = I_{leak} V_{dd} T_{task} \tag{1.1}$$

where I_{leak} is the leakage current, V_{dd} is the operating voltage, and T_{task} is the time to complete a task in which the static energy is integrated. Note that I_{leak} is dependent on V_{dd} because of effects such as drain-induced barrier lowering (DIBL) which modulates the threshold voltage of a device, V_t. I_{leak} has an inverse exponential dependence on V_t, so a lower V_t will increase leakage current exponentially. In more recent technologies exhibiting short-channel effects, increasing V_{dd} will reduce V_t because of DIBL and therefore further increase static energy consumption beyond the linear dependence on V_{dd} [2].

Dynamic energy, which is essentially the 'working' energy of switching capacitances in a circuit, can be expressed as

$$E_{dynamic} = CV_{dd}^2 \tag{1.2}$$

where C is the capacitance that must be switched in order to complete a task. There are many sources of capacitance in a circuit, but the main ones to consider for this analysis are gate capacitance (capacitance of the transistor's gate) and parasitic capacitance, which include transistor junction capacitance and wire capacitance to interconnect logic gates. The time or latency needed to complete the task can be modeled by the alpha power law as

$$T_{task} = \frac{V_{dd}}{(V_{dd} - V_t)^\alpha} \tag{1.3}$$

where α is historically 2 but is close to 1.3 in more recent planar technologies with bad short channel effects [29].

A typical logic circuit, such as the datapath in a core, has a very high *activity factor* meaning there is a high probability that a circuit node will switch during a given cycle. This implies that its dynamic energy consumes the majority of power, as on any given cycle much of the gate and parasitic capacitances are switching. As can be seen from Equation 1.2, $E_{dynamic}$ has a quadratic relationship on V_{dd}. Considering only $E_{dynamic}$, lowering V_{dd} can greatly reduce $E_{dynamic}$. However, as we shall see throughout this dissertation many factors, such as leakage, performance, parallelism, and variation, limit voltage scaling and the minimum obtainable energy in a circuit.

Digital voltage and frequency scaling (DVFS) is a conventional method to adjust CPU operating voltage and frequency, typically controlled by operating system software to reduce

frequency and voltage under low CPU load conditions. This differs from near-threshold in a number of different ways. First, the lowest voltage in most DVFS implementations is still much above the threshold voltage. Second, DVFS designs are traditionally still targeted for nominal voltage operation and voltage is lowered only when CPU load is minimal. Finally, DVFS is generally very slow to change, on the order of 10,000s of CPU cycles to make voltage and frequency adjustments [9]. Near-threshold, on the other hand, is meant to be the primary operating condition and any nominal voltage operation is used as needed. Fast boosting techniques, discussed below, are proposed to quickly adjust the voltage of a core in 10s to 100s of cycles, much faster than traditional DVFS implementations.

Existing power savings techniques that are heavily used, even in commercial products, are power gating and clock gating. However, these techniques are not mutually exclusive with near-threshold and could be used within a near-threshold implementation. Power gating uses PMOS headers or NMOS footers to turn off cores or other large circuit block when they are not in use. Typically all state data is lost when a block is powered down, thus there are overheads to reinitialize the block when it is powered on. Thus, power gating is not ideal for continuous operation but can be used intermittently.

Clock gating disables clock signals to unused blocks to save energy that would have been used to switch capacitance internal to that block, including gate and parasitic capacitance on both clock signals as well as any associated data or control signals. Unlike power gating, clock gating can respond very fast (single cycle) to the needs of a block since it is uses logic gates to impede the propagation of a clock signal, as oppose to waiting for the intrinsic capacitance of a block to be charged. Additionally, clock gating loses no state as all sequential elements remained powered. As mentioned above, clock gating is an orthogonal technique to near-threshold that could be used to save additional power.

Ultra-low voltage (ULV) or subthreshold designs are related prior work, but often target minimizing power rather than energy, especially for specific applications that do not have demanding performance requirements, such as some sensor nodes [30]. Certain algorithms parallelize very well and lost performance from slow clock frequency can be regained through parallelism. An early example is a 180 mV, 164 Hz, 1024-point Fast Fourier Transform processor (FFT) fabricated in 0.18 µm [31]. While the processor was targeted and designed

for 180 mV operation, the authors found energy was minimized near threshold consuming 155 nJ/transform at 350 mV . A more recent 65nm ULV, 1024-point FFT design operated at 0.27V and achieved 17.7 nJ/transform [32] through aggressive pipelining of the design. Another example of a low-voltage hardware accelerator is a computer vision feature extraction engine implemented in 28nm CMOS and operating at a very slow 27Mhz, but providing 3.5× energy efficiency than previous designs [33]. Within this work near-threshold is viewed as a flexible approach to general purpose computing, and not limited to highly parallelizable hardware accelerators.

This dissertation advances near-threshold computing by providing a methodical definition of "near threshold" and an in-depth examination of NTC across past and future CMOS process technologies. By systematically defining near-threshold, trends and tradeoffs between different technologies are closely analyzed, lending insight in how best to design and optimize near-threshold systems. Also proposed in this dissertation are techniques for fast voltage boosting, in which a core's operating voltage is dynamically adjusted depending on workload, further improving energy efficiency.

In Chapter 2, we start by investigating the limit of voltage scaling together with task parallelization to maintain task completion latency. When accounting for Amdahl and architectural parallelization overheads, minimum task energy is obtained at "near threshold" supply voltages across six commercial technology nodes from 180nm to 32nm. Operating in near-threshold improves overall energy efficiency for a power-constrained CMP by 4× for a representative set of scientific benchmarks tested. A study of technology trends show near-threshold becoming less effective with newer technologies as transistor dynamic voltage range decreases and leakage worsens. Additionally, an initial study of differences between near-threshold high activity circuits, such as core logic, and low activity, such as SRAM, is given.

A drawback of near-threshold is task parallelizsation, as some tasks parallelize better than others, and the energy efficient operating voltage changes with ease of parallelism. Thus, changing workloads makes a design targeted to operate at a fixed voltage during design time impractical for most applications. Additionally, within a task it is beneficial to be able to "boost" the voltage of a core quickly to meet latency or performance constraints

of an application. As an example, consider a MapReduce [34] style task running which operates in two parts. First, the map phase splits the task in parallel across eight cores, each processing their own section of data. Second, the reduce phase coalesces data from the map phase using a single core. As this single-core map phase becomes a processing bottleneck, it is useful to rapidly raise the voltage of this core near-threshold, optimized for parallel processing during the map phase, to nominal Vdd. At nominal Vdd the single-task performance rapidly increases to help overcome serial bottlenecks.

The NTC analysis in Chapter 2 is extended in Chapter 3 by including voltage boosting into the energy optimizations. Therefore, we examine improvements in energy efficiency and parallelism when serial portions of code can be overcome through quickly boosting the operating voltage of a core. When accounting for parallelization overheads, minimum task energy is obtained at "near threshold" supply-voltages across six commercial technology nodes and provides 4× improvement in overall CMP performance. We find boosting is most effective when the task is modestly parallelizable, but not highly parallel or serial.

Even for single threaded applications fast voltage boosting techniques may be useful for fine-grained periods of low core activity. In Chapter 4 we characterize four SPEC2000 benchmarks and find, on average, a 30% improvement in energy-efficiency on an out-of-order core. The extracted idleness occurs at a finer granularity than traditional DVFS and heterogeneous architectural techniques can provide.

A novel core supply boosting technique, called *Shortstop*, is proposed in Chapter 5 and boosts a 3nF core in 26ns while maintaining acceptable supply voltage droop. The technique is proposed as an alternative to on-chip regulators which require expensive inductors. Instead Shortstop leverages the innate parasitic inductance of a dedicated dirty supply rail is used as a boost-converter and combined with an on-chip boost capacitor. Shortstop is demonstrated in a wirebond implementation and is able to boost a core up to 1.8× faster than a header-based approach, while reducing supply droop by $2-7\times$, and can be used in near-threshold computing to overcome serial code bottlenecks.

The initial prototype of Shortstop in Chapter 5 is demonstrated in a wirebonded package. However, modern day processors use flip-chip packages via a bumping technology, such as controlled collapse chip connect (C4) [35], which meets growing I/O demands, delivers

more reliable power, and consumes a smaller area footprint. Because Shortstop depends on parasitic inductance of a package, which are much lower for flip-chip compared to wirebond, Chapter 6 introduces a second-generation *Shortstop FC* design aimed at flip-chip implementations. Shortstop FC is a much more modular test platform including sixteen emulated cores (composed of current sources and capacitors) that can be configured into usage scenarios. The physical implementation of Shortstop FC is improved, with more robust power delivery and distributed power switch headers and footers. Lastly, an automatic tuning algorithm is proposed to quickly tune a Shortstop system to raise the voltage of a core in the quickest time possible.

Foundries have embarked on a fundamental switch from planar transistors to FinFET at the $22-16$ nm node and below, opening a new chapter in Moore's law. With the introduction of FinFET devices, the semiconductor industry is making a dramatic shift to devices that exploit the third dimension (3D), and the full impact of this change is still being assessed by the design community. FinFET differs significantly from planar technology, with much improved channel characteristics, which have the potential to dramatically improve near-threshold performance. The impact of FinFET on voltage scaling and, in particular, near-threshold operation has not been studied and is of great interest as device engineers work to optimize the next generation FinFETs, circuit designs develop clever techniques to improve energy efficiency while mitigating variation, and architects develop increasingly efficient arrangements of cores and peripherals to be placed on a chip.

In Chapter 7, FinFET's impact on near-threshold is explored. Using six technology models, three planar and three FinFET, from 40nm to 7nm, we examine the impact of device characteristics on near-threshold efficiency and performance. We start by presenting an analytical model of performance-sensitive, near-threshold energy gain. Next, we analyze how individual device characteristics, such as transistor threshold voltage, subthreshold slope, DIBL, and back-end-of-line parasitics, impact near-threshold. With this knowledge the six technology nodes are targeted for near-threshold, and we show how FinFET differs from planar technologies and what the trends are for the most recent and future technology nodes. Finally, we delve into more detail on how to include variation and area in near-threshold, and we explore additional observations of FinFET's advantages over planar. Our main finding

is that FinFET's improved channel characteristics more than double energy efficiency gains compared to recent planar nodes, while area density improvements allow near-threshold operation to be realized. Thus, the switch to FinFET CMOS technology allows for a return to strong voltage scalability, reversing trends seen in planar technologies, while dark silicon has created an opportunity to add cores for parallelization.

Chapter 8 expands the scope of Chapter 7 by studying how three different processing elements— general-purpose cores, throughput-oriented cores, and accelerators— are best designed in 7nm FinFET compared to planar, including key voltage scaling techniques to improve energy efficiency. Using circuit simulations we examine efficiency gains from three voltage scalability scenarios, maximizing aggregate task throughput, single task throughput, or a mix of the two, and use these results to guide architectural design. Unlike prior studies, we also consider overall system area constraints in our methodology.

While the techniques in Chapter 8 generalize for many applications, we illustrate how they could be used to design a system for wide-angle motion imagery. The proposed system uses a heterogeneous mix of the three processing elements and operating voltages to achieve high energy efficiency for data parallel imaging tasks, yet leverages less-aggressive voltage scaling to improve performance of serialized event recognition tasks. FinFET CMOS technologies offer high area density relative to practical power constraints, and very good voltage scalability, making them ideal to realize high-performance near-threshold embedded systems. Our final results show an improvement of $2.6 - 8.9\times$ in energy efficiency for fixed and unconstrained latency tasks, while latency reductions of up to 20% are possible for tasks in which latency is minimized for a 5% serial coefficient. For single tasks with less serial code, up to a 65% latency reduction is possible.

Concluding this dissertation is Chapter 9, which enumerates remaining work to be done, including testing of Shortstop V2 and completion of the FinFET NTC study. A list of related publications, that were that were generated as a product of this dissertation, is shown at the end of Chapter 9.

Key observations of this thesis include:

- In order to maximize energy efficiency, near-threshold computing requires supply voltages lower than in conventional dynamic voltage scaling.

- For performance sensitive applications, near-threshold is able to improve energy efficiency as long as the technology scales reasonably well with voltage (good circuit delay scaling) and excess area is available to add cores for parallelization.

- Short-channel effects, such as those seen in recent planar technologies, adversely impact circuit delay scaling. Thus, recent planar technologies are not ideal for near-threshold even if area is available for additional cores.

- FinFETs feature good circuit delay scaling and high area density. Thus, FinFETs are ideal for near-threshold performance-sensitive applications.

- Latency of a single task is improved by up to 65% in FinFET, even when imposing area constraints, if at least 90% of the task is parallelizable. This was not possible in planar technologies, either because of poor circuit delay scaling (in recent planar technologies) or poor area density (in older planar technology).

CHAPTER 2

Defining Near-Threshold

2.1 Motivation

The first consequence of this supply voltage stagnation has been the inability to increase processor frequency while still meeting power density constraints. Instead, processor designs have added more cores without significant increase in their frequency, leading to a prevalence of chip multiprocessor (CMP) [4] in contemporary commercial architectures. However, since the die area of a server class chip has remained approximately constant at $\sim 300 - 600\,\mathrm{mm}^2$, and since the number of cores has been increasing geometrically with each process step, the total chip power has again started to increase, despite relatively flat core frequencies. In practice the maximum allowable power dissipation of a single die is constrained by thermal cooling limits and is roughly 150W without advanced cooling technologies [1]. Hence, the second consequence of supply voltage stagnation is a limit on the number of cores that can be active simultaneously on a die and thus the maximum attainable performance of a modern CMP.

For instance, a $600\,\mathrm{mm}^2$ CMP could accommodate 23 Intel Westmere cores [1] in 22nm CMOS, which would dissipate 211W when all simultaneously executing, far exceeding the practical thermal dissipation limit. This would result in 40% of the cores (9 of 23) being idle. The problem of power-constrained core under-utilization has been recently observed in the literature and is sometimes referred to as *dark silicon* [5]. If scaling trends continue to 16nm, a similar CMP would consist of 46 cores, consume a max of 300W, and 50% of the cores

(23 of 46) would be idle. As a result the most recent server-class CMPs have incorporated extensive power gating methods to turn off idle cores to free thermal budget for active cores [1].

Because modern CMP performance is now limited by power and not die-area, it is necessary for a paradigm shift in CMP design: cores are plentiful but powering them is not. The overall CMP performance can be best measured as task throughput: the number of completed tasks per second. In a power-constrained CMP, the task throughput is limited by the number of tasks that can be simultaneously active on the CMP within the thermal constraint. Thus, if we are able to lower task energy, the number of simultaneous tasks on the CMP (and hence activated cores) can be increased, improving task throughput.

The most effective knob for reducing energy consumption of a task running on a microprocessor is lowering the operating voltage. In tandem, processor frequency is reduced and task completion latency is increased. This type of voltage reduction has been widely used in DVFS, but since it impedes task completion latency, it is not generally applicable for high-performance applications. To address this, parallelization can be used to counteract lower clock frequencies and maintain latency. In this approach, the execution code of the task is parallelized so that the task executes on multiple cores in the CMPs, each operating at a lower frequency and voltage. In this way, the completion time remains the same as when the same task was executed serially on a single core at full voltage, while significant savings in total energy expended for completing the task is obtained. This reduction in energy consumption in turn allows more tasks to be executed on the CMP, thereby increasing overall CMP task throughput.

This combined voltage / parallelization approach is similar to the simpler circuit based parallelization approach proposed earlier in [36] which trades-off energy for latency. The method envisioned here instead parallelized the algorithm and thereby maintains task latency while still obtaining energy improvement. In fact, with the emergence of CMPs, many key applications are currently being parallelized by software developers. However, parallelization entails a number of overheads, which tend to increase as the task is parallelized into smaller subtasks. These overheads limit the obtainable energy improvement from the proposed approach, as the overhead eventually dominates over the quadratic energy gains

from voltage reduction. Hence, there exists a minimum energy point, at which a task is optimally parallelized and voltage scaling reaches its efficiency limit.

To our knowledge, no systematic analysis has been performed to determine where this energy minimum lies. Hence, in this chapter, we study this energy minimum, its associated energy gains, core operating voltage and task parallelization. We model three key factors limit energy-efficient parallelization in modern CMOS technologies: The leakage of a transistor, Leakage Overheads; the inability to achieve ideal code parallelization, Amdahl Overheads; the impact of coherence, interconnect, and memory system design, Architectural Overheads. All these overheads are interrelated and limit the obtainable energy efficiency gains from voltage scaling, the optimal energy voltage (Vopt) and the number of parallel subtasks required for frequency drop compensation (Nopt). In addition, we study the behavior of Vopt and Nopt across process nodes from 180nm to 32nm technology, using commercial process models.

Our key finding is that when realistic application-dependent overhead is included the optimal operating voltage is near threshold, roughly $200-400$mV above the threshold voltage, and that this voltage range is valid across the six generations of industrial technologies as well as across transistor Vt selection. When accounting for all three overheads, operation at Vopt yields an energy efficiency gain of $4\times$ compared to operation at nominal voltage in 32nm and therefore allows a $4\times$ increase in CMP task throughput under thermal constraints. Additionally, we find the maximum amount of energy-efficient parallelism, Nopt, across SPLASH2 benchmarks has a median value of approximately 12. Because running at lower supply voltage increases sensitivity to variation, we also explore the impact of variation on Vopt and include this in our analysis.

2.2 Scaling Limiters

There are three key limiters to energy-efficient scaling when a task is parallelized to maintain constant latency: leakage, Amdahl, and architectural. Each of these contributes to increased minimum energy and raises the energy-efficient operating point Vopt. The three key limiters are analyzed in the following subsections.

2.2.1 Leakage

First, we will assume a task can be perfectly parallelized across cores to compensate for frequency loss at a lower voltage, and the only non-ideality from running at a slower clock frequency is transistor leakage. It is well known that reducing the supply voltage initially increases energy efficiency of a computation quadratically, yielding dramatic energy efficiency gains [31]. In the last 7 years, it has also been shown that leakage energy poses a fundamental limiting factor to energy efficiency gains through voltage reduction [37, 38]. The required energy to complete a task can be divided into two categories, dynamic and static, and the classic relationship between energy and operating voltage is:

$$E_{total} = E_{dynamic} + E_{static} = CV_{dd}^2 + I_{leak}V_{dd}T_{task}$$

Dynamic or active energy is the energy consumed in charging and discharging the transistor and interconnect capacitances associated with the task being executed. Static or leakage energy is due to the always present subthreshold and gate oxide currents integrated over the time Ttask to complete a task. While dynamic energy represents the energy needed to complete a task, static energy is parasitic and only poses an overhead on the computation. Although leakage can be mitigated in standby mode using techniques such as power gating and body biasing, it is more difficult to do so in active mode. Hence, leakage forms an unavoidable and fundamental limit on energy-efficiency.

To understand how leakage energy scales with Vdd, clock frequency scaling must be considered since as clock frequency is reduced the time to complete a task increases. For illustration purposes, the relationship between operating voltage and clock frequency is approximately:

$$\frac{1}{T_{task}} \propto f \propto \frac{(V_{dd} - V_t)^\alpha}{V_{dd}}$$

where V_t is the threshold voltage and α is process dependent but close to 2. For our results we simulated industrial transistor models in Cadence Spectre to obtain energy and performance. The canonical circuit topology was a chain of 31 fanout-of-4 inverters along with dummy devices for realistic input and output slew rates. The logic activity factor was

chosen as 15% to emulate a core where 15% of the logic gates switch on average per clock cycle [39]. In addition chains of other types of logic gates were simulated to confirm that the result obtained for an inverter chain were representative of other logic structures as well.

Initially, when Vdd is large relative to Vt, frequency scales proportionately to Vdd as shown in Figure 2.1. As Vdd is further reduced and nears Vt, frequency scales exponentially with Vdd because the transistor is no longer fully activated. Instead the transistor drive current comes from subthreshold leakage current which scales exponentially with the gate-to-source voltage, and thus exponentially with Vdd.

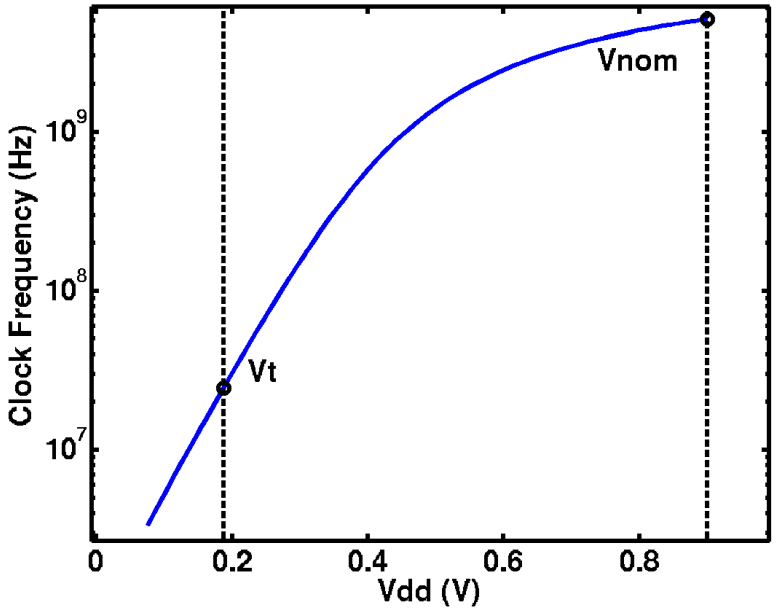


Figure 2.1: **Clock frequency of a logic chain versus operating voltage.** Data is from a simulation of a chain of inverters in 32nm. As Vdd is further reduced and nears Vt, frequency scales exponentially with Vdd and performance degrades significantly.

As operating voltage is lowered, the static energy increases since the time to complete a task scales inversely with clock frequency. Eventually, at very low voltages, static energy dominates over dynamic energy, Figure 2.2. The operating voltage where total energy is minimized is called Vopt and occurs when the derivatives with respect to Vdd of the two energies are equal, $(dE_{static})/dVdd = (dE_{dynamic})/dVdd$ [38]. Beyond this point static energy increases more rapidly than dynamic energy decreases, and the total energy increases away

from the energy minimum. For a 32nm node, Vopt when considering leakage overheads is 300 mV.

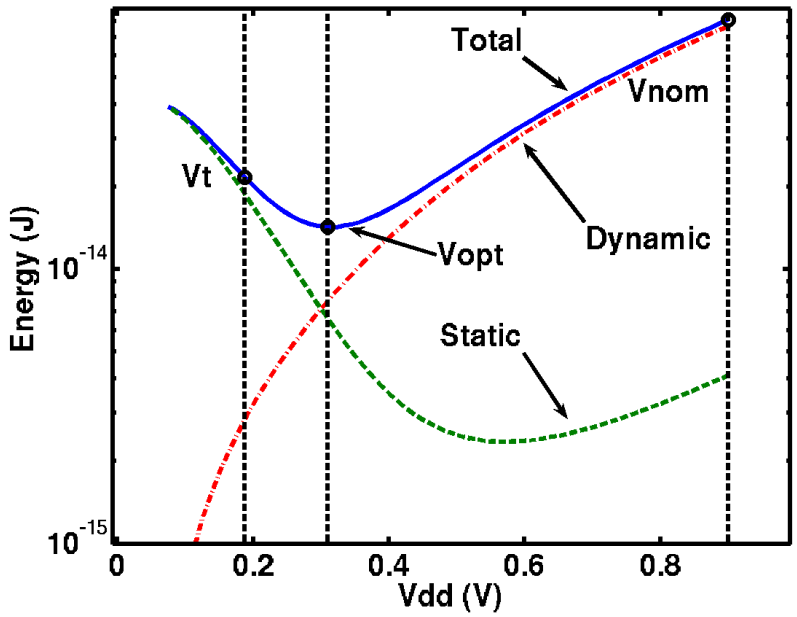


Figure 2.2: **Total, static, and dynamic energy across Vdd for a 32nm process.** Energy is minimized when the slope of static and dynamic energy is equal. Below the minimum energy point, energy efficiency is not improved through power consumption may decrease.

In recent years, several sensor processors that operate at this Vopt, which typically lies below the device threshold voltage, have been designed and demonstrated as much as 10× energy efficiency gains over operation at nominal supply voltage [30, 40]. However, these sensor processors also incur phenomenal frequency loss, often operating at clock frequencies of 100s of kHz.

To fully compensate for a frequency loss of X because reduced voltage operation, a task with k instructions must be parallelized across X cores. If frequency is not compensated the total execution time would increase proportionally to $X*k$. But, since the task is parallelized, each of the X cores runs k/X instructions so the total execution time is $X * (k/X) = k$. Thus, no performance is lost from parallelizing. While most scientific and high-performance applications have been parallelized to operate on CMPs, it is not practical to recover a factor of 100's or 1000's in frequency loss without enormous parallelization overheads. Therefore,

Vopt, considering optimization overheads will be at a higher voltage level, which will be discussed in the next subsection.

2.2.2 Amdahl

As discussed earlier, scaling voltage is essential in the CMPs to achieve maximum computational performance for a fixed thermal budget, since the number of simultaneous tasks that can fit in a TDP is directly proportionate to the energy efficiency of the task. When scaling supply voltage for a latency-sensitive task, slower clock frequency can be compensated by executing the task in parallel across more cores. For real applications the process of subdividing a task includes non-idealities, such as serial portions of code, and thus incurs parallelization overhead. To compensate a task of k instructions for a frequency loss of X requires X cores (each running k/X instructions) plus m additional instructions of parallelization overhead. These extra m instructions consume additional energy, penalizing lower voltage operation, and therefore increase Vopt. Hence, parallelization overheads compound the impact of leakage overheads which limits the voltage scalability of a latency-sensitive task. Compensating below Vopt by further subdividing the task results in a net energy increase due to leakage and parallelization overheads.

The well-known Amdahl's law [41] shows that speedup of algorithms as they are parallelized over an increasing number of cores is limited by the parallelizable portion of the code and by new code introduced to initialize and decompose the program. Speedups are therefore bounded asymptotically as parallelization increases because the serial portion eventually dominates. These overheads will be referred to as Amdahl overheads and include only the impacts of algorithmic parallelization.

The gem5 [42] system simulator is used to evaluate the impact of Amdahl overheads on a Network-on-Chip (NoC) system. We evaluated the SPLASH-2 benchmark suite which is a set of highly parallelized scientific algorithms applicable to CMPs. Each core is an Alpha architecture with one instruction-per-cycle running at 1GHz. To separate Amdahl overheads from additional architectural non-idealities, we simulated the system with infinite intercon-

nect bandwidth and an ideal memory with 1 cycle latency. Architectural non-idealities are addressed in the next subsection.

The effective speedup of parallelizing by running on 1 to 64 cores is shown in Figure 2.3. For illustration only three representative benchmarks are labeled, but the entire suite is plotted in the figure. Some benchmarks, such as `Barnes`, have nearly ideal speedup indicating very little Amdahl overheads and perfect parallelization. Other benchmarks, such as `LUNC`, reach a speedup of only 10 with 64 cores indicating a high percentage of serial code. These benchmarks represent a range of parallelized scientific applicable to CMPs and, as the number of CMP cores continues to increase, more high-performance applications will be similarly parallelized.

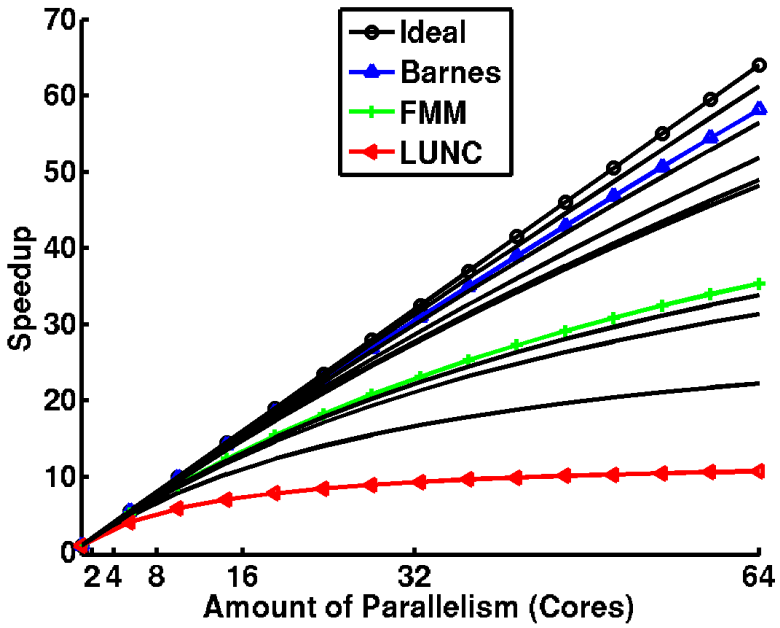


Figure 2.3: **Speedup versus amount of parallelism demonstrating application-dependent Amdahl Overheads.** An ideal speedup corresponds to the black diagonal line, where the number of cores is exactly equal to the speedup. Increased parallelization overheads degenerate the line, so that most cores are needed to achievable comparable speedup. Three of the SPLASH-2 benchmarks are labeled for demonstration, with `Barnes` being close to ideal and `LUNC` very non-ideal.

Amdahl's Law [41] gives $Speedup = n/(1 - P_s + P_s n)$, where n is the number of cores parallelized over and P_s is the Amdahl serial coefficient. We fitted the SPLASH-2 benchmark speedups to Amdahl's law and applied it to the voltage scaling calculations to obtain Vopt

when considering non-ideal parallelization. The benchmarks were parallelized to fully compensate for frequency loss from lower voltage operation. Figure 2.4 shows Vopt increasing in 32nm due to Amdahl overheads. When Amdahl overheads are added, the Vopt operating range for most overheads is 25-150 mV above the leakage overheads only case. Although the serial coefficient is highly application-dependent, the range of Vopt for the benchmarks is small, varying by only 150mV. If the serial coefficient is 100% (e.g., none of the code is parallelizable) then nominal voltage would be optimal.

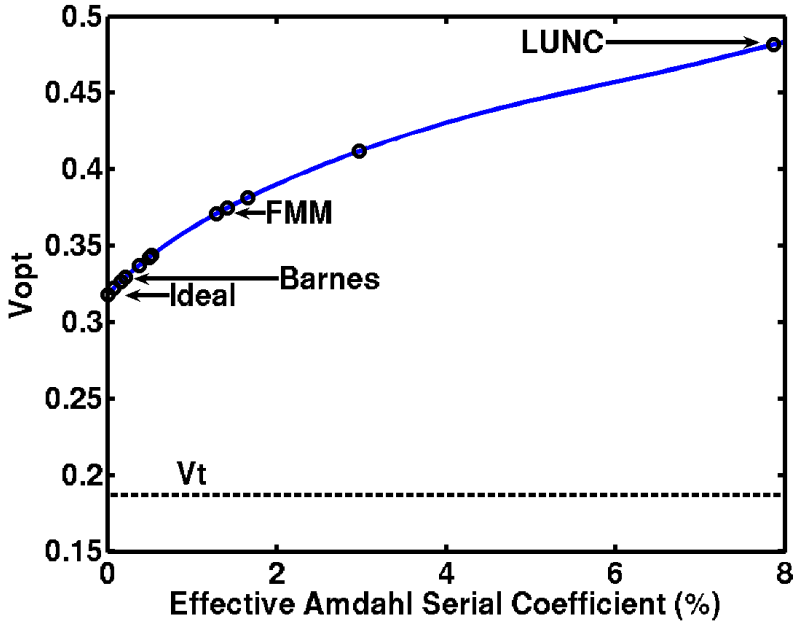


Figure 2.4: **Vopt vs. Amdahl coefficient for all SPLASH-2 benchmarks (three labeled) in 32nm.** Higher Amdahl coefficient, *e.g.*less parallelizable workload, increases V_{opt}. A coefficient of 100% corresponds to a V_{opt} of max Vdd.

2.2.3 Architectural

Architectural features, such as coherency, inter-core communications, and cache pollution, further add overhead to a CMP system as voltage is reduced and a task is parallelized. Furthermore, application memory access patterns can affect overhead. For example, a subtask competes for L2 cache resources and may evict another subtask's data. Coherence overhead is added when a multiple subtasks share a single block of data. Communication

overhead is increased when there is heavy communication between distant cores on an NoC because data must transverse multiple hops.

To quantify architectural overheads, the SPLASH-2 benchmarks were simulated with gem5 as in Section 2.2 but the configuration was changed to add non-ideal memories, caches, and interconnect. The NoC simulations were run using a tiled Mesh topology where each tile contains a core, private L1 caches, and a slice of a shared L2 cache. A MOESI directory protocol is used to maintain coherence. Table 2.1 lists the detailed simulation parameters.

Table 2.1: Gem5 simulation parameters used to measure architectural overheads.

Feature	Description
Cores	1 to 64 one-IPC Alpha cores @ 1GHz
L1 Caches	32 kB, 1 cycle latency, 4-way associative, 64-byte line size
L2 Caches	Shared 1MB divided evenly between cores, 10 cycle latency, 8-way associative, 64-byte line size
Interconnect	2-GHz Routers, 128-bit, 2-stage routers, 50 cycle-access to main memory

Architectural overheads from memory and interconnect non-idealities reduce the obtainable speedup when parallelizing. These non-idealities were added in the Vopt calculation when parallelizing and the benchmarks were again parallelized to fully compensate for frequency loss. Like leakage and Amdahl overheads, architectural overheads further increase the minimum energy consumption and Vopt as shown in Figure 2.5. Certain benchmarks are highly parallelizable before caches and coherency is introduced, while others have negligible architectural overheads. For example, ocn has almost no Amdahl overheads but significant architectural overheads. To contrast, lun has little architectural but significant Amdahl overheads. Across the benchmarks shown Vopt increases by no more than 200mV. Thus, architectural overheads are another key limiter to voltage scaling, increasing Vopt and the minimum obtainable energy consumption when a task is parallelized to compensate for frequency loss.

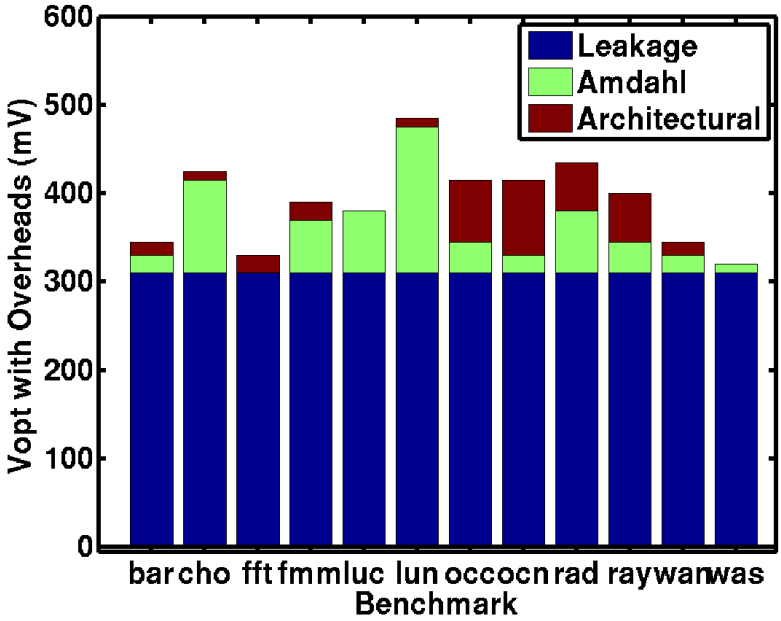


Figure 2.5: **Vopt in 32nm with leakage, Amdahl, and Architectural overheads.** Architectural and Amdahl overheads increase Vopt by no more than 200 mV.

2.3 Impact of Technology and Circuit Features on NTC

The previous section discussed the three key limiters of energy-efficient scaling. However, Vopt is also impacted by additional technology and circuit factors, including technology node, transistor Vt, and process variation, which are discussed below.

2.3.1 Technology

In the previous section Vopt was analyzed at single 32nm technology node. To identify if there is a voltage scaling and parallelization guideline consistent across many technologies, we calculated Vopt for SPLASH-2 across 6 industrial technologies when accounting for all three voltage scaling overheads, as shown in Figure 2.6. Circuit simulations of energy and performance were done in Cadence Spectre using industrial foundry technology kits from 32nm to 180nm.

The process node affects Vopt primarily because technologies have become more leaky generation-to-generation due to reduced threshold voltage. Higher leakage increases Vopt, however, the lower threshold voltage will also improve the frequency degradation with voltage

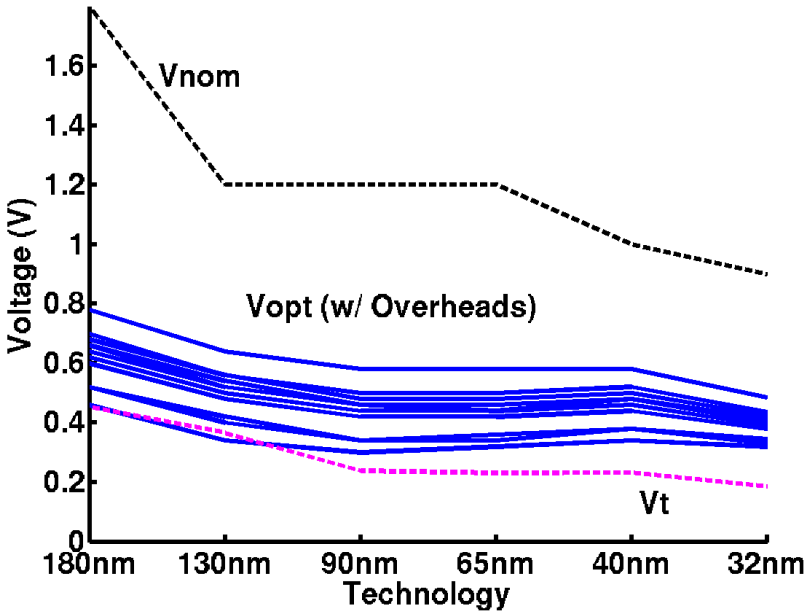


Figure 2.6: **Vopt across technologies when including all three overheads.** Vopt has been trending downward with each generation and is 200-400mV above Vt for most benchmarks.

scaling which will reduce Vopt. A key finding of this work is that Vopt consistently tracks 200-400mV above the threshold voltage for most benchmarks across the six technology nodes. We define this region above the threshold voltage as the near-threshold (NTC) region. Three benchmarks, `Barnes`, `FFT`, and `Water Spatial`, were close to ideally parallelizable and are not contained in the NTC region. However, most general-purpose, high-performance CMP applications will have some degree of parallelization overhead and thus lie in the NTC region.

The median energy gains at Vopt operation and optimal number of cores to parallelize across to compensate for clock frequency loss, Nopt, for SPLASH-2 across technology nodes is shown in Figure 2.7. Table 2.2 includes a breakdown of energy gains and optimal number of cores for each benchmark in the SPLASH-2 suite. Energy gains have diminished by 1.8× from 180nm to 32nm as leakage has increased and the dynamic range available for voltage scaling has narrowed from 180nm to 32nm. This difference is less dramatic with less scalable benchmarks, since the parallelism overheads are higher and thus the amount of voltage scaling in older technologies is limited. The energy gains in newer technology from operating at Vopt instead of at nominal voltage are 4× and Nopt has a median of

12, and no more than 25, cores for SPLASH-2 in 32nm. This increased energy efficiency directly increases CMP performance when limited by a thermal budget. Thus, to maximize thermally-limited CMP performance, tasks should operate in the near-threshold region and parallelize on no more than 25 cores in 32nm. The energy gains and optimal amount of parallelism has decreased with each generation.

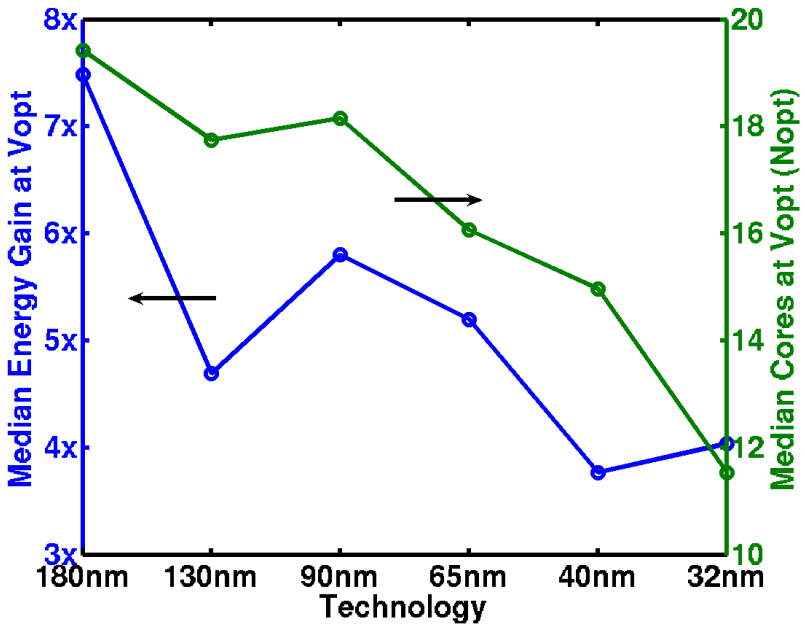


Figure 2.7: **Median energy gains and optimal number of cores, Nopt, when operating at Vopt as compared to nominal voltage for SPLASH-2 benchmarks.** Energy gains and optimal number of cores has trended downward from 180nm to 32nm as leakage has increased and the dynamic range available for voltage scaling has narrowed.

If Amdahl and architectural overheads can be neglected because an application is latency insensitive (for instance, sensor applications) then only the fundamental leakage overhead needs to be considered. To provide a comparison with the trend of Vopt for latency-sensitive applications, we show in Figure 2.8 the fundamental lower bound on Vopt across technologies, where leakage is the only voltage scaling overhead. In 180nm and 130nm Vopt for a perfectly parallelizable task is below threshold. Because technologies are becoming leakier with process scaling, Vopt has been trending upward with each generation and becomes super-threshold in 90nm. For a perfectly parallelizable task the energy gain has decreased from 52× in 180nm to 6× in 32nm, Figure 2.9. Likewise, the optimal number of cores Nopt has decreased from

Table 2.2: Energy gain and optimal number of cores Nopt (in parenthesis) across SPLASH-2 benchmarks and technologies when including the three voltage scaling overheads.

Benchmark	180nm	130nm	90nm	65nm	40nm	32nm
bar	12.7× (71)	7.9× (55)	10.0× (69)	7.8× (43)	5.7× (44)	5.2× (19)
cho	6.1× (13)	3.9× (10)	4.6× (14)	4.3× (12)	3.2× (11)	3.5× (9)
fft	13.2× (68)	8.3× (82)	10.4× (67)	8.0× (42)	5.8× (43)	5.2× (23)
fmm	7.7× (21)	4.8× (20)	6.0× (19)	5.3× (16)	3.9× (16)	4.1× (12)
luc	8.6× (26)	5.3× (25)	6.7× (24)	5.9× (18)	4.2× (21)	4.4× (13)
lun	4.1× (8)	2.7× (6)	3.1× (7)	3.0× (6)	2.3× (6)	2.6× (6)
occ	6.9× (14)	4.3× (16)	5.3× (17)	4.8× (14)	3.5× (13)	3.8× (9)
ocn	6.8× (15)	4.2× (12)	5.2× (17)	4.7× (14)	3.5× (14)	3.8× (9)
rad	5.7× (11)	3.6× (12)	4.3× (12)	4.0× (10)	3.0× (9)	3.4× (8)
ray	7.3× (17)	4.6× (15)	5.6× (16)	5.0× (17)	3.7× (13)	4.0× (11)
wan	12.6× (71)	7.8× (56)	9.9× (70)	7.8× (32)	5.7× (45)	5.2× (19)
was	18× (186)	11.3× (250)	13.0× (121)	9.0× (51)	6.8× (79)	5.5× (25)

21,000 (clearly unachievable) in 180nm to 29 in 32nm. Though parallelizing across thousands of cores in older technologies is not achievable, the gains and Nopt in recent technology nodes have dramatically decreased even when neglecting Amdahl and architectural overheads.

2.3.2 Process Variation

A challenge of operating at a reduced voltage is increased sensitivity to process, temperature, and supply voltage variations that causes variability in circuit delay and energy consumption. A slower critical path and leakier devices decrease energy-efficiency thus increasing Vopt. Figure 2.10 shows the 3-sigma delay variation relative to mean for a single gate and a chain of logic in 40nm technology using industrial variation models. Process variation can be global, affecting all transistors uniformly across a die, or local which causes delay mismatch between different devices and paths on a chip.

In the NTC region, local variation accounts for 30% of 3-sigma delay of a single gate. Since a CMP's maximum clock frequency is limited by the worst-case critical path, mismatch between different critical paths raises Vopt as the leakiest path runs at clock frequency set by the slowest path. However, local variation is reduced for deeper logic depths since local variation is usually uncorrelated and hence averages out along a path. Thus, for a chain 31 gates, local variation is only 10% of total variation, so its impact is minor.

Global variation raises Vopt, Figure 2.11 for three technology nodes, but all paths will either: (1) slow and have less leakage or (2) have more leakage but run fast, so the total

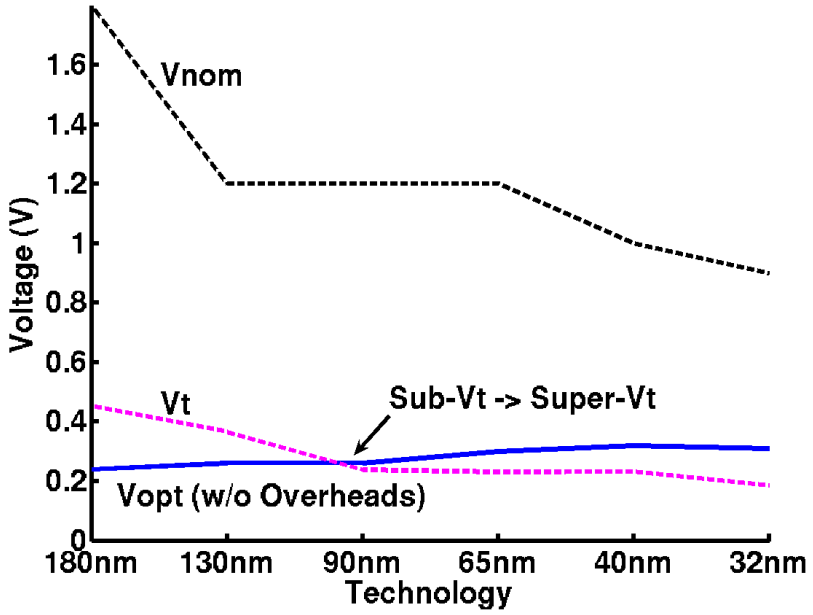


Figure 2.8: **Theoretical lowest Vopt across six technology nodes with leakage overheads only.** Vopt has been trending upward with each generation and becomes super-threshold in 90nm.

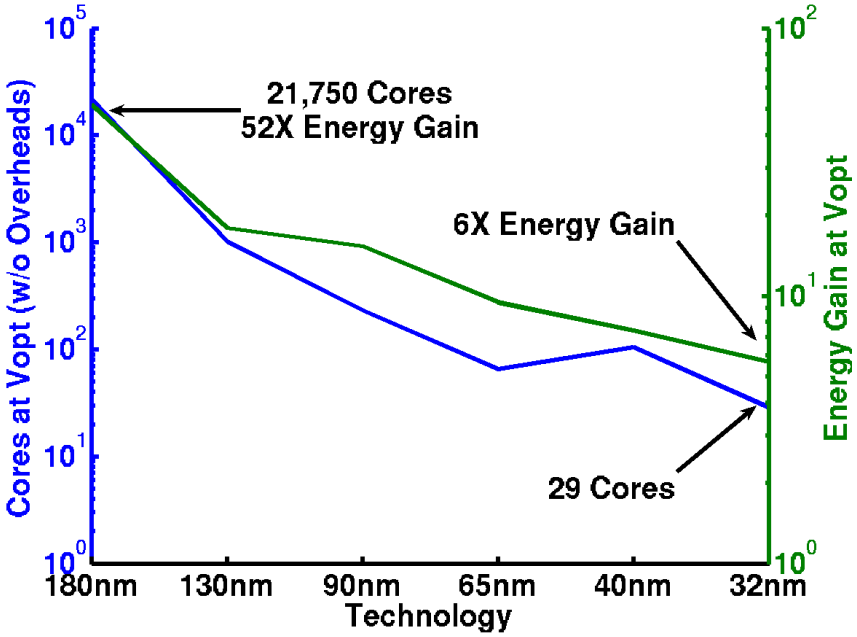


Figure 2.9: **Theoretical maximum energy-efficient parallelism Nopt and energy gains across six technology nodes with leakage overheads only.** Energy gain has reduced from 52× in 180nm to 6× in 32nm.

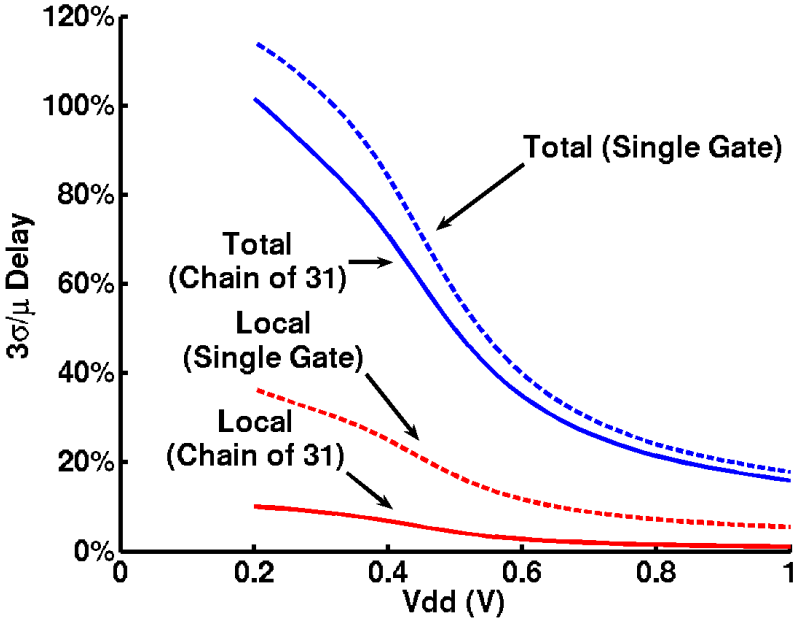


Figure 2.10: **Change of delay for total and local process variation of a single gate and a logic chain of 31 gates.** Longer gate chains average out mismatch.

leakage overhead is relatively constant. This is unlike local variation where the leakiest path is run at the slowest clock frequency, thus global variation's contribution to increasing Vopt is less than local variation. Total delay variation at Vopt is significant, but high-performance CMPs are usually binned for speed so that each die can run at its optimal frequency. Thus, the range of bins will increase but, since each die is tuned to its optimum speed, global variation does not significantly increase Vopt.

The increase in Vopt when considering 3-sigma delay variation and the parallelization overheads described above is 30mV-60mV for an average SPLASH-2 benchmark. The delay variation also depends on the number of critical paths in a design, since local variation reduces by taking a maximum across multiple paths. As the number of paths increases the mean shifts up, but the variation is reduced, shown in Figure 2.11. Thus, variation does impact delay but its impact on Vopt and minimum energy are small.

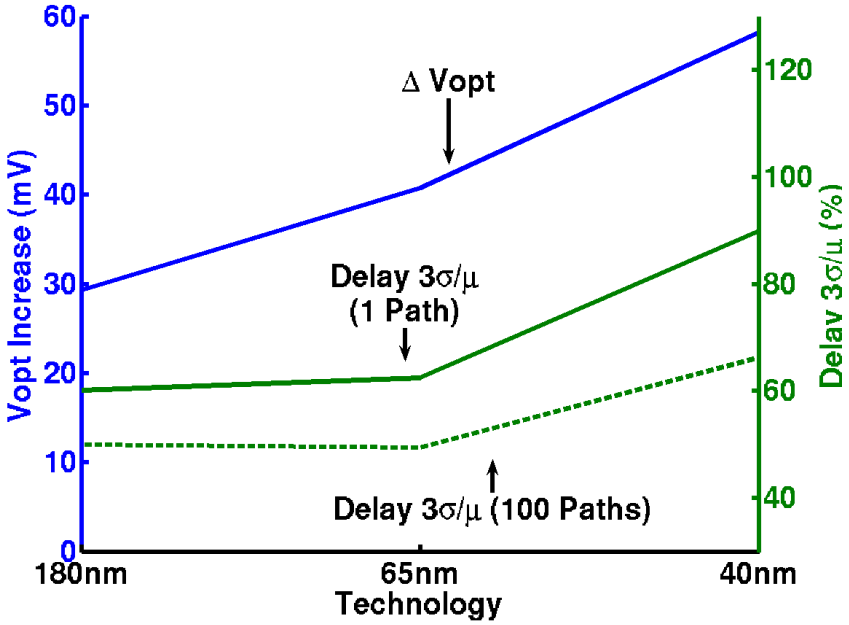


Figure 2.11: **Increase in Vopt because of 3-sigma variation across generations.** Variation increases Vopt by 30mV-60mV on average for SPLASH-2 benchmarks.

2.3.3 Transistor Threshold Voltage

The energy-efficient operating voltage Vopt also depends on transistor threshold voltage selection. Conventionally regular threshold voltage transistors are used for high-performance applications, since they have the best drive strength, whereas the higher threshold voltage transistors are used where low static-power is a concern, such as in mobile applications.

When considering a parallelized task, Amdahl and architectural overheads limit the energy-efficiency and voltage scalability, thus setting Vopt, Figure 2.12 (top). As threshold voltage is reduced, leakage begins to dominate until the voltage scalability is limited by leakage overheads and not Amdahl or architectural overheads. The energy drops initially as threshold is reduced, since the task can run faster, and Vopt correspondingly tracks. Once leakage dominates the energy stays relatively constant. As a rule-of-thumb, the optimal threshold voltage is at the inflection point (approximately 250mV in figure) between the parallelism-dominant and leakage-dominant region, since above this point energy increases and below this point the process becomes unnecessarily leaky.

For comparison, Figure 2.12 (bottom) shows Vopt when a task only includes leakage overheads. Since there is no Amdahl or architecture overheads, Vopt lowers as threshold voltage increases, since the integrated leakage current is reduced, until Vopt enters the subthreshold regime. Once Vopt is subthreshold, raising the threshold voltage does not change the energy-efficient operating voltage or energy consumption to first-order [37].

Tasks that are not latency sensitive can operate in subthreshold with high Vt transistors, but this is not optimal for latency-sensitive applications. To achieve maximum performance in latency sensitive applications, even when limited by a thermal budget, the threshold voltage should be reduced until leakage starts to dominate voltage scalability.

2.4 Related Work: Energy-Delay Product

The energy gains by running at a lower voltage come at the cost of decreased clock frequency. A common optimization goal in this trade-off has been minimizing the energy-delay-product (EDP) [43]. Figure 2.13 shows the marginal percentage gain in energy by running slightly slower as voltage is scaled. Since clock frequency initially increases linearly as the voltage is scaled, while dynamic energy decreases quadratically, the marginal tradeoff between is energy and delay is above unity, e.g., a 5% decrease in frequency results in more than 5% energy improvement.

When marginal decrease in frequency is equal to the increase in energy, the energy-delay product (EDP) is minimized, labeled as Vedp in Figure 2.13. Many designs are targeted to minimize EDP since it provides a reasonable tradeoff between energy and delay of a circuit. Alternatively, metrics that weigh delay more heavily have been proposed, such as ED2P [44]. However, all of these metrics, including EDP, tradeoff energy for delay. In a thermally-constrained system, maximizing energy-efficiency is essential to maximize performance. Thus, instead of trading off energy and delay or minimizing EDP when targeting a design, energy itself should be minimized in these systems which occurs at Vopt.

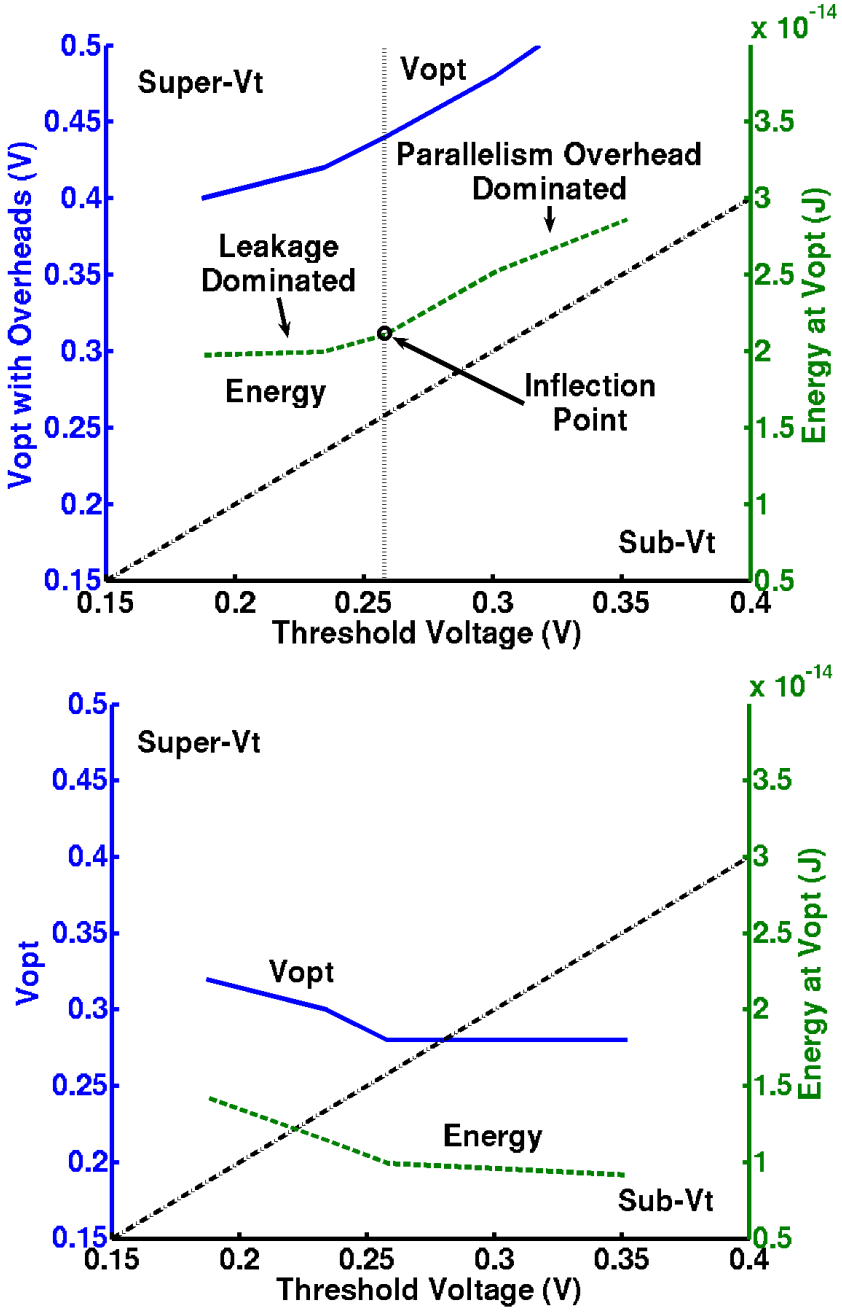


Figure 2.12: **Vopt vs. Vt in 32nm with Amdahl and architecture overheads (top) and leakage only (bottom).** With ideal workloads, higher threshold voltage increases energy efficiency until Vopt is below Vt. With non-ideal workloads, a lower Vt improves energy efficiency.

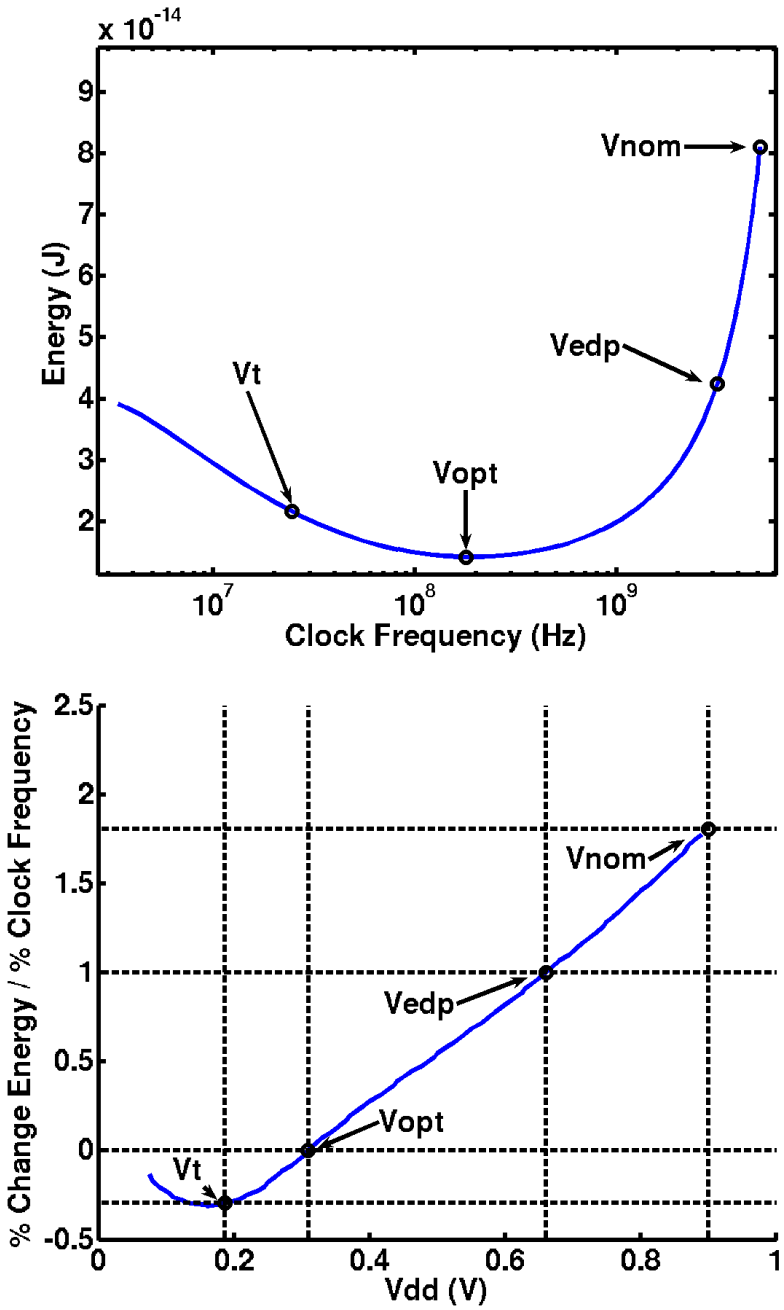


Figure 2.13: **Energy and performance tradeoff in 32nm and marginal cost.** Tradeoff between performance and energy (top) in 32nm as Vdd is swept showing the energy minimum. Energy decreases towards Vopt / Eopt before static energy dominates. Marginal gains in energy for marginal decreases in frequency (bottom) show that energy gains diminish as voltage is scaled. The energy-delay optimal point is when the marginal gain is 1 while the energy-minimal Vopt is at 0.

2.5 Conclusions

We have detailed the limits of voltage scaling for latency-sensitive applications, when slower clock frequency is compensated by parallelization across multiple cores. As CMPs become limited by thermal cooling constraints, near-threshold operation is needed to maximize computations for a fixed thermal design power. The three voltage scaling limiters, leakage, Amdahl, and architectural, contribute to increasing the minimum energy and optimal supply voltage Vopt to maximize total CMP performance. As a guideline, the near-threshold region for maximum energy-efficiency is roughly 200mV-400mV above threshold voltage for most applications and this trend held for the six technology nodes we examined.

NTC operation increases energy-efficiency of a core by approximately 4× in 32nm for the SPLASH-2 benchmarks we investigated, roughly translating to a 4× improvement in performance for a thermally-limited CMP. Additionally, the maximum amount of energy-efficient parallelism is no more than 25 cores in 32nm. Delay variation increases in the NTC region, but has little impact on Vopt. For latency sensitive applications threshold voltage should be minimized until leakage dominates voltage scalability, whereas latency insensitive applications benefit from subthreshold operation.

CHAPTER 3

NTC with Voltage Boosting

3.1 Motivation

The previous chapter defined near-threshold and the three key overheads that limit voltage scaling. However, we showed that parallelism overheads such as Amdahl and architectural, can severely limit energy efficiency because serial portions of code cannot be split among cores and eventually bottleneck performance. Assuming that a workload can be split into parallelizable and serial components, energy efficiency could be improved by dynamically adjusting a core's voltage between near-threshold and nominal for parallelizable and serial components, respectively. Serial components benefit the most from as much single-threaded performance as possible, as there is no way to decrease their latency besides raw processor performance. Thus, the optimal operating voltage for serial code, assuming latency cannot degrade, is at nominal voltage. Any reduction in operating voltage for a serial task will increase latency, which cannot be recovered through parallelization by definition.

In this chapter we expand on the results from Chapter 2 by including fast boosting effects in the near-threshold analysis, and assume that a workload can be subdivided into serial and parallelizable components. By raising a core's operating voltage quickly from near-threshold to nominal, in a handful of cycles, we should further energy improvement for the benchmarks tested. Additionally, the near-threshold voltage Vopt narrows as energy becomes less sensitive to the serial component of a workload.

3.2 Reexamining the Scaling Limiters

We will reexamine the key factors that limit energy-efficient scaling when a task is parallelized to maintain constant latency. The three scaling limiters are leakage, Amdahl overheads, and architectural overheads. Each of these limiters contributes to increased minimum energy and raises the energy-efficient operating point, Vopt. However, only Amdahl and architectural are parallelism overheads, where leakage is not impacted by fast boosting. In a system with boosting, we will assume Amdahl overheads can be eliminated and Vopt lowered as Amdahl represents the 'algorithmic' overhead of a task. For the purposes of this study we assume this algorithmic overhead can be subdivided perfectly into parallelizable and serial components.

3.2.1 Amdahl

As in Chapter 2, we fitted the SPLASH-2 benchmark speedups to Amdahl's law and applied the law to the voltage-scaling calculations to obtain Vopt when considering non-ideal parallelization. The benchmarks were parallelized to fully compensate for frequency loss from lower-voltage operation. Figure 3.1 shows Vopt increasing in 32nm because of Amdahl overheads. Some SPLASH-2 benchmarks, such as `Barnes`, have nearly ideal speedup, indicating very little parallelization penalty from Amdahl overheads. Other benchmarks, such as `LUNC`, reach a speedup of only 10 with 64 cores, indicating a high percentage of serial code. These benchmarks represent a range of parallelized scientific workloads applicable to CMPs. When Amdahl overheads are added, the Vopt operating range for most overheads is 25 mV to 150 mV above the leakage-overheads-only case. Although the serial coefficient is highly application-dependent, the range of Vopt for the benchmarks is small, varying by only $\sim 150\,\text{mV}$. If the serial coefficient were 100 percent (that is, if none of the code were parallelizable), then nominal voltage would be optimal.

If the serial code portion could be efficiently detected in a parallelized algorithm, then the system could operate between two simultaneous voltage modes, depending on whether an algorithm's serial portion or a parallel portion were running. In a system equipped with voltage boosting [45–47], where a single core's operating voltage can be rapidly increased for

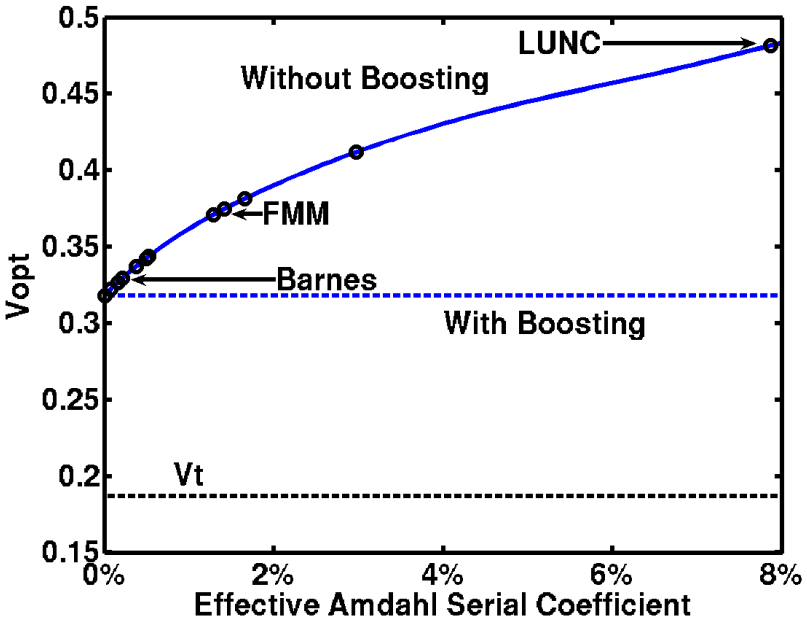


Figure 3.1: **Comparison of Vopt vs. Amdahl serial coefficient with and without boosting.** Vopt vs. Amdahl coefficient for all SPLASH-2 benchmarks (three labeled) in 32nm. If parallel and serial portions of code are separated, the parallel portions operate at the lowest possible Vopt. Meanwhile, the serial portion will run at its optimal voltage of max Vdd.

high-performance singlethreaded operation, then the serial portion can be quickly overcome, regaining parallelism speedup and extending the number of cores for which an algorithm can be further parallelized. Recent work has shown that cores can be boosted within approximately 9 ns, or roughly one cycle, when operating at low-voltage clock frequencies [47]. We modeled this case, by assuming that the serial code portion can be perfectly separated from the parallel portion and that it is run on a dedicated boosted core operating at full voltage. Because frequency loss from voltage scaling of a serial portion cannot, by definition, be compensated by parallelizing, the serial code runs most efficiently at full voltage. Similarly, the parallel portion now runs most efficiently at the leakage-only Vopt, because all serial portions have been removed. Thus, Vopt's dependence on the Amdahl serial coefficient is flat in Figure 3.1. Of course, serial code cannot be perfectly separated from parallel code, and in a real system nonidealities would increase Vopt somewhere between the two "without boosting" and "with boosting" extremes.

In 32 nm, energy gain by operating at Vopt decreases from nearly 6× to 1× as the Amdahl serial coefficient increases from 0 percent to 100 percent (see Figure 3.2). Without boosting, energy gains decrease dramatically as the Amdahl coefficient is increased and, with a 15 percent coefficient, energy gain is only 2×, or one-third of the ideal gain. Additional energy can be recovered by introducing boosting, where at a 15 percent Amdahl coefficient, the energy gain is 3.3×. At an Amdahl coefficient of 50 percent, no energy can be gained by parallelizing and operating at Vopt without boosting, while boosting can recover 63 percent more energy.

3.2.2 Architectural

Unlike with Amdahl overheads, we did not assume architectural overheads could be perfectly separated into serial and parallel phases for the purposes of boosting. As a comparison, Figure 3.3 includes the additional impact of Amdahl overheads on Vopt for architectures without boosting. Certain benchmarks are highly parallelizable before caches and coherency are introduced, while others have negligible architectural overheads. For example, the `OCN` benchmark has almost no Amdahl overheads but significant architectural overheads. In con-

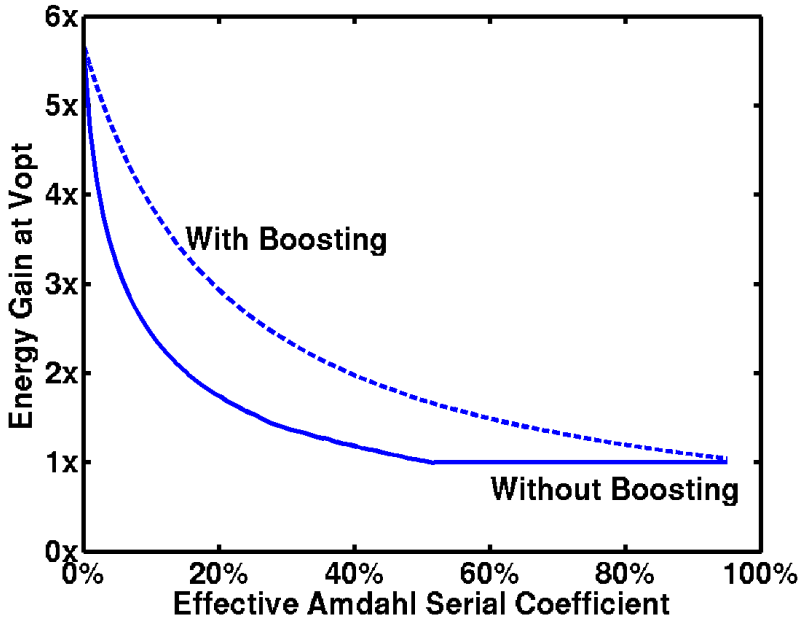


Figure 3.2: **Comparison of NTC energy gain vs. Amdahl serial coefficient with and without boosting.** Boosting can increase energy efficiency, especially with a moderate Amdahl coefficient. Without boosting, no energy gain can be achieved above 50% serial coefficient.

trast, the `LUN` benchmark has little architectural but significant Amdahl overheads. Across the benchmarks shown, Vopt increases by no more than 100 mV with boosting as compared to 200 mV without boosting. Thus, boosting lessens the impact of parallelism overheads.

3.3 Reexamining the Impact of Technology on NTC

As with the above, we now extend the across technology analysis from Chapter 2 to include the effects of boosting. In the previous section, Vopt was analyzed at single 32-nm technology node. To identify whether a voltage-scaling and parallelization guideline is consistent across many technologies, we calculated Vopt for SPLASH-2 across six industrial technologies when accounting for all three voltage-scaling overheads, as shown in figure 3.4 and 3.5. Circuit simulations of energy and performance were done in Cadence Spectre using industrial foundry technology kits from 32 nm to 180 nm.

The process node affects Vopt primarily because technologies have become more leaky with each generation due to reduced threshold voltage. Higher leakage increases Vopt; how-

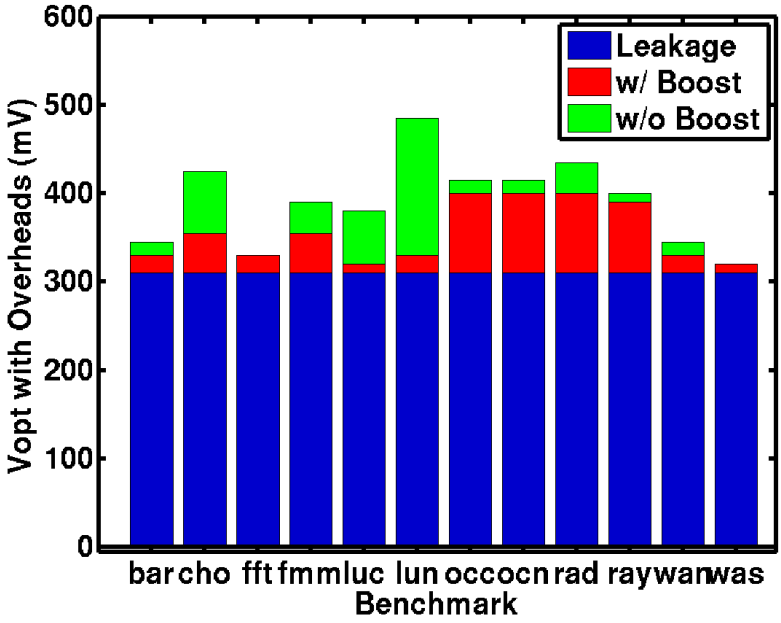


Figure 3.3: **Vopt across SPLASH-2 benchmarks with and without boosting.** Architectural and Amdahl overheads increase Vopt by no more than 100 mV with boosting and 200 mV without boosting. However, this varies by benchmark.

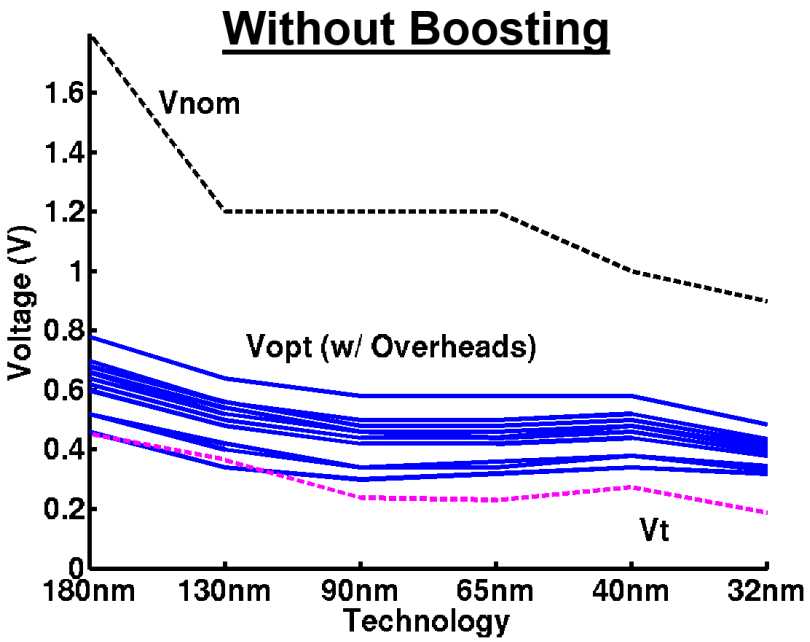


Figure 3.4: **Vopt across technologies when including all three overheads without boosting.**

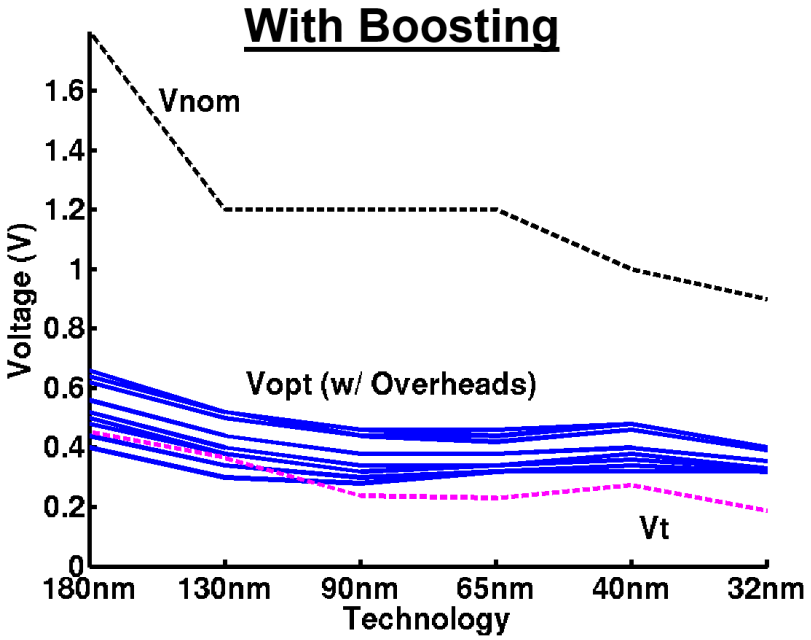


Figure 3.5: **Vopt across technologies when including all three overheads with boosting.** Boosting tightens the range of Vopt as Amdahl overheads are removed.

ever, the lower threshold voltage will also improve the frequency degradation with voltage scaling, which will reduce Vopt. A key finding of this work is that, for most benchmarks across the six technology nodes, Vopt consistently tracks 200 mV to 400 mV above the threshold voltage without boosting and 100 mV to 200 mV with boosting. We define this region above the threshold voltage as the near-threshold computing (NTC) region. Three benchmarks- `Barnes`, `FFT`, and `Water Spatial`- were close to ideally parallelizable and are not contained in the NTC region. However, most general-purpose, high-performance CMP applications will have some degree of parallelization overhead and thus lie in the NTC region.

Figure 3.6 shows the median energy gains at Vopt operation, with and without boosting, and the optimal number of cores to parallelize across to compensate for clock frequency loss, Nopt, for Splash-2 across technology nodes. Energy gains have diminished by $\sim 1.8\times$ from 180 nm to 32 nm as leakage has increased and the dynamic range available for voltage scaling has narrowed from 180 nm to 32 nm. This difference is less dramatic with less-scalable benchmarks, because the parallelism overheads are higher and thus the amount of voltage scaling in older technologies is limited.

In newer technology, the energy gains from operating at Vopt without boosting instead of at nominal voltage are $\sim 4\times$, and Nopt has a median of ~ 12, and no more than 25, cores for SPLASH-2 in 32 nm without boosting. With boosting, the energy gain in 32 nm is $4.5\times$, while the median number of cores is ~ 20 and no more than 25 cores. However, gains by boosting are expected to improve for benchmarks that are less parallelizable, with an Amdahl coefficient above 10 percent, but with negligible architectural overheads. This increased energy efficiency directly increases CMP performance when limited by a thermal budget. Thus, to maximize thermally limited CMP performance, tasks should operate in the NTC region and parallelize on no more than 25 cores with and without boosting. The energy gains and optimal amount of parallelism has decreased with each generation.

3.4 Conclusions

We have expanded the limits of voltage scaling for latency-sensitive applications, when slower clock frequency is compensated by parallelization across multiple cores, by adding the effects of fast voltage boosting. As CMPs become limited by thermal-cooling constraints, near-threshold operation is needed to maximize the computations for a fixed thermal design power. However, many obstacles remain before near-threshold designs can be fully realized in commercial systems. Process variation and supply-noise sensitivity can be very high in the near-threshold region. Increased clock skew and hold-time uncertainty further inflate timing margins, limiting clock frequency and achievable energy gains. These effects, however, can be mitigated through soft clocking and in-situ error detection techniques [6]. Additionally, a fundamental challenge of boosting a core between near-threshold and super-threshold supply voltages is circuit scalability. For example, fewer repeaters are required for an on-chip interconnect in near-threshold than in super-threshold design, because wire delay becomes relatively faster compared to circuit delay as VDD is reduced. Thus, design optimizations to improve performance in near-threshold may negatively affect super-threshold performance. Developing techniques to minimize super-threshold impact is critical for realizing a high-performance near-threshold system.

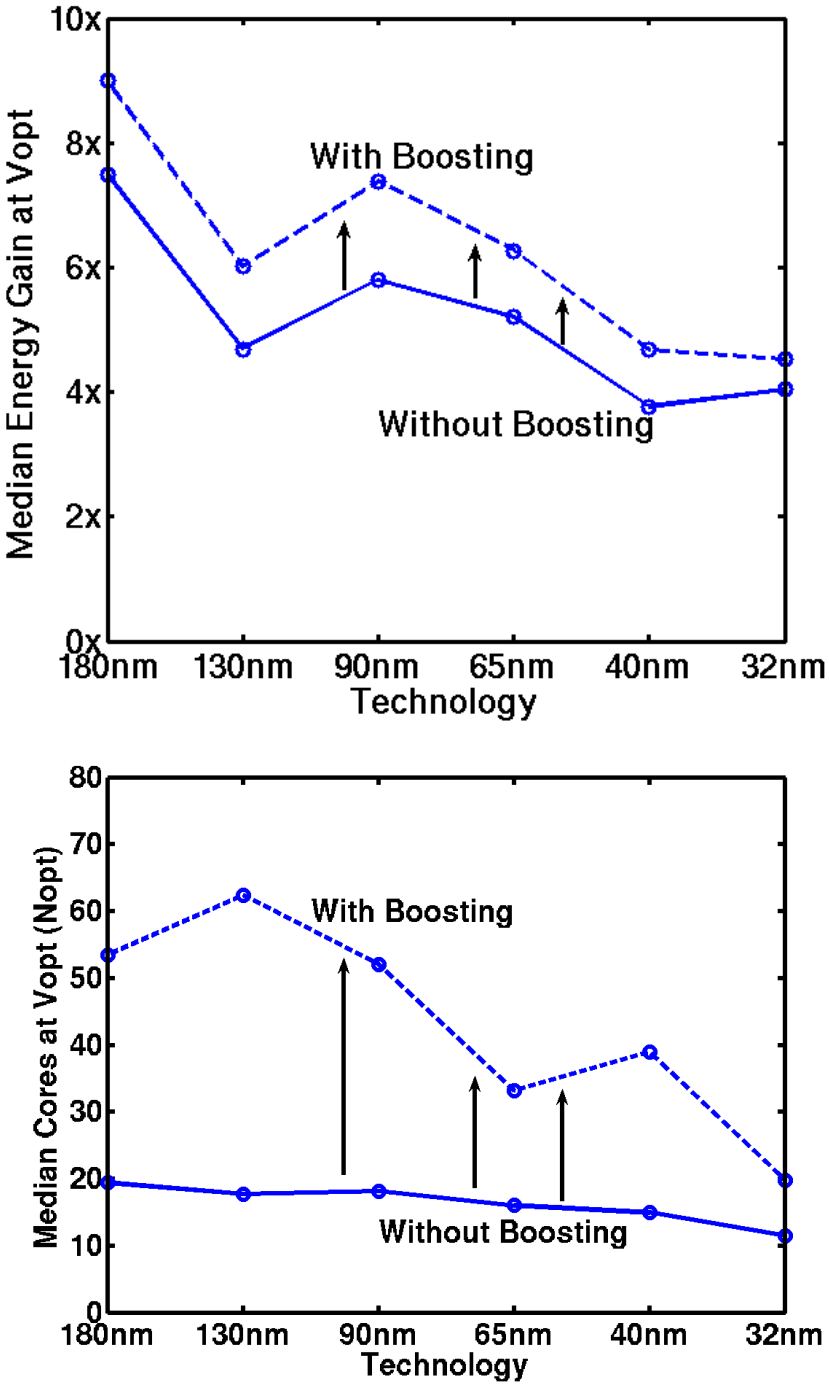


Figure 3.6: **Median energy gains and optimal number of cores, Nopt, when operating at Vopt as compared to nominal voltage for SPLASH-2 benchmarks.** Boosting regains $\sim 0.5\times$ additional energy on average for SPLASH-2 benchmarks.

CHAPTER 4

Fast Boosting: Reclaiming Idle Cycles

4.1 Motivation

After decades of exponential processor performance improvements, Moore's law has now stagnated and power limits the maximum utilization of a CMP [5]. High-performance systems have been proposed [10, 12, 48–51] that use NTC and parallelism to save energy and increase overall performance for parallelizable applications. To optimize energy efficiency core voltages must be adjusted depending on workload. Traditionally this is implemented with a digital voltage and frequency scaling scheme (DVFS) controlled by software. A major drawback of DVFS is long switching times limited by off-chip regulator delay.

Recently fast boosting DVFS techniques have been proposed that reduce core voltage switching times from 100,000s of cycles to 10s of cycles [52, 53]. An alternative approach to increase energy efficiency without voltage scaling is with heterogeneous architectures [54–56] where a big core is optimized for performance while a little core is optimized for energy-efficiency. ARM recently proposed a heterogeneous architecture using an ARM Cortex-A15 and an ARM Cortex-A7. However, migrating threads to cores is controlled by software and requires roughly 20,000 cycles to complete.

Given improvements in fast boosting DVFS, we explore implications on single-threaded applications, such as those common a mobile phones or desktop computers. In particular, we characterize the granularity of idle CPU cycles, such as those caused by a cache miss, for single-thread benchmarks and quantify energy gains for ideal conditions. Previous studies

[52] have focused on parallel applications, such as FFT, in evaluating energy reduction when using fast boosting. Server processors already include simultaneous multithreading to quickly switch when a thread stalls, but this is not common on mobile platforms. Thus, we propose using fast boosting to save power during a cache miss and compare this technique to traditional DVFS and heterogeneous architectures.

4.1.1 Voltage Scaling

Dynamic voltage and frequency scaling (DVFS) has traditionally been used to reduce core power consumption during long periods of low workload, when a core is mostly idle. Reducing frequency quadratically reduces energy consumption near nominal voltage. Figure 4.1 shows an energy efficiency versus performance curve for core logic in a typical 65nm process. Performance and energy are normalized to nominal voltage, and clock frequency is used as a proxy for performance.

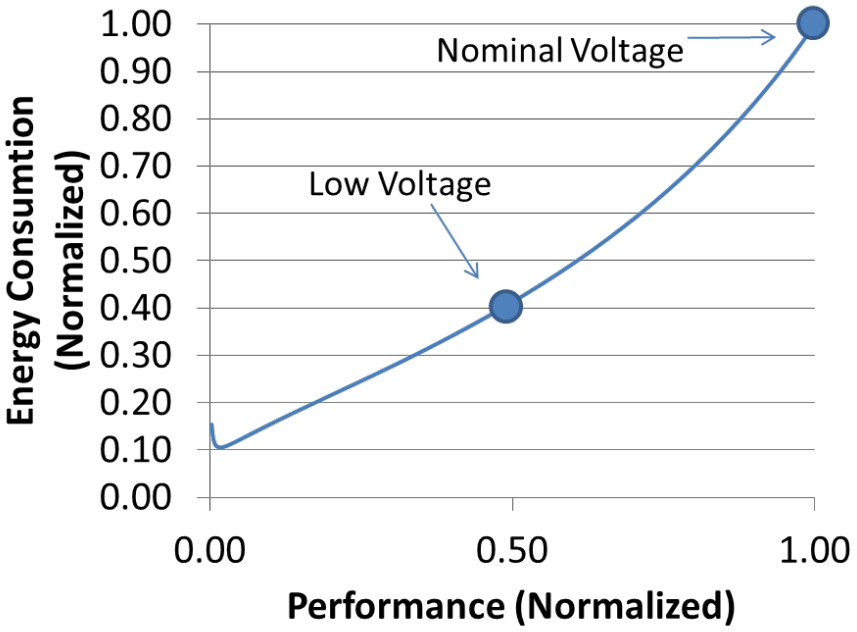


Figure 4.1: **Traditional DVFS increases energy efficiency but requires hundreds of thousands of cycles to switch.** Show here from an industrial 65nm technology, energy consumption is reduced by 60% when operating at a low voltage.

As can be seen from Figure 4.1 , a 50% reduction in performance yields roughly a 60% reduction in energy consumption for a given task. Energy reduction is not perfectly quadratic

due to overheads, such as leakage current. However, traditional DVFS switching speed is limited to approximately 100,000 cycles (1 millisecond) because it is controlled by software and actuated by an off-chip voltage regulator, creating long delay paths to switch. Thus, with traditional DVFS, voltage is scaled at a very coarse granularity.

4.1.2 Heterogenous Architectures

An alternative approach to reduce energy consumption is with heterogeneous cores, where heavy workloads are run on a big core and light workloads are run a little core. ARM has proposed [55] a big/little architecture using a Cortex-A15 as the big core a Cortex-A7 as the little core. From nominal to 50% performance reduction they find a 70% reduction in energy consumption, as shown in Figure 4.2. The ARM Big.LITTLE architecture requires 20,000 cycles to migrate data between cores. Migration time is an order of magnitude faster than traditional DVFS switching speed, but still requires a considerable amount of CPU cycles. This technique is not orthogonal to DVFS, and both techniques may be implemented to allow greater accuracy in picking an optimal operating condition based on workload.

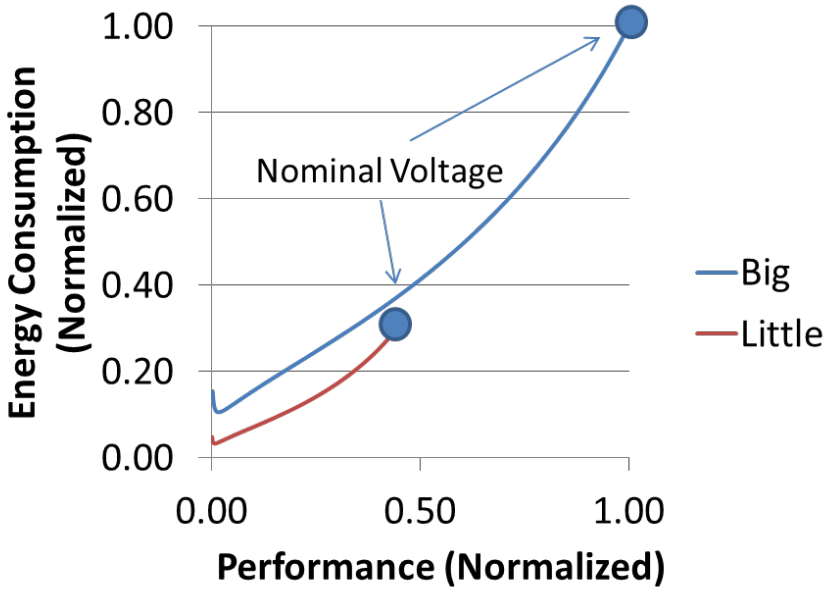


Figure 4.2: **Big.LITTLE further reduces energy consumption at 50% performance.** Since Big.LITTLE includes architectural improvements, energy consumption improvements exceed straight voltage scaling for a fixed workload.

4.2 Methodology

Both energy-reduction techniques, traditional DVFS and Big.LITTLE, require many cycles of power supply switching or core data migration and, thus, are implemented at a coarse granularity. To understand the implications of fast boosting DVFS techniques, which can switch in 10s of cycles, we characterized SPEC2000 and Phoenix 2 benchmarks in GEM5 using a system emulated 64-bit Alpha architecture. Simulation parameters and benchmarks are detailed below.

4.2.1 GEM5 Simulation

The GEM5 simulator [42] was used to simulate performance of a single-threaded in-order and out-of-order 64-bit Alpha architecture. The Alpha architecture was simulated in system emulation mode, so that operating system-dependent overheads were not included. Table 4.1 summarizes the processor, cache, and memory architectural parameters used in the simulation.

Table 4.1: Architectural Parameters for GEM5.

Parameter	Value
Architecture	Alpha 64-bit @ 2 GHz
L1 D$	64 kB, 2-way set associative, 64b cache line
L1 I$	32 kB, 2-way set associative, 64b cache line
Memory	512 MB, 30ns latency
Issues/cycle	1 for in-order, 8 for out-of-order

4.2.2 Benchmarks

Four SPEC2000 benchmarks and one Phoenix 2 benchmark was used to compare architectures. Table 4.2 lists and describes benchmarks simulated. The benchmarks were chosen to represent a range of applications, from science (matrix multiplication) to highly single-threaded pointer chasing (GCC), while maintaining tractable simulation time. SPEC2000 benchmarks were pre-compiled for Alpha architecture, while the matrix multiplication was isolated from Phoenix 2 and cross-compiled for Alpha.

Table 4.2: GEM5 Benchmarks Simulated.

Benchmark	Description
GCC	C Compiler
Matrix	Multiplication Matrix Multiplication
MCF	Combinatorial Optimization
TWOLF	Place and Route Simulator
Vortex	Object-oriented Database

4.3 Results

The benchmarks listed above were simulated in GEM5 and percentages of idle CPU cycles relative to total number of CPU cycles were recorded. For this analysis, an idle CPU occurs when the CPU completes no actions, including: no issues, no commits, no scheduling, and no execution. This is a very conservative definition of idle and future work could explore a relaxed definition of consistency. Idle cycles generally occur during outstanding cache misses and more advanced prediction, prefetching and striding could alleviate this.

Each simulation was split into segments of a fixed number of cycles of 100, 1K, 10K, and a single segment over the entire run. For each segment of cycles, called a window, the percentage of idle cycles (idleness) was computed and recorded. Both in-order and out-of-order architectures were simulated in GEM5 and the results are shown below.

4.3.1 Out-of-Order Core

Figure 4.3 shows a cumulative histogram of idleness for varying window sizes of GCC running on an out-of-order core. Windows are binned according to minimum amount of idleness. For example, the "Idle 0%-10%" blue-colored bin for a window size of 100 cycles is 100%. That indicates that all 100-cycle windows have at least 0% idleness. This is the expected trivial case, since all windows have at least no idleness. As the bin threshold increases to "Idle 10%-20%" (red bar) the percentage of windows with at least 10% idleness is 60%. At the high 70% threshold (pink bar) the percentage of windows with at least 70% idleness is reduced to 5%. For the purposes of voltage scaling or migrating to a slower core, a higher percentage of windows with high percentage threshold are desirable.

For the two techniques we explored, DVFS and core migration, we chose an idleness threshold of 50%. This corresponds to an energy-efficiency increase of 2.5× for voltage scaling and 3.2× for Big.LITTLE, as mentioned above.

The GCC benchmark for out-of-order, shown in Figure 4.3, has a low idleness threshold averaged across the entire benchmark run ("Max" shown in the figure). The average idleness is in the 30%-40%, which is below our chosen 50% idleness threshold. This indicates potential for energy savings through the reduction techniques, but the code must be segmented into windows to extract periods of high idleness. As the window size decreases to 10,000 cycles and 1,000 cycles, 10% and 28% of the windows have an idleness of at least 50%. At a window size of 100 cycles the percentage increases to 55%, indicating a fine granularity with periods of idleness. Ideally during these high-idleness periods the core could be voltage scaled, or the state could be migrated to a smaller core, to improve energy efficiency that could not otherwise be extracted at a coarse, task-level granularity.

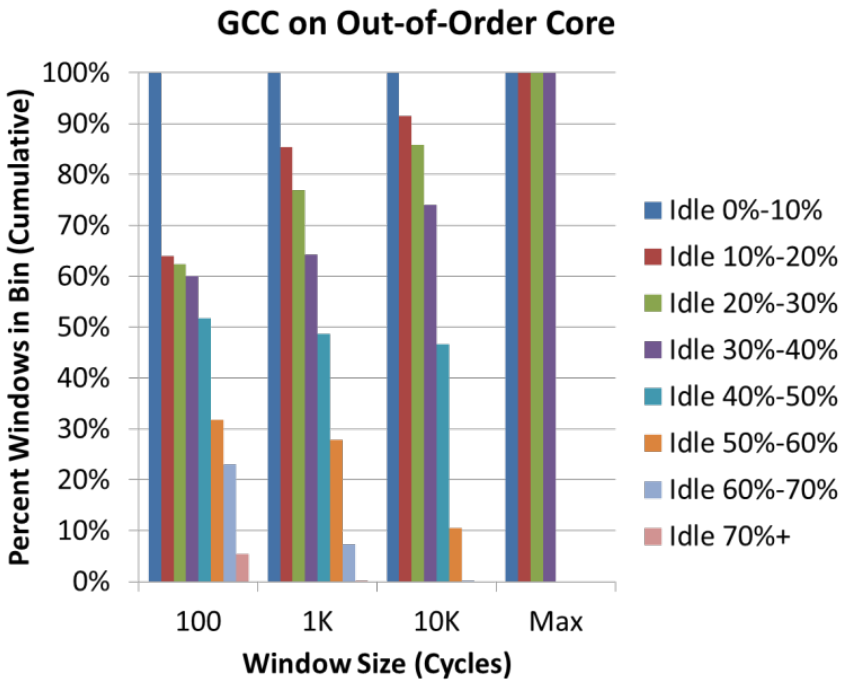


Figure 4.3: **Percent windows are idle for varying window sizes. GCC benchmark on out-of-order core.** Approximately 30% of windows are idle more than not (idle cycles 50%+) with window size of 100.

Percent of windows with high idleness for Vortex on an out-of-order core is shown in Figure 4.4. Again, at a coarse granularity the idleness is small (¡ 5% of windows have high idleness), but as window size is reduced the idleness increases to 25% at a window size of 100.

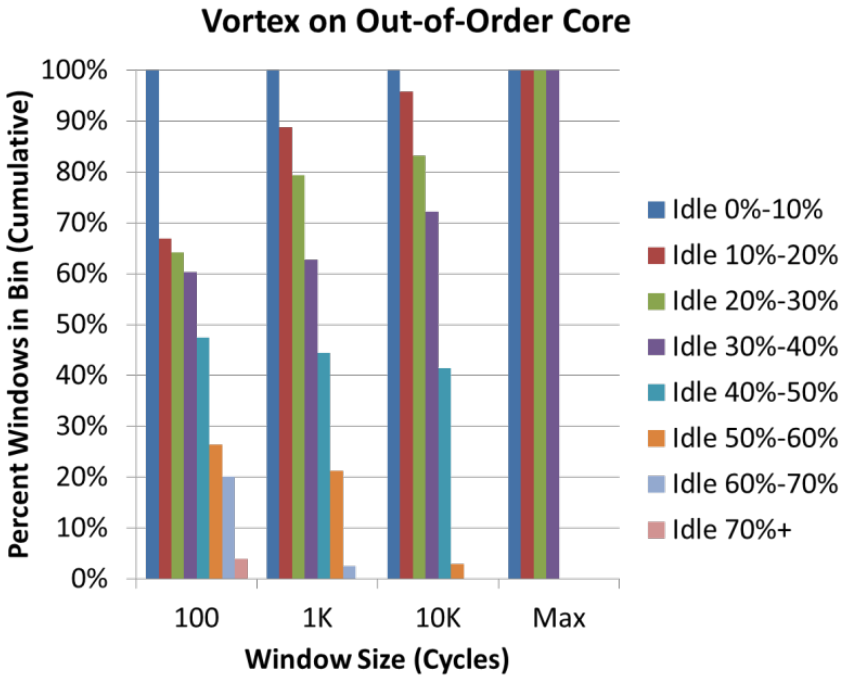


Figure 4.4: **Percent windows are idle for varying window sizes. Vortex benchmark on out-of-order core.** Approximately 25% of windows are idle more than not (idle cycles 50%+) with 100-cycle windows.

The matrix multiplication benchmark is shown in Figure 4.5. Unlike GCC and Vortex, this benchmark is highly regular and cache misses are rare. In this case, the idleness remains exceedingly low with less than 5% of windows idle across all window sizes explored. This is expected as the processor rarely stalls for this type of application. Thus, extracting idleness is useful for general-purpose but not scientific applications on an out-of-order core.

4.3.2 In-Order Core

The SPEC2000 benchmarks were also simulated for an Alpha in-order core architecture. A plot of idleness for GCC on in-order is shown in Figure 4.6. Across the entire program the average idleness increased from out-of-order to between 40% and 50%. As window size

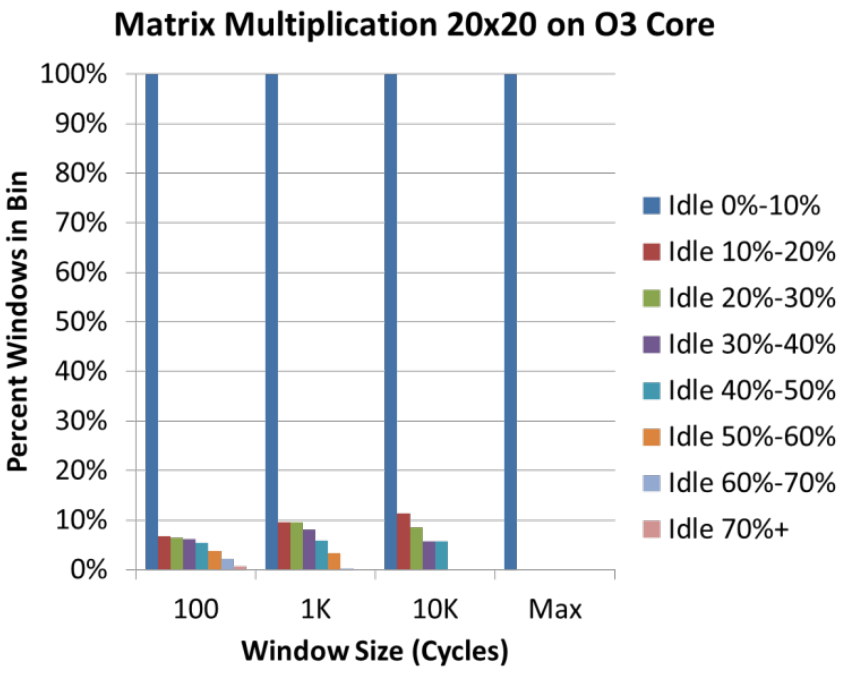


Figure 4.5: **Percent windows are idle for varying window sizes. Matrix multiplication benchmark on out-of-order core.** Approximately 3% of windows are idle more than not (idle cycles 50%+) with 100-cycle windows.

is reduced to 10,000 cycles the percentage of windows with high idleness increases to over 50%, much quicker than the out-of-order case. This is to be expected as every cache miss stalls the core. Similar results for Vortex on an in-order core are shown in Figure 4.7.

A summary of percentage windows containing high idleness (idle cycles $> 50\%$) within window across the SPEC2000 benchmarks simulated is given in Table 4.3. GCC and Vortex contain the highest amount of idleness for small window sizes, while MCF (combinatorial optimization) and TWOLF (place and route simulator) do not exceed 8% with a window size of 100 cycles for out-of-order. For an in-order core, all benchmarks except TWOLF experience a high idleness at small windows sizes, with MCF possessing higher idleness than GCC and Vortex at 88% vs. 55%. On average across the four benchmarks, a window size of 100 cycles had 49% of idle windows for in-order and 18% for out-of-order. In-order plateaus at 10,000 cycles (or potentially even at larger window sizes, though this was not simulated) and out-of-order at 1,000 cycles.

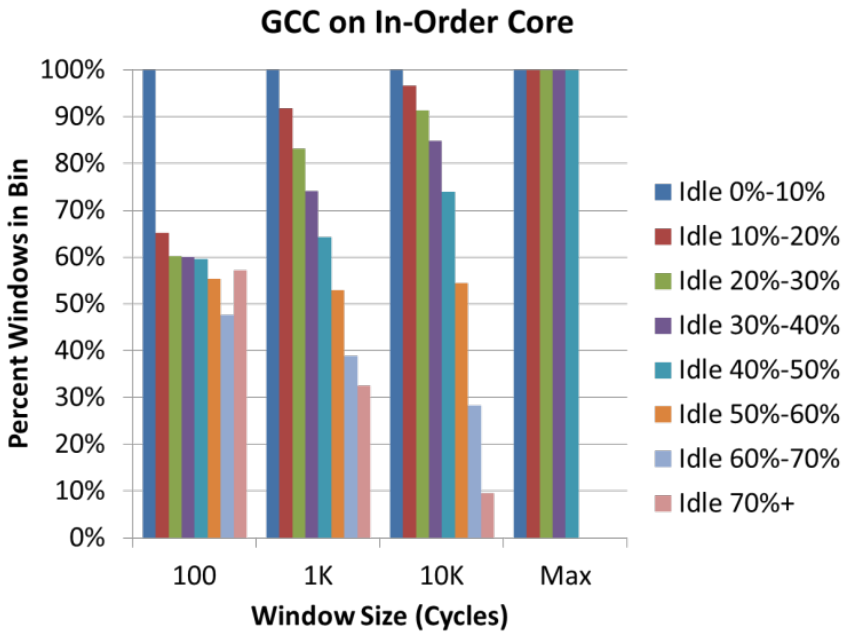


Figure 4.6: **Percent windows are idle for varying window sizes. GCC benchmark on in-order core.** Approximately 55% of windows are idle more than not (idle cycles 50%+) with window size of 100.

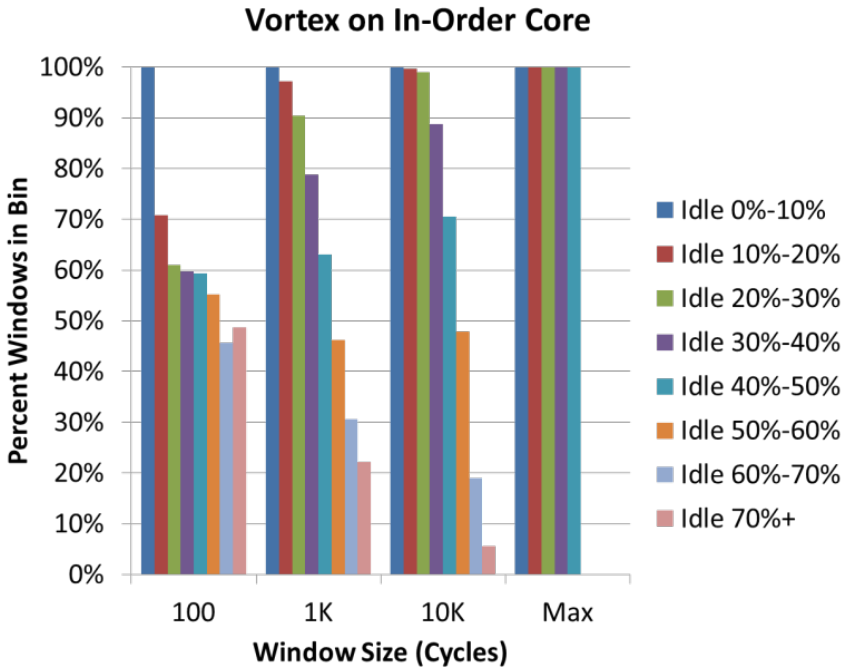


Figure 4.7: **Percent windows are idle for varying window sizes. Vortex benchmark on in-order core.** Approximately 55% of windows are idle more than not (idle cycles 50%+) with window size of 100.

Table 4.3: Percentage of windows with idleness > 50% for varying window sizes. Both in-order and out-of-order are included.

Benchmark	100		1K		10K		Max	
	IO	O3	IO	O3	IO	O3	IO	O3
GCC	55%	32%	53%	28%	54%	11%	0%	0%
Vortex	55%	26%	46%	21%	48%	3%	0%	0%
MCF	84%	8%	85%	6%	84%	5%	100%	0%
TWOLF	1%	5%	1%	3%	0%	0%	0%	0%
Average	49%	18%	46%	15%	47%	5%	25%	0%

4.3.3 Fast Boosting

During windows of high idleness, it may be possible to voltage scale the core or migrate data to a smaller core and save energy. Under ideal conditions where DVFS has no overhead, energy savings are computed by

$$E_{eff,total} = (1 - P_l) + P_l * E_{eff,slow}$$

where $E_{eff,total}$ is total energy-efficiency improvement across the entire benchmark, $E_{eff,slow}$ is energy-efficiency improvement for a slow window (assumed to be 2.5× for voltage scaling) and P_l is percentage of time running at a low-frequency during a period of high idleness. For example, if $P_l = 50\%$ then $E_{eff,total} = 1.75$ indicating an energy-efficiency improvement of 75%. Table 4.4 lists the energy-efficiency improvements for the four SPEC2000 benchmarks. With a window size of 100 energy-efficiency is improved by 30% on average for out-of-order and 70% for in-order. GCC for out-of-order has a 50% improvement with a window size of 100, but diminishes to 9% with a window size of 100,000 cycles.

Table 4.4: Ideal energy improvement using fast boosting with idleness > 50% for varying window sizes. Both in-order and out-of-order are included.

Benchmark	100		1K		10K		Max	
	IO	O3	IO	O3	IO	O3	IO	O3
GCC	1.8×	1.5×	1.8×	1.4×	2.62×	1.33×	1.0×	1.0×
Vortex	1.8×	1.4×	1.7×	1.3×	2.44×	1.09×	1.0×	1.0×
MCF	2.3×	1.1×	2.3×	1.1×	3.52×	1.15×	2.5×	1.0×
TWOLF	1.0×	1.1×	1.0×	1.0×	1.0×	1.0×	1.0×	1.0×
Average	1.7×	1.3×	1.7×	1.2×	1.7×	1.1×	1.4×	1.0×

Figure 4.8 plots the average energy-efficiency improvement across window sizes for in-order and out-of-order. Again, a smaller window size corresponds to a higher energy efficiency

improvement for out-of-order. This indicates that a fast boosting technique requiring 10s of cycles would be able to extract idleness and improve energy efficiency. Traditional DVFS with switching times of 100,000s of cycles are unable to extract idleness for out-of-order.

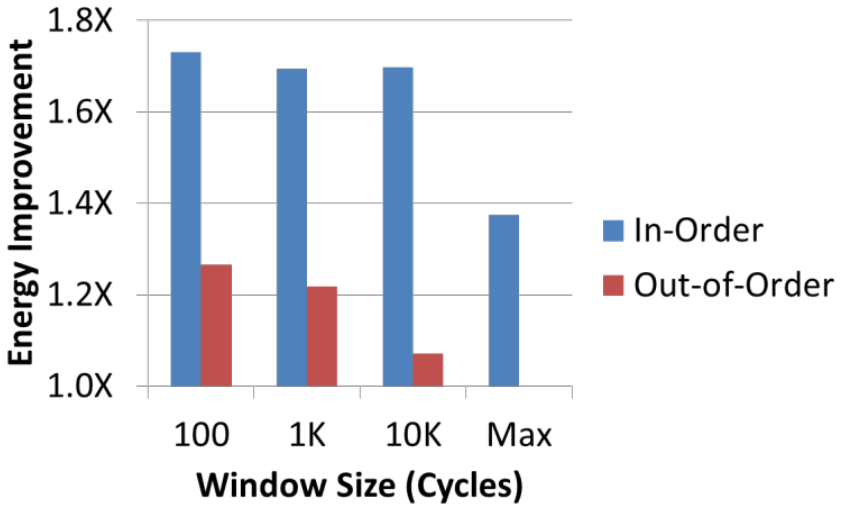


Figure 4.8: **Ideal energy improvement using fast boosting with idleness $> 50\%$ for varying window sizes and averaged across four SPEC2000 benchmarks.** Both in-order and out-of-order results are shown.

4.4 Conclusions

Extracting idleness can improve energy-efficiency through voltage scaling or migrating cores. However, the granularity is fine (typically on the order of 100s or cycles) on an out-of-core for many benchmarks. This granularity is 3-4 magnitudes smaller than current DVFS implementations allow, thus fast boosting benefit extractable energy efficiency by operating at a much finer granularity. Simply scaling voltage for the entire run of a program will severely impact windows of low-idleness and thus does not improve energy-efficiency.

Current heterogenous architecture techniques [55], though an order of magnitude quicker than off-chip DVFS, require roughly 20,000 cycles to migrate data and thus cannot extract substantial idleness. If migration is able to operate in 10s of cycles with low overhead, then it could potentially save more energy than fast boosting alone. For GCC on an out-of-order core, with a window size of 100 cycles, the improved energy-efficiency is 30% using fast

boosting and 40% with Big.LITTLE. Combining fast boosting with Big.LITTLE has the potential for even better energy-efficiency.

SPEC2000 benchmarks were simulated for this study, but characterizing a mobile application benchmark (such as BBench) would indicate the amount of extractable idleness and guide high-level architectural design. Additionally, idleness of a single-thread was explored but many applications include multiple threads running on many cores and with simultaneous multithreading. This would reduce the amount of useful idleness, though the trends would remain. Finally, overheads for voltage boosting and data migration were not included in this study, but could be amortized over many cycles for large window sizes and so would reduce obtainable energy-efficiency for small window sizes.

CHAPTER 5

Shortstop: An On-Chip Fast Supply Boosting Technique

5.1 Motivation

Transistor threshold voltages have stagnated in recent technology nodes, deviating from constant-voltage scaling theory and directly limiting supply voltage scaling. To overcome the resulting energy and power dissipation barriers, energy efficiency is improved through aggressive voltage scaling, and recently there is increased interest in operating at "near-threshold" supply voltages [6]. In this region sizable energy gains are achieved with moderate performance loss for parallel applications.

Even for applications that parallelize fairly well, serial portions of code remain. In a near-threshold scenario where most cores run at low voltage, it is therefore advantageous to rapidly increase core voltage to address the need for fast execution of these serial fragments [21] and respond to varying workloads. Such dynamic voltage and frequency scaling (DVFS) is traditionally implemented with an off-chip regulator, which requires hundreds or thousands of CPU cycles to transition to, and stabilize at, a new voltage [57]. Thus, it is not suitable for fast voltage control that responds to fine grain code sequences. To improve performance, on-chip low-dropout regulators have been proposed at the expense of degraded energy efficiency and overall power dissipation. Recent work [58, 59] used on-chip regulators to improve speed. Alternatively, DVFS can be implemented on-chip with multiple power

rails connected dynamically to a core with PMOS headers. This is faster than off-chip regulators and more efficient than on-chip regulators but incurs voltage droop during transitions, causing timing failures in other cores sharing the power rail.

5.2 Technique

We propose a new circuit technique Shortstop that addresses power supply droop seen by other cores while boosting a core from a low (0.4V) to high voltage (1.0V) within 26 to 142 ns for ARM M3 or Intel Atom-sized cores, respectively. Shortstop adds a second "dirty" supply rail and an on-chip boost capacitor to rapidly boost the core. The key idea is to transition the cores to high voltage using the dirty supply, thereby decoupling the transition from the clean high voltage supply and isolating other cores from supply droop. In addition, we use the dirty supply's wirebond/C4 innate parasitic inductance in a boost converter arrangement, thereby exploiting this inductance as an asset rather than barrier to fast supply transitions. Finally, on-chip decoupling capacitance is configured as a boost capacitor, further aiding supply transition. The boost capacitor and additional dirty supply are shared between multiple processors, amortizing their overhead.

The key challenge in Shortstop is to boost the supply quickly without destabilizing the power rails used by other cores. A PMOS header implementation has three drawbacks: 1) unavoidable wirebond/C4 inductance creates droop and ringing during fast switching; 2) droop must be small (e.g., $< 10\%$) so cores sharing a power rail are not disturbed; 3) adding on-chip decoupling capacitance to the power rail to reduce droop and ringing incurs large area costs.

Shortstop addresses these issues through the use of dirty VDD, Vdirty, (Figure 5.1). Vdirty is connected to the high supply voltage (e.g., 1V) off-chip and does not require additional off-chip regulation. Since the Vdirty supply is used only for transitioning a single core at a time, it does not need to be as robust as nominal operating supplies. This greatly reduces overhead since Vdirty need only use a small number of pads that are amortized across many cores. Shortstop consists of one header block per core and a single shared boost block. Figure 5.2 illustrates the basic operation of Shortstop. A core is initially connected

to the low voltage supply Vlow and the on-chip capacitor is charged to Vhigh (Step 1). The core is then switched from Vlow to the capacitor, which partially boosts the core while Vdirty is simultaneously shorted to a dirty ground to energize Vdirty's inductance, similar to boost converter operation (Step 2). The on-chip capacitor must be large, similar to the intrinsic capacitance of a core, but is shared across several cores to reduce area overhead. Once charge sharing is complete the core is switched from the capacitor to Vdirty supply (Step 3). Since by this time, significant energy has built up in the Vdirty inductor, Vdirty quickly boosts the core to full voltage.

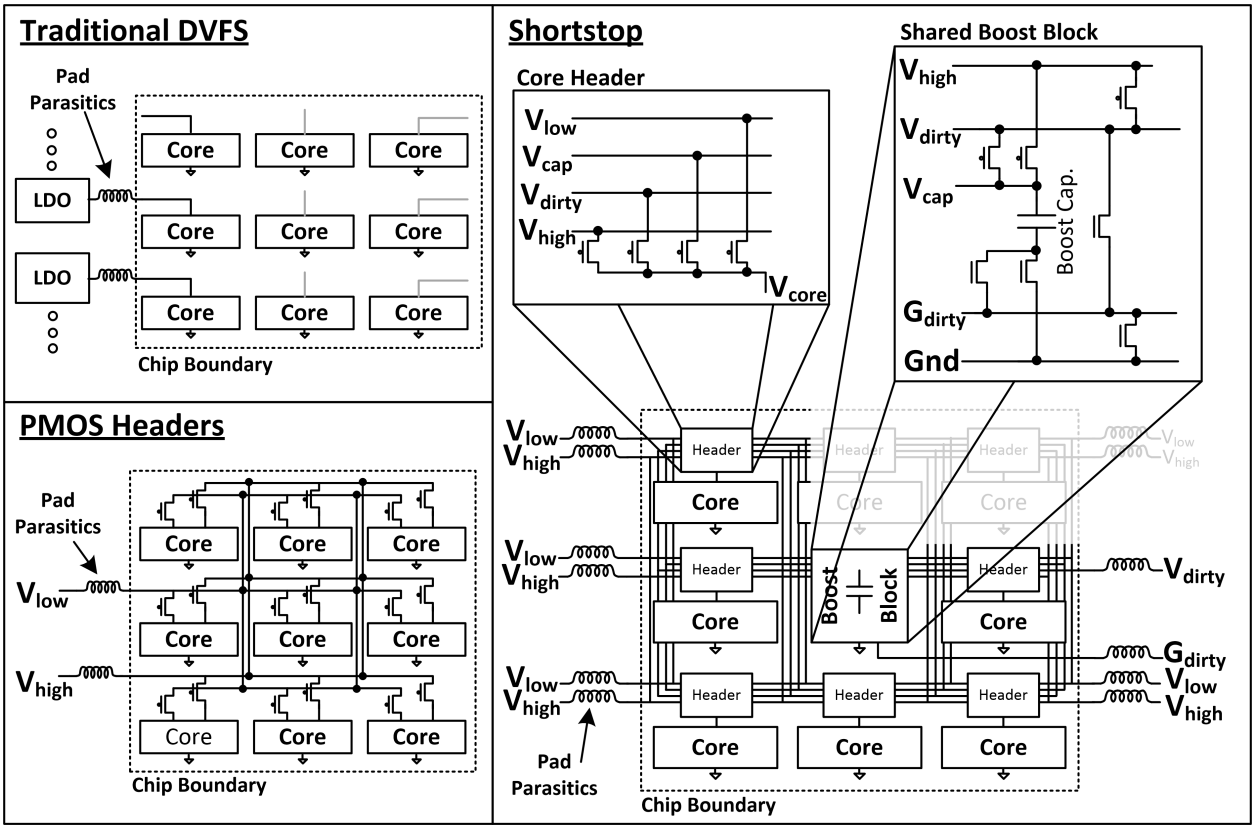


Figure 5.1: **Shortstop high-level concept compared to other boost approaches.** Traditional DVFS uses off-chip regulators to adjust core voltage (top left), but takes 100s - 1000s of cycles to adjust. A PMOS header approach (bottom left) is very fast, requiring a handful of cycles, but destabilizes the high power supply through droop and ringing. Shortstop (right) uses PMOS headers, but adds a Vdirty power supply acting as a boost converter and an on-chip boost capacitor to boost a core in several cycles without destabilizing the Vhigh supply.

As the core reaches the target high-voltage supply, it is switched from Vdirty to the nominal Vhigh supply (Step 4). Since the core is already charged to a level near Vhigh, this

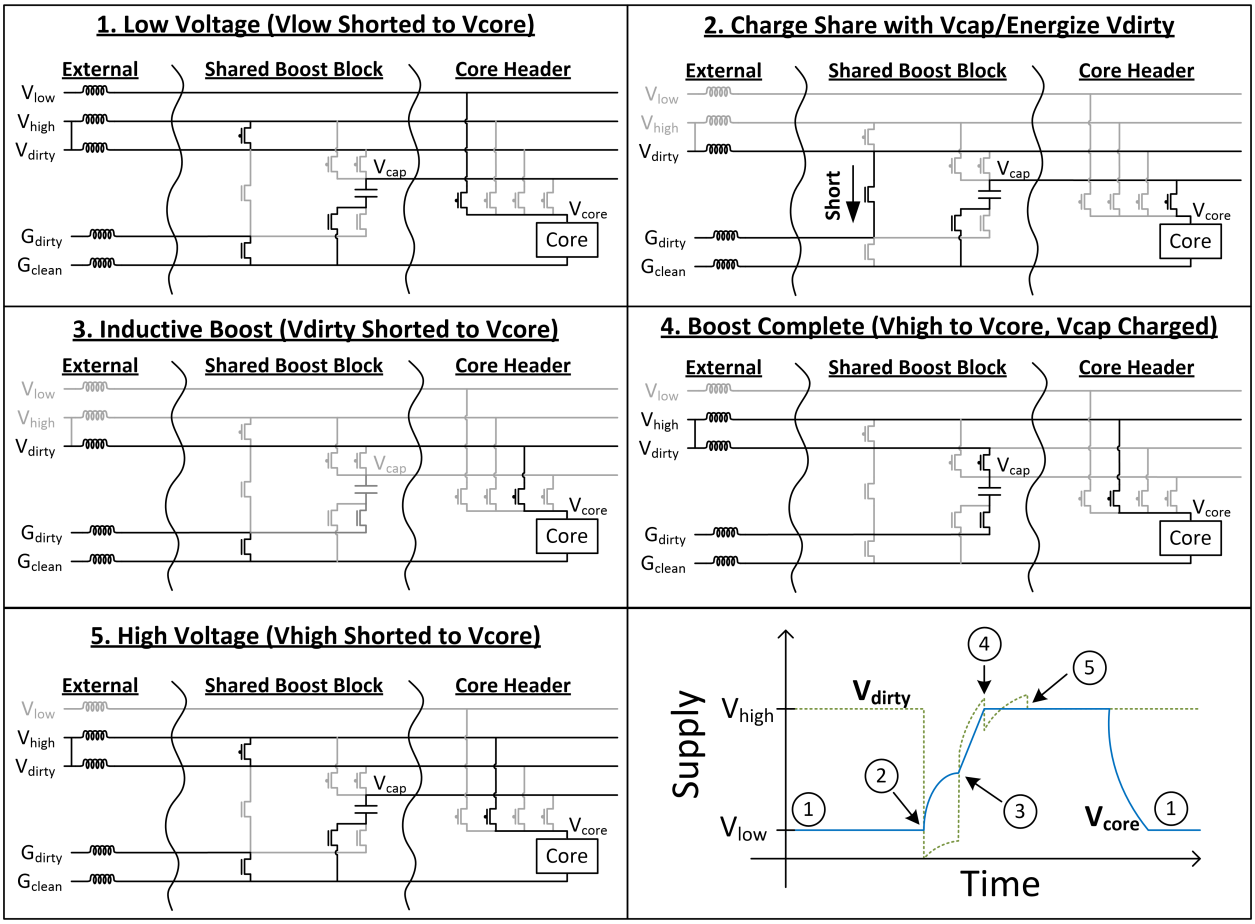


Figure 5.2: **Steps of Shortstop to boost core supply rail.** The five steps include: 1) The core is initially connected to the low voltage supply; 2) the boost cycle begins by charge sharing between a core and on-chip capacitor for a fast initial boost, while a dirty supply is shorted to ground to energize the wirebond/C4 parasitic inductance; 3) after charge sharing completes, the core is boosted the remaining amount to the high voltage by shorting to the Vdirty supply rail; 4) when the core reaches the target high voltage, it is switched to the stable Vhigh supply and the on-chip capacitor is connected to the Vdirty rail to quickly recharge; 5) once the on-chip capacitor is charged to its peak value, the Vdirty supply is disconnected and clamped to the Vhigh supply to prevent ringing.

step does not incur significant droop or otherwise destabilize the Vhigh supply, which thus can be shared by a number of other cores. However, Vdirty will incur significant ringing and actually overshoot the high supply voltage, which is undesirable. Two techniques are used to avoid this ringing: 1) when Vdirty is disconnected from the core, it is immediately connected to the on-chip capacitor to use the remaining wirebond/C4 inductance energy to charge the capacitor, preparing it to boost another core. When Vdirty has transferred its energy to the boost capacitor and reaches its maximum voltage, Vdirty is disconnected and clamped to Vhigh to immediately suppress any further ringing (Step 5). Since Vdirty's inductor has was discharged when clamped, and Vdirty has no on-chip decoupling capacitance, this step does not disturb Vhigh.

The Shortstop boosting steps must be timed accurately (100s of ps) to function efficiently. This is accomplished using programmable on-chip delay generators that are tuned for a particular package and chip configuration. Alternatively, high-speed comparators [60] could be used in an automated timing architecture. The on-chip timing circuitry includes a 1.25 GHz asynchronous clock generated by a ring oscillator, 16 delay generators with fine (25 ps) and coarse delay (800 ps) steps, and maskable XOR trees that combine multiple timing signals into arbitrary digital waveforms for the switches. The test chip architecture (Figure 5.8) includes two on-chip variable capacitors/current sources to emulate large cores, as well as an actual implemented ARM Cortex M3 core. In addition to timing control circuits, headers, and a boost capacitor, on-chip samplers monitor power rails using a sample-and-hold averaging technique [61] that enables an effective bandwidth of $\sim 40\,\text{GHz}$.

Shortstop requires precise timing of header and footer switches for correct and efficient boosts of a processor core's power rail, with timing accuracy on the order of 100ps in a wirebond version [53]. Failure to correctly time switches can lead to excessive ringing, shorting, and droops, which waste energy, decrease efficiency, and can lead timing violations.

Shortstop includes two elements for accurately timed switch control, delay generators and maskable XOR trees, shown in Figure 5.3. The timing system is triggered from a latched *boost_go* signal and includes 16 delay generators that each assert after a chosen amount in time, set by a scannable configuration bits. The maskable XOR trees select a subset of these delayed signals to generate pulsed switch enables, shown in Figure 5.4. In the example, three

delay generators assert at three points in time: T1, T2, and T3. The XOR combines all three delay generators to assert a pulse from T1 to T2, and reassert at T3. The XOR could have been set to exclude delay generator 1, in this case the XOR output would have been a pulse asserted from T2 to T3.

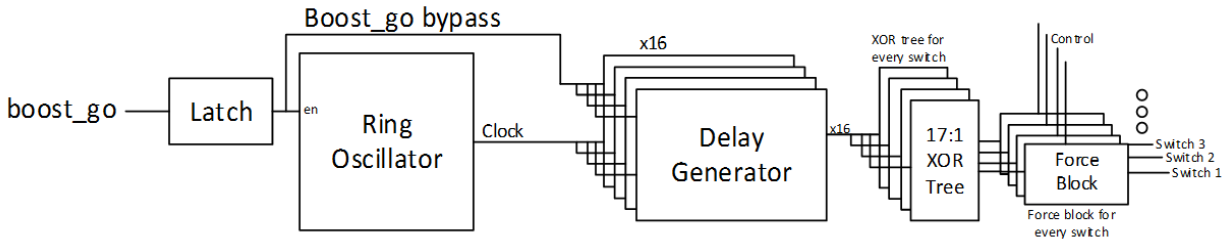


Figure 5.3: **Complete system of delay generators, XOR trees, force blocks.** A shared ring oscillator, enabled with the *boost_go* signal is used to provide coarse delay adjustment between the delay generators. Since the delay generators count edges of the ring oscillator, all of the delay generators are synchronized for coarse adjustment. Within each delay generator is a tunable delay chain that provides fine adjustment. Configurable XOR trees are used to generate pulses from the delay generator step outputs. Force blocks directly before the switches are used to prevent glitching and safely reset the timing system.

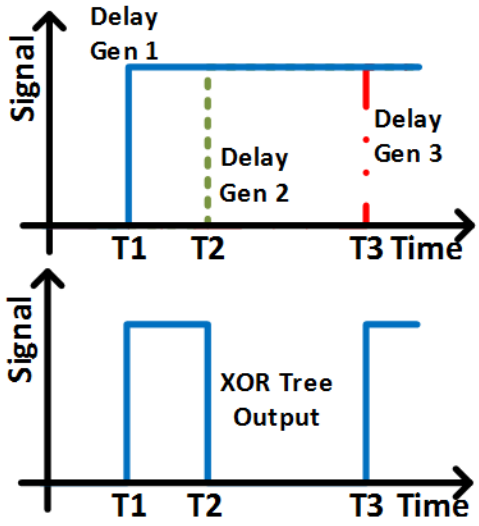


Figure 5.4: **Principle of delay chain + XOR tree operation.** Three delay chains select points in time T1, T2, and T3 (top of figure). The XOR tree combine the three points to create a pulsed output (bottom of figure).

The first silicon prototype of Shortstop [53] includes 16 delay generators, though in practice Shortstop only requires five delay generators: (1) charge share core to cap; (2) start

shorting of dirty supply rails to charge parasitic inductor; (3) connect core to dirty supply rail for inductive boost; (4) connect core to high supply rail; (5) disconnect cap from high supply rail once charged and no energy remains in parasitic inductor. The extra delay generators were included on the first prototype to provide allow flexibility in testing different boost arrangements, and because each header/footer switch was split into sub-switches of varying sizes to allow gradual actuation. The sub-switches were sized in a binary fashion, so that switch strength could be increased over time by correctly timing additional delay generators to larger switches sizes.

A block diagram of the delay generator is shown in Figure 5.5, and each generator includes fine- and course-grained controls. The fine-grain delay is adjusted by multiplexing between taps of a 31-buffer delay chain, while the course delay is adjusted by counting cycles of an asynchronous fast clock. The fast clock is enabled through the *boost_go* trigger signal and, if the coarse adjustment count is zero, this trigger is fed directly into the 31-element delay chain. The fine delay adjustment can be measured by putting the delay generator into a loopback mode, which configures the delay chain into a ring oscillator arrangement. The out of the loopback mode is divided down by a series of toggle flops, so that it can be observed off-chip. An additional loopback chain can be added to the ring oscillator so that max time violations do not occur in the toggle flops.

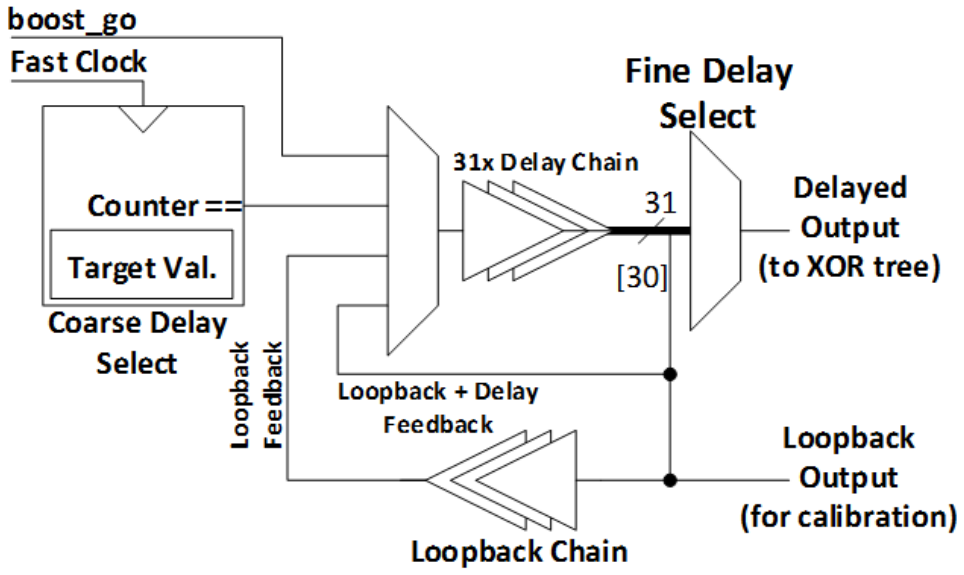


Figure 5.5: **Delay generator element.** Shortstop includes 16 of these delay elements.

A schematic of the maskable XOR tree, shown in Figure 5.6. The circuit is very simple, allowing any or all of the 16 delay generator outputs to be used to enable/disable the switches, but care must be taken that each path is as balanced as possible to not introduce skew between different delay inputs changing the output value. A 17th input to the XOR tree is used to select polarity of the XOR output, i.e. whether the output is HIGH or LOW when no delay blocks have been asserted. This is used to match whether a header/footer should be enabled or disabled at the start of a boost cycle.

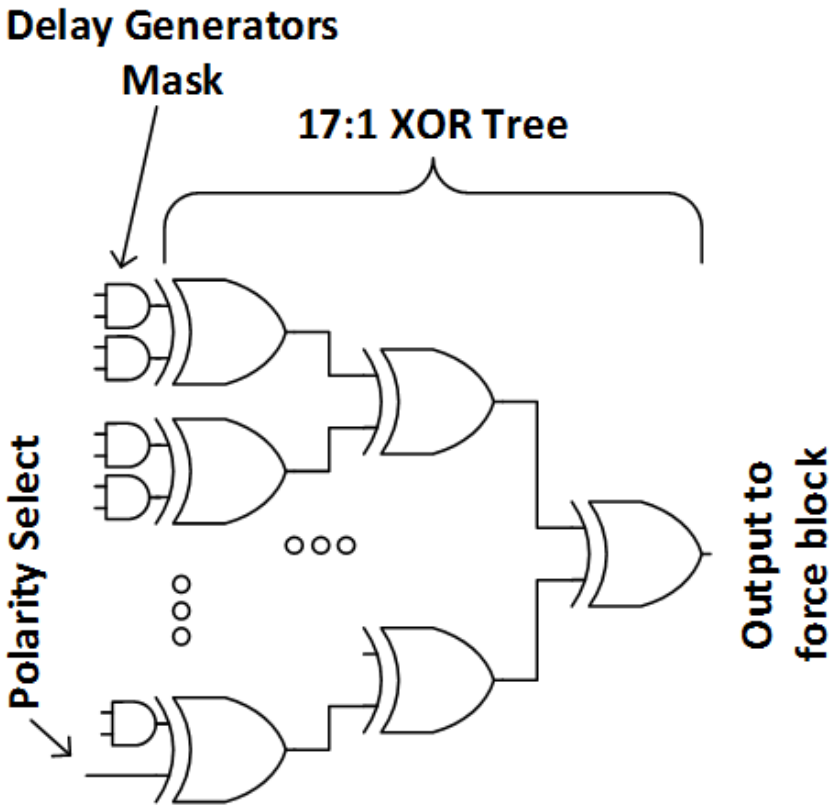


Figure 5.6: **Maskable XOR trees to select a subset of delay elements to control each switch.** An additional input controls polarity, whether the switch is active high or low.

When the delay generators are reset, the generator outputs may propagate through the XOR at different times, causing unintended glitching on the header/footer switches. This is very undesirable since the core supply rail voltage may fluctuate, or power sources may be accidentally shorted. A force block, Figure 5.7, is used between the XOR tree and the header/footer switches, where a HIGH or LOW value is "forced" glitch-free until the delay

circuitry has been reset completely. A scan chain bit controls the *force_bit* value of the output, and a *force_en* is driven by a control FSM while the delay circuitry is being reset. This ensures the switch inputs are held at stable values in between boosts.

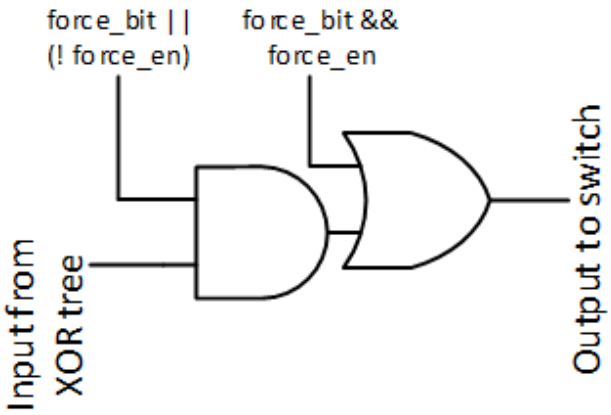


Figure 5.7: **Force block to prevent glitching on switches.** The scan signal *force_bit* configures the value of the bit, while *force_en* actuates forcing the output to the switch to the programmed value.

5.3 Measured Results

Shortstop is validated in a 28nm CMOS test chip measuring 3.9 mm^2 (Figure 5.9). Table 5.1 summarizes the test chip's specifications. The chip is wirebonded to an 88-pin QFN package and a 108-pin ceramic PGA package with two wirebond lengths to vary package parasitics. Figure 5.10 shows silicon measurements comparing boosting time for the included M3 core using a baseline PMOS header based approach and Shortstop. The 1-pin baseline assumes Shortstop's hardware overhead can be amortized across multiple cores and hence is negligible, while the 2-pin baseline is a conservative estimate where the number of dirty supply pins equals the number of high supply pins. For the M3 core, boost latency and droop are improved by 1.7× and 6×, respectively.

Figure 5.11 compares supply droop and boost latency, defined by rise time within 10% of Vdd, for the baselines and Shortstop across different emulated core sizes. As core size decreases, Shortstop exhibits slightly increased gains against baselines, while supply droop is relatively constant at 6× and 3× for the 1-pin and 2-pin baseline, respectively. For a

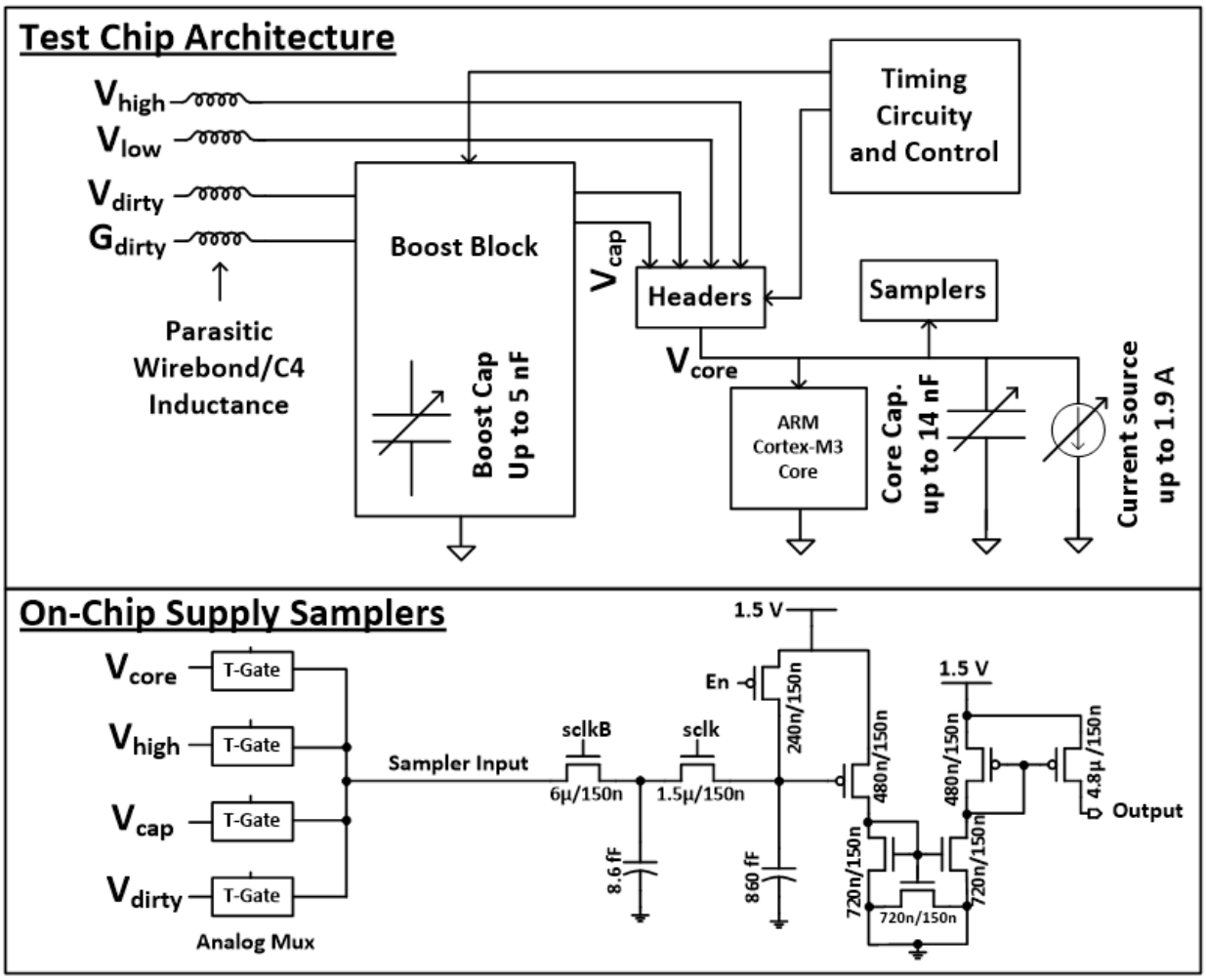


Figure 5.8: **Test chip architecture for Shortstop.** The Shortstop test chip architecture includes on-chip variable boost and core capacitors to mimic large cores. An ARM Cortex-M3 and on-chip samplers are also included on the test chip. The on chip samplers are based on [61] can observe power supply transients by 20 point averaging a sampled supply voltage provided by an analog mux. The sampler was sized to hold values for roughly 1 ms with minimal leakage so that boosting experiments can be repeated and observed.

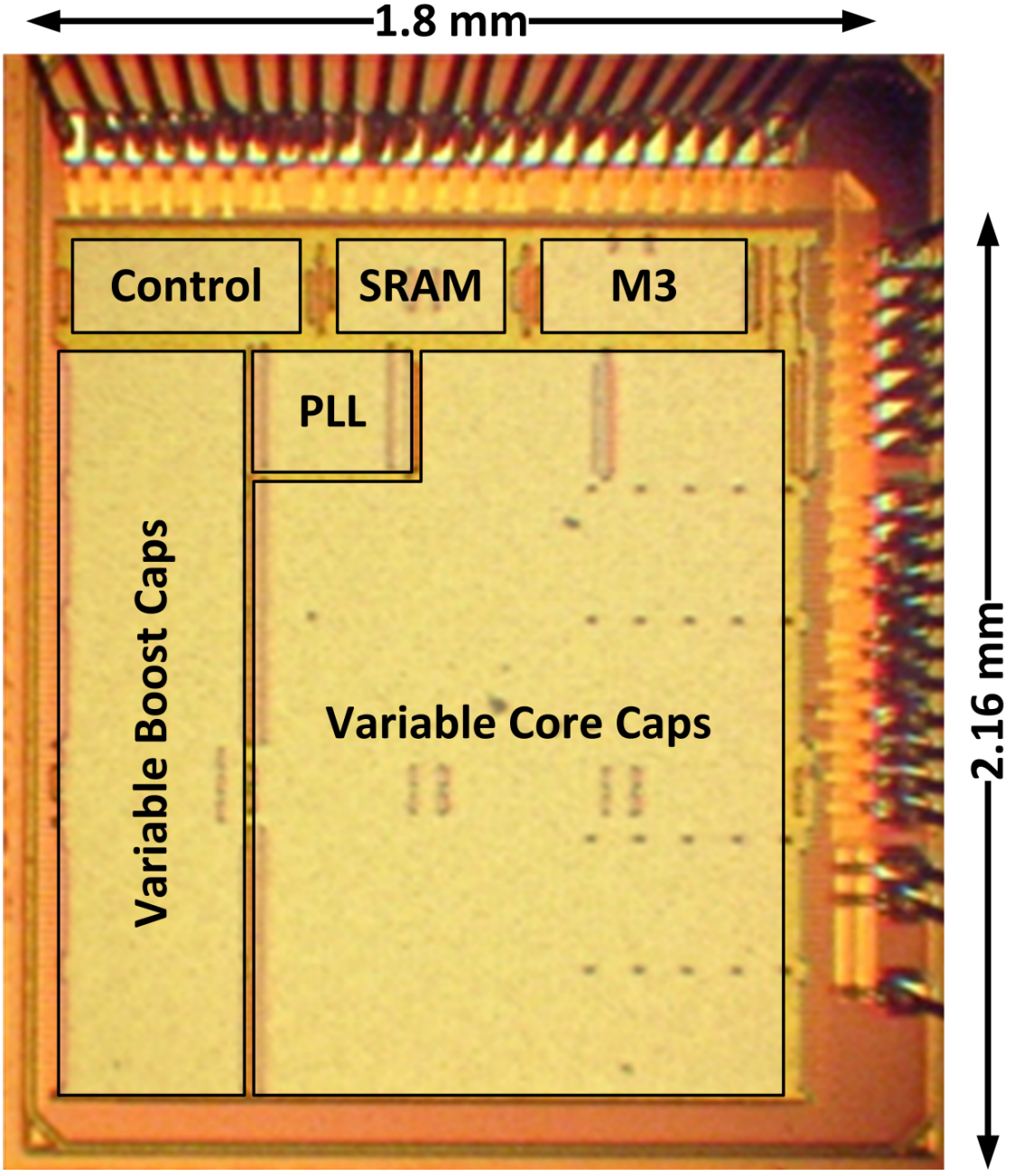
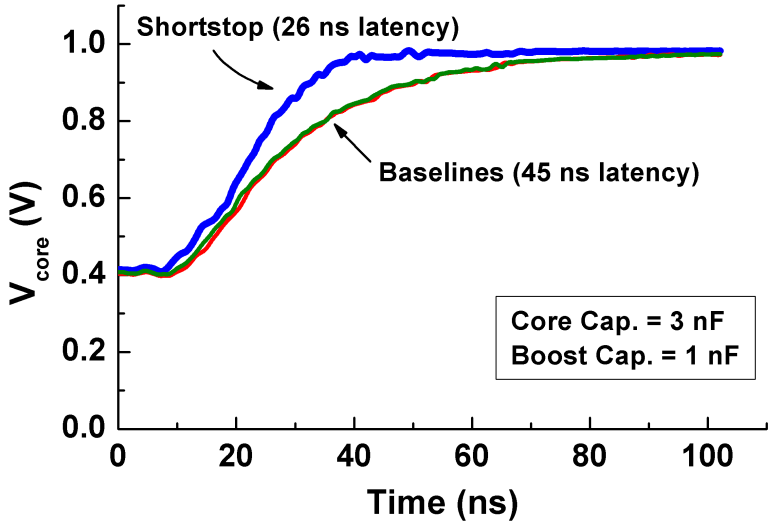
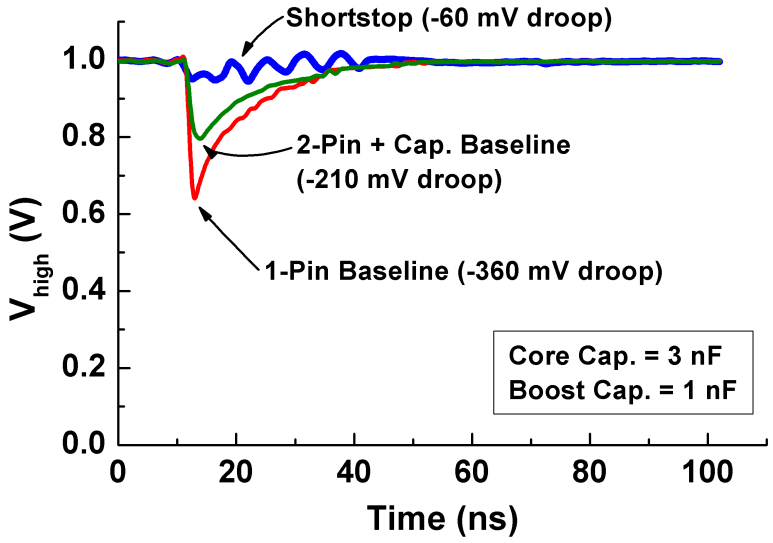


Figure 5.9: **Photomicrograph of 28nm test chip and chip specifications.** The chip measured 3.9 mm^2 and was wirebonded to CPGA and QFN packages. Variable core and boost capacitors dominate chip area, to emulate different boosting scenarios.



(a) Vcore on-chip measurement.



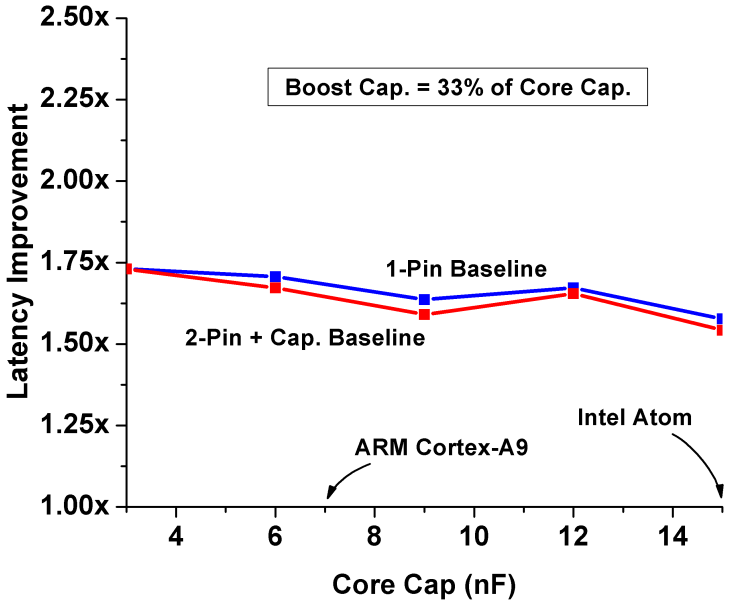
(b) Vhigh on-chip measurement.

Figure 5.10: **Measured rail voltages for 3 nF core (QFN package).** For the M3 core, boost latency is improved by 1.7× and and droop reduced by 6×.

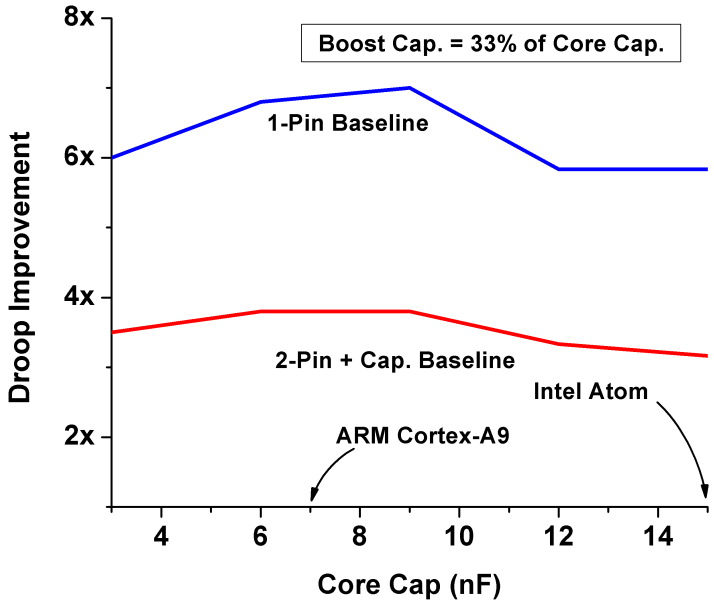
Table 5.1: Shortstop Test Chip Specifications.

Technology	28 nm
Area	3.9 mm^2
Processor Core	ARM Cortex-M3
Max Core Cap.	15 nF
Max Boost Cap.	5 nF
Package Variants	88-Pin QFN (short bond wires)
	108-Pin CPGA (med. bond wires)
	108-Pin CPGA (long bond wires)

15 nF core (an Intel Atom-sized core), boost latency is improved by 1.6× in addition to a 6× droop reduction. Figure 5.12 shows the impact of boost capacitance size on supply droop and latency indicating that 30-40% of intrinsic core capacitance is sufficient to obtain most of Shortstop's performance gains. Finally, Figure 5.13 shows Shortstop maintains a 1.4× latency improvement and 4× droop reduction across the three packages tested. As package parasitics decrease, the baseline latency and droop improves but this is balanced by decreased parasitics on the dirty supply which shortens boost time to energize the parasitic inductance.

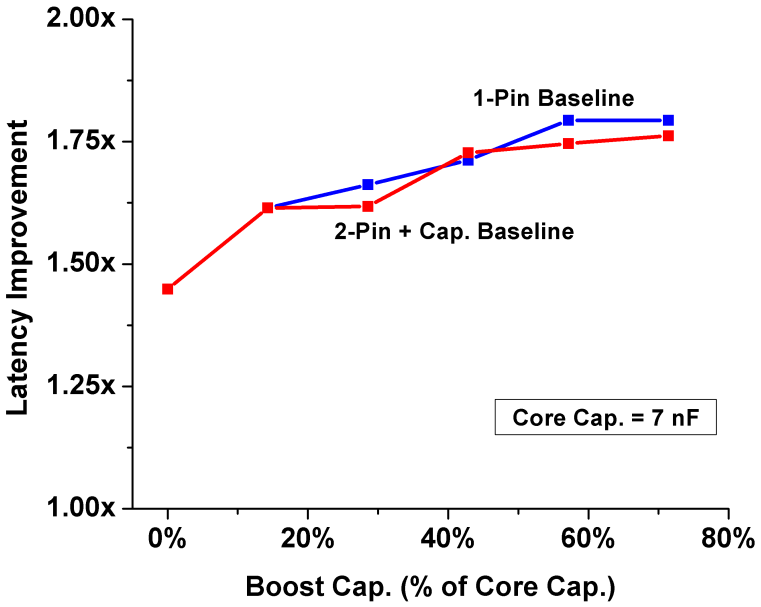


(a) Latency improvement vs core capacitance.

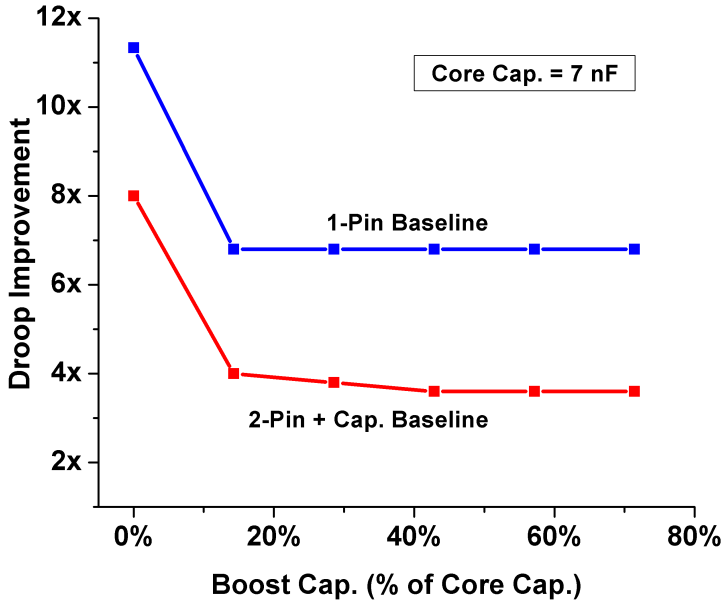


(b) Droop improvement vs core capacitance.

Figure 5.11: **Measured latency/droop improvement for varying core cap. (QFN package).** As core size decreases, Shortstop exhibits slightly increased gains against baselines. Supply droop is relatively constant with core size.

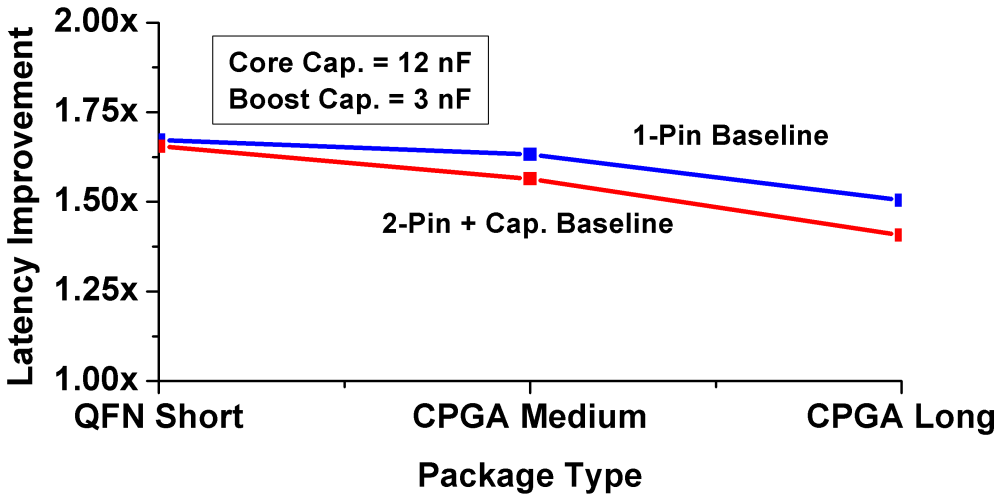


(a) Latency improvement vs boost capacitor as a percentage of core capacitance.

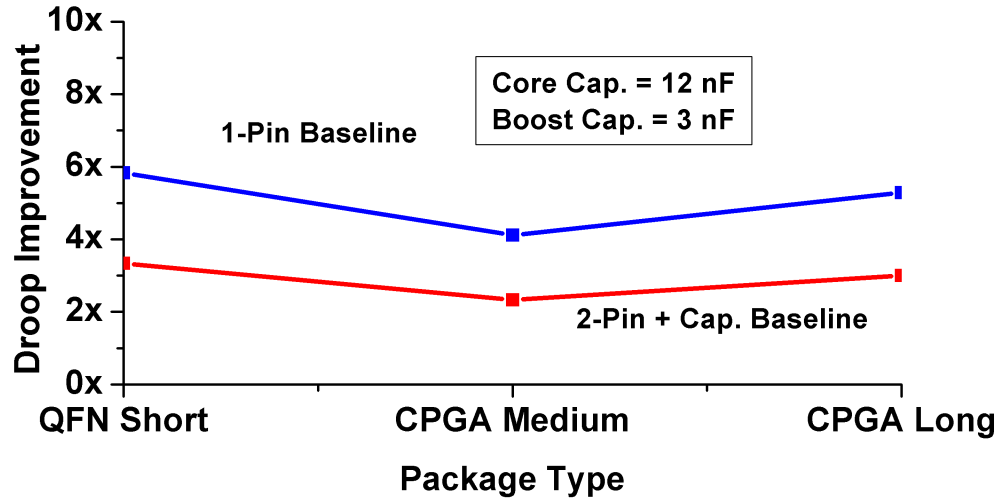


(b) Droop improvement vs boost capacitor as a percentage of core capacitance.

Figure 5.12: **Measured droop and latency for varying boost cap. (QFN package).** A boost cap sized for 30-40% of intrinsic core capacitance is sufficient to achieve most of Shortstop's performance gains.



(a) Latency improvement vs wirebond length.



(b) Droop improvement vs wirebond length.

Figure 5.13: **Measured performance improvements with varying packages and wirebond lengths.** Shortstop is fairly insensitive to these wirebond lengths.

CHAPTER 6

Shortstop FC: A Fast Boosting Flip-Chip Implementation

6.1 Motivation

The fast power supply boosting technique presented in Chapter 5 was targeted for wire-bond designs. However, modern processors, both high-performance server and mobile, are packaged as flip-chip. Instead of thin wires with relatively high inductance, flip-chip packaging uses bumps to connect high-density power and I/O signals to a substrate or circuit board. Because Shortstop leverages the parasitic inductance of a package, demonstration in flip-chip, with lower parasitic inductance, is necessary.

The second-generation design of the technique from Chapter 5, named *Shortstop FC*, and is an evolution of the previous prototype that has been extended to include a flip-chip implementation, improved power distribution and boost topology, and an automated tuning circuit to calibrate delay generators used with the boosting technique.

6.2 Improved Architecture

The Shortstop architecture was improved to reduce the number of headers required in the core area and metallization usage in the power grid by containing on-chip boost capacitor and transient dirty supply power rails within the shared boost block instead of over the

core area, Figure 6.1. Instead an intermediate on-chip V_{boost} supply is distributed over core areas, which is connected between the boost capacitor and dirty supply within the boost block. This reduces the need for one PMOS head in the core areas at the cost of two PMOS headers in the shared boost block. However, since there are many more cores than shared boost blocks less area is consumed with the core area for power switches.

An important goal of the Shortstop FC architecture is improved power delivery. A bottleneck of the wirebond prototype (Chapter 5) was centrally located power switch headers and footers, which required longer power stripes and a weaker power grid than a traditional design would require. This led to increased IR drop in the power delivery network. To remedy this, Shortstop FC distributes the power switches across core area on a chip, not unlike standard power gated designs.

The boost technique was updated to include the state of the new topology, shown in Figure 6.2, but otherwise unchanged from the original Shortstop technique. The state of the transistors in shared boost block is shown on the left of each step in the figure and the core header switch shown on the right. Unlike the original Shortstop technique, the V_{boost} header within the core domains is used during the boost transition, while the shared boost block multiplexes V_{boost} between the on-chip boost capacitor and the V_{dirty} transient boost supply.

6.3 Automatic Tuning Algorithm

Tuning of the first Shortstop prototype is done by hand through scannable flip-flops to change delay values, and observing the supply rails with on-chip, continuous-time samplers. However, this is impractical for a production design, since expensive testing time is required, instead an automated tuning algorithm is preferable to automatically set delay values. A commercial implementation of Shortstop could implement a simple finite state machine (FSM) paired with an on-chip comparator to tune the delay generators and minimize boost latency, as proposed here in Figure 6.3.

The on-chip comparator's negative input pin in connected to a voltage reference that is slightly below the target, clean, high-supply rail voltage, and the positive pin is connected to the core supply rail (Figure 6.3). The comparator is clocked using the core to clean,

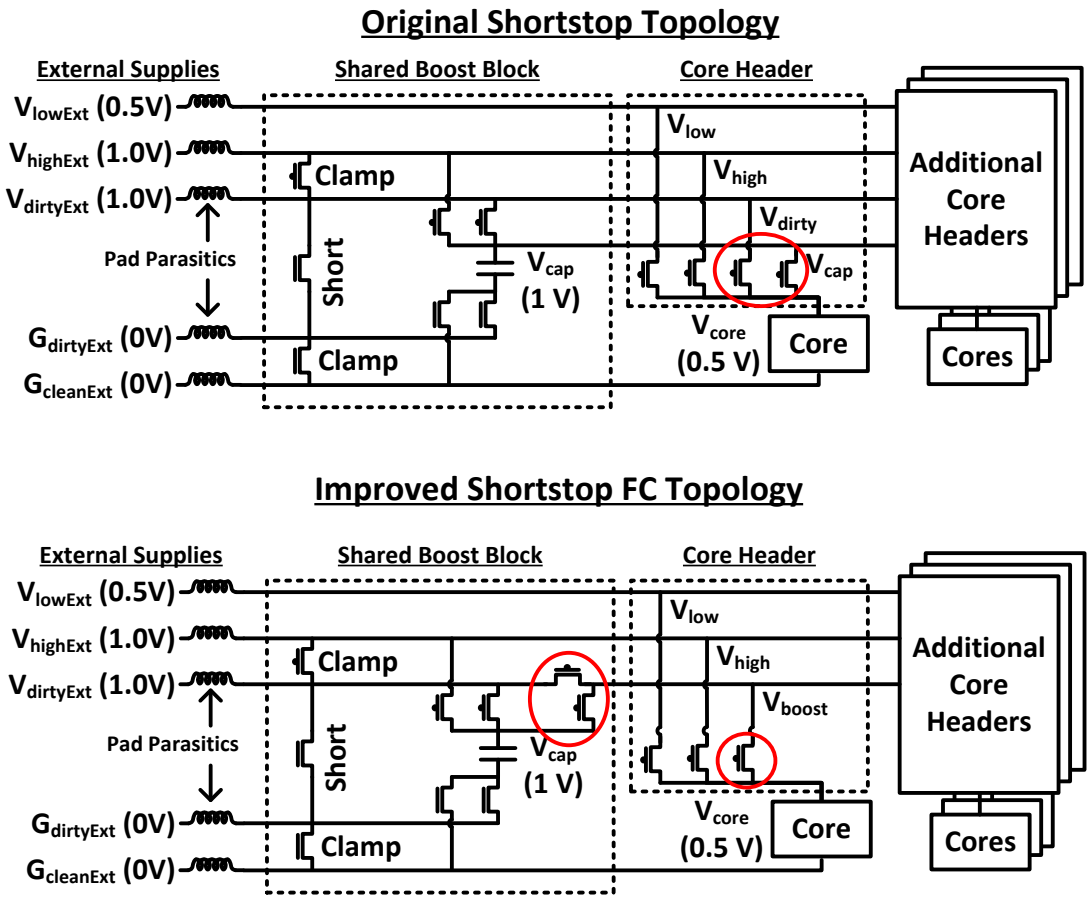


Figure 6.1: **Shortstop FC improved topology.** Core areas include three instead of four switches, at the cost of one switch in the shared boost block, to reduce area overhead.

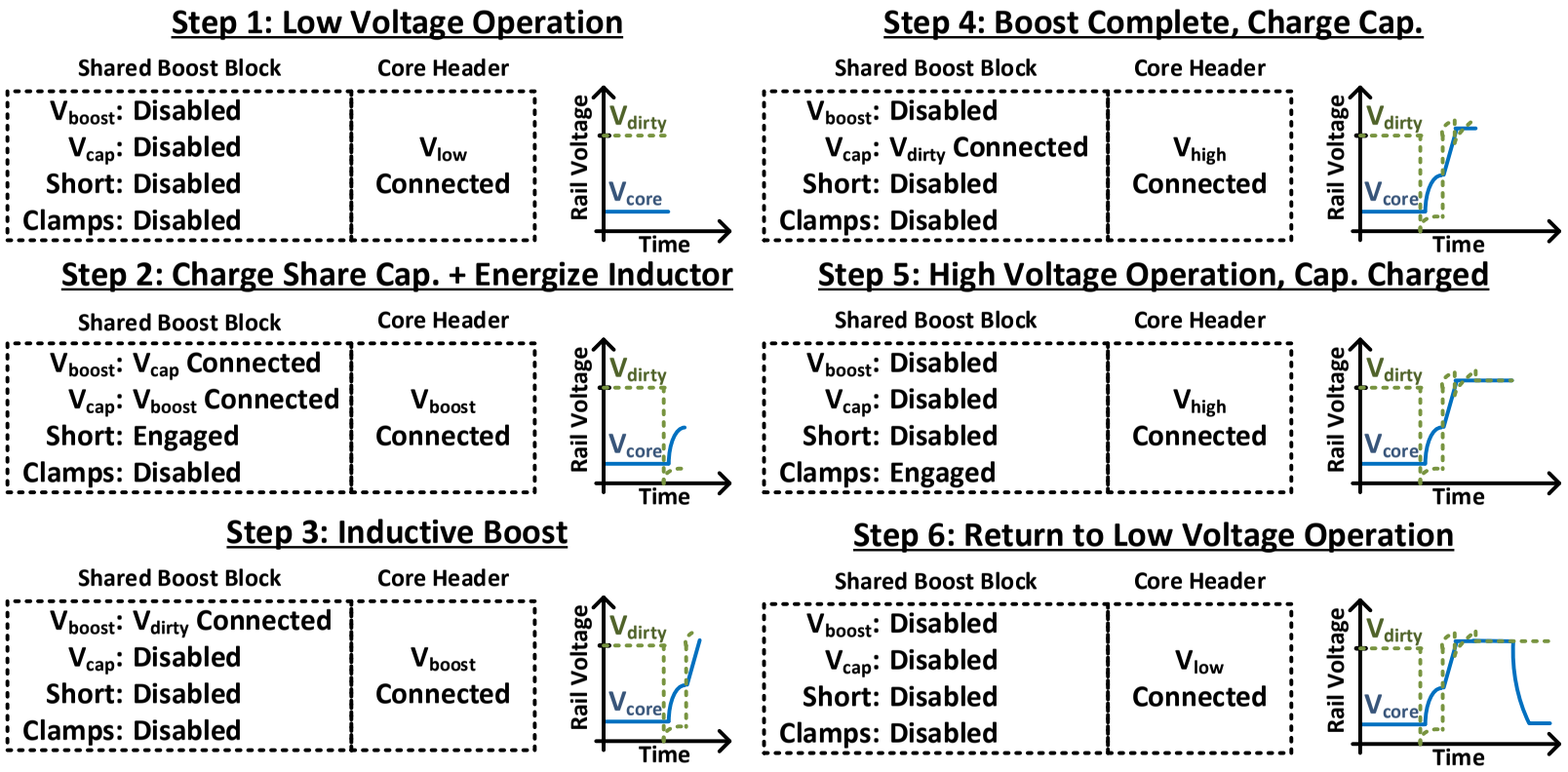


Figure 6.2: **Shortstop FC boost steps with improved topology.**

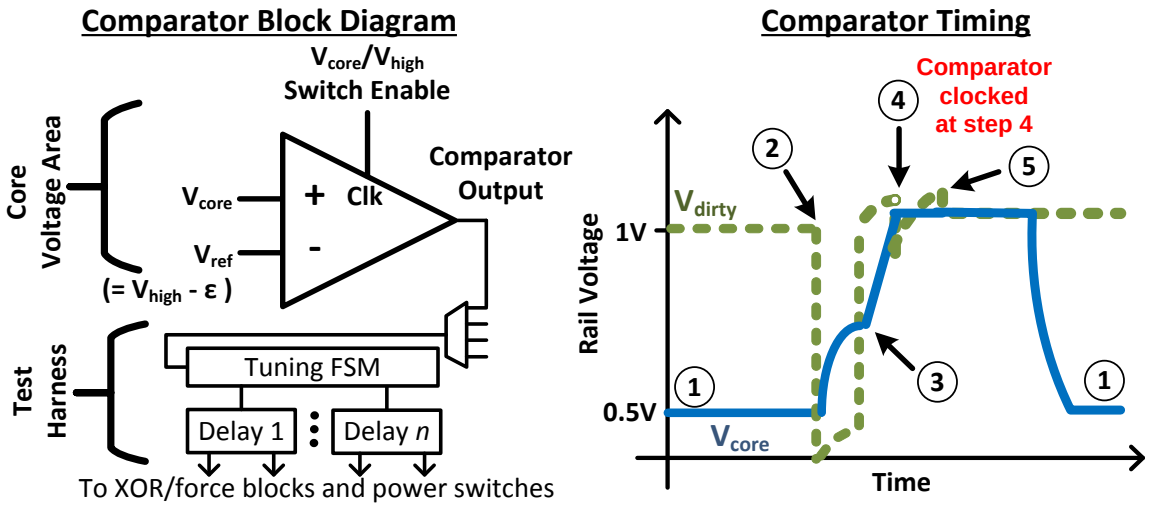


Figure 6.3: **Shortstop FC tuning concept.** A comparator is clocked on the switch of the core supply from Vdirty (Vboost with the improved architecture) to Vhigh. The comparators returns a '0' or '1' depending on if the core supply was above a droop threshold or not.

high-supply header switch enable signal, so that the comparator will always trigger when the core is connected to the clean supply rail. Thus, the comparator asserts if the core was at the clean, high-supply rail voltage when connected to the high supply rail, and deasserts if it was below. The goal is to raise the core's voltage to the high supply rail voltage before it is connected, so that droop from raising the core's intrinsic capacitance is small, while minimizing time to boost.

Figure 6.4 shows the basic steps of the tuning FSM. The FSM works by repeatedly boosting the core, reading comparator outputs, adjusting delay values, and boosting the core again with the updated delay values, until an optimal set of delays is found. In step 1, the core is connected from the low supply to dirty supply at the very start of the boost cycle, when a boost request arrives. During step 1, the delay of of connecting the core from the dirty supply to the clean, high-supply rail is gradually increased until the comparator resolves to a '1' instead of "0."

Delay values in each iteration of step 1 are increased in large steps, to reduce the time needed to tune the system. Step 2 then slowly decreases the core to clean supply delay time in fine steps, while also averaging over multiple cycles, until the comparator starts to return

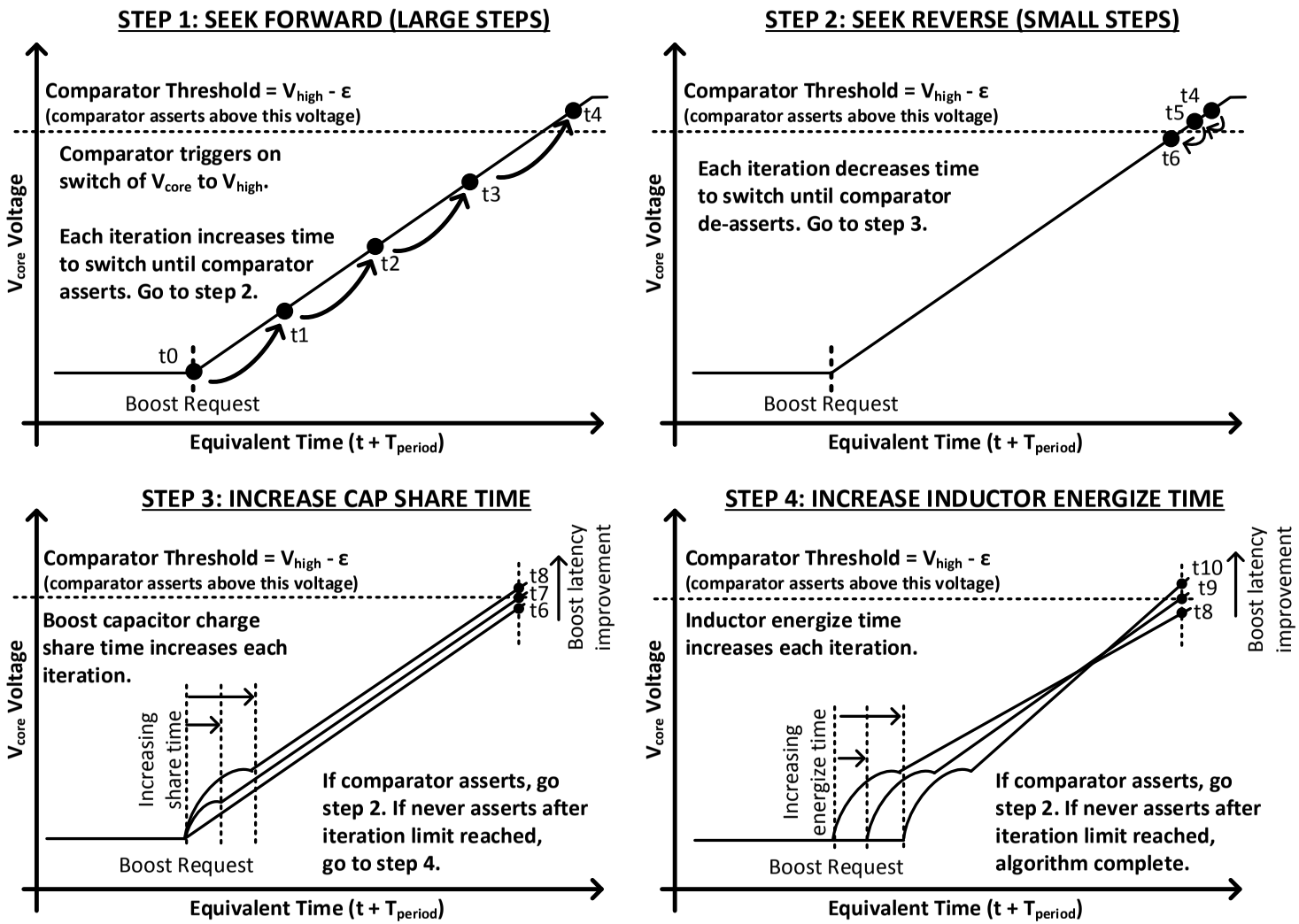


Figure 6.4: **Steps of the automatic tuning algorithm.** The four steps are used to minimize boost latency subject to a droop constraint on Vhigh.

a ‘0.’ This indicates that the core is very close to the high-supply voltage, and the delay value is used as a baseline boost latency.

The remaining steps, 3 and 4, reduce the boost latency through charging of the parasitic inductance and charge sharing of the on-chip boost capacitor with the core. Delays for the parasitic inductance energize time and boost capacitor charge sharing time on gradually increased until the comparator starts reliably asserting a ‘1’ instead of ‘0,’ indicating that the boost latency has been improved as the core supply rail is now slightly higher than when using the previous delay values. When this happens, the FSM goes back to step 2, slowly reducing the delay for the core to be connected to the clean, high-supply rail until the comparator starts to assert a ‘0,’ thereby creating a new baseline delay value for steps 3 and 4 to try to improve. This process repeats until no additional changes of charge share or inductor energize time causes the comparator to assert a ‘1,’ indicating the system is at a minimum boost latency. To better avoid local minimas, the algorithm can also be modified to decrease charge share or shorting times if increasing times do not yield improved boost latencies. In implementation the maximum amount to increase and decrease charge share and short times are completely configurable through scan bits. The remaining steps, 3 and 4, reduce the boost latency through charging of the parasitic inductance and charge sharing of the on-chip boost capacitor with the core. Delays for the parasitic inductance energize time and boost capacitor charge sharing time on gradually increased until the comparator starts reliably asserting a ‘1’ instead of ‘0,’ indicating that the boost latency has been improved as the core supply rail is now slightly higher than when using the previous delay values. When this happens, the FSM goes back to step 2, slowly reducing the delay for the core to be connected to the clean, high-supply rail until the comparator starts to assert a ‘0,’ thereby creating a new baseline delay value for steps 3 and 4 to try to improve. This process repeats until no additional changes of charge share or inductor energize time causes the comparator to assert a ‘1,’ indicating the system is at a minimum boost latency. To better avoid local minimas, the algorithm can also be modified to decrease charge share or shorting times if increasing times do not yield improved boost latencies. In implementation the maximum amount to increase and decrease charge share and short times are completely configurable through scan bits.

6.4 Physical Implementation

The test chip includes sixteen emulated core areas, of varying sizes, that can be connected and disconnected from power rails for different boosting scenarios. As mentioned above, to minimize area and metallization overhead in core areas, the boosting topology was optimized to reduce a set of header switches in the core area at the cost of two additional sets of header switches in the share boosting block. Figure 6.5 shows the proposed top-level floorplan, including the power and I/O bump pattern. Boundaries of the core areas are shown by the dashed purple, with four sized by 200 µm × 200 µm, two sized 200 µm × 400 µm, eight sized 400 µm × 400 µm, and two sized 800 µm × 800 µm. Additionally, four on-chip boost capacitors and two individual boost shorting areas are included in the center of the chip, and shared amongst the 16 cores.

Instead of a single core capacitor and boost capacitor as was included on the first Shortstop prototype, Shortstop FC is a modular design, shown in Figure 6.5. Instead of a single adjustable core capacitor, the chip is composed of many small core areas, each with adjustable current sources, fixed decoupling capacitors, and virtual supply rails. Each core supply can be individually connected to the statically high or low supply, or boosted through Shortstop. Within each core area is a test island that includes samplers and comparators for the test harness and automatic tuning circuitry. The test harness and shared boost block (with the boost capacitors) are centrally located and shared among the core areas.

As of the writing of this dissertation, a test chip implementing the revised Shortstop FC architecture in a 40nm bulk CMOS process has been designed and fabricated and is undergoing assembly. The custom BGA packaging substrate includes three on-package inductors to increase parasitic inductance of V_{dirty} supply without incurring added cost of using discrete components or on-chip inductors. Results are planned to be published in a conference or journal paper once testing is completed.

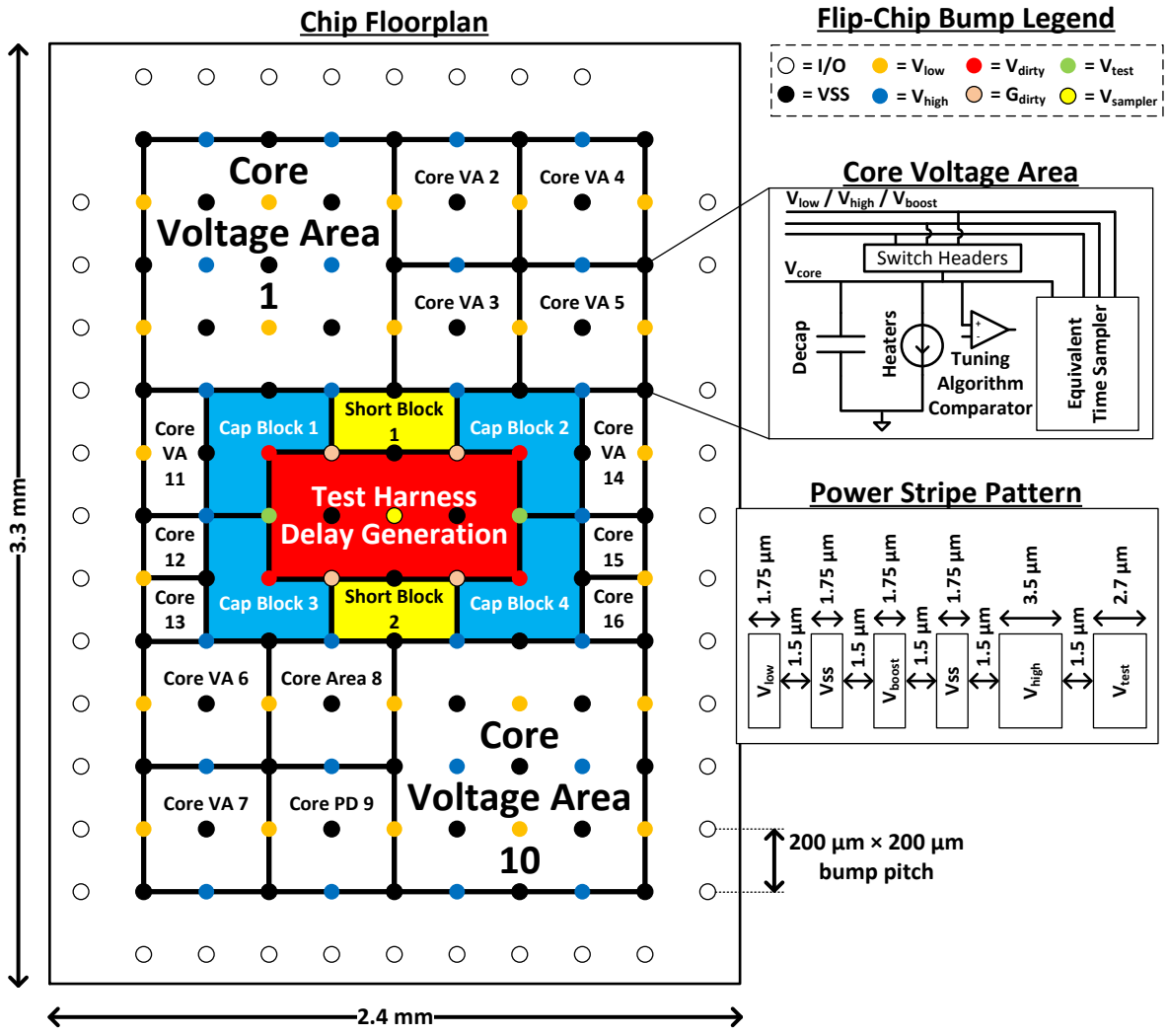


Figure 6.5: **Top-level test chip layout.** Colored circles indicate flip-chip bumps. I/O signal bumps are around the chip perimeter, while power is supplied on all inner bumps.

CHAPTER 7

Near-Threshold in FinFET Technologies: Impact of Process on Voltage Scalability

7.1 Motivation

Near-threshold (NT) was systematically defined in chapters 2 and 3, and the NT region was evaluated across planar nodes ($180 - 32$ nm) to observe technology trends. However, power density is becoming an even greater problem as transistors continue to shrink in size. Figure 7.1 shows example scaling from 40nm to 7nm, derived from publicly available data. In 7nm, power of a core logic has improved by roughly 4.7× yet can fit in an area 19× smaller, thus power density has increased by 4.2×. As shown in Chapter 2, NTC energy improvement is becoming less effective with each generation in planar nodes, with only a 4× energy gain in 32nm for performance sensitive workloads. While this gain is not insignificant, NTC is needed most in new technology nodes because of increased power density, yet energy gain in 32nm is nearly half of the gain in 180nm (7.5×).

Foundries have initiated a fundamental switch from planar to FinFET transistors at the $22 - 16$ nm node and below, opening a new chapter in Moore's law. However, NTC in FinFET is largely unexplored. FinFET differs significantly from planar technology, with much improved channel characteristics, which have the potential to dramatically improve near-threshold performance.

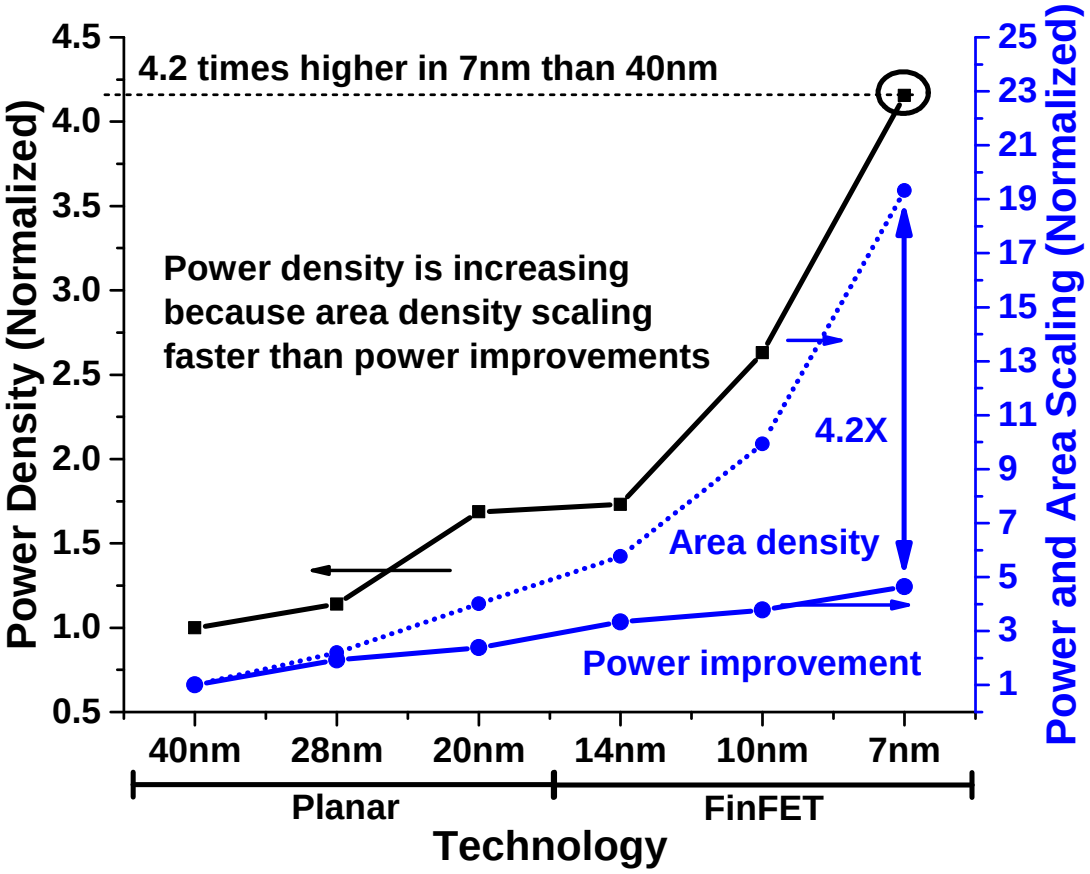


Figure 7.1: **Power and area scaling normalized to 40nm.**

As with the previous chapters, evaluation of near-threshold is considered for performance-sensitive workloads, specifically when latency is fixed to that of a single core at nominal voltage. Alternatively, nominal operation targets absolute single-thread performance (minimizing latency), and ultra-low, sub-threshold supply voltages target performance insensitive applications, such as low-power sensors. Initial results for predictive 7nm FinFET models show energy gain improvements over planar technology for performance insensitive workloads. However, the nominal supply-targeted 7nm device has poor energy gain when performance sensitive. Through near-threshold analysis we re-target the 7nm device for near-threshold, improving low-voltage energy consumption by 60%. Our results show greater than 8× energy-efficiency improvement for an NT-targeted device in FinFET, reversing trends seen in planar technology and even surpassing 180nm energy gains. When constraining the area of a system to that of a single core at 40nm, 2.5× throughput gain is possible for a fixed power budget by voltage scaling, as compared to nominal in 7nm FinFET, while in 20nm planar only 1.3× is possible through voltage scaling. When area is unconstrained, 8.2× is possible in 7nm FinFET and 3.3× in 20nm planar.

In Section 7.2, we begin by comparing a predicted 7nm FinFET device to trends seen in previous studies. In Section 7.3 we present an analytical model of near-threshold's energy gain and, by using this model in Section 7.4, identify the key device characteristics that are responsible for near-threshold performance in FinFETs. With this knowledge, in Section 7.5 we co-optimize a FinFET device targeted for NTC operation, which exhibits a significant reduction in energy over a standard low standby power (LSP) device for performance sensitive applications. We conclude the chapter with an extension of our previous studies [10, 11] by comparing NTC performance in three planar technologies and three FinFET technologies from 40nm to 7nm. Finally, in Section 7.6 we add effects of variation margining, area budget, and back-end-of-line parasitics when comparing across technologies—additional effects that were not previously well-modeled.

7.2 Near-Threshold in 7nm FinFET

7.2.1 Background

Voltage scaling can be viewed as a continuum of three operating scenarios from traditional nominal voltage operation to ultra-low voltage operation (Table 7.1). Nominal voltage operates a core at its peak clock frequency, therefore single-threaded performance is maximized. However, nominal voltage also consumes the most power and thus limits the system to the fewest number of cores that can be active within a thermal design power budget. Scaling down voltage to the ultra-low, sub-threshold region greatly reduces power demands, allowing for far more cores to operate within a power budget. However, voltage scaling also significantly degrades clock frequency, so ultra-low voltage is not suitable for workloads that are latency sensitive.

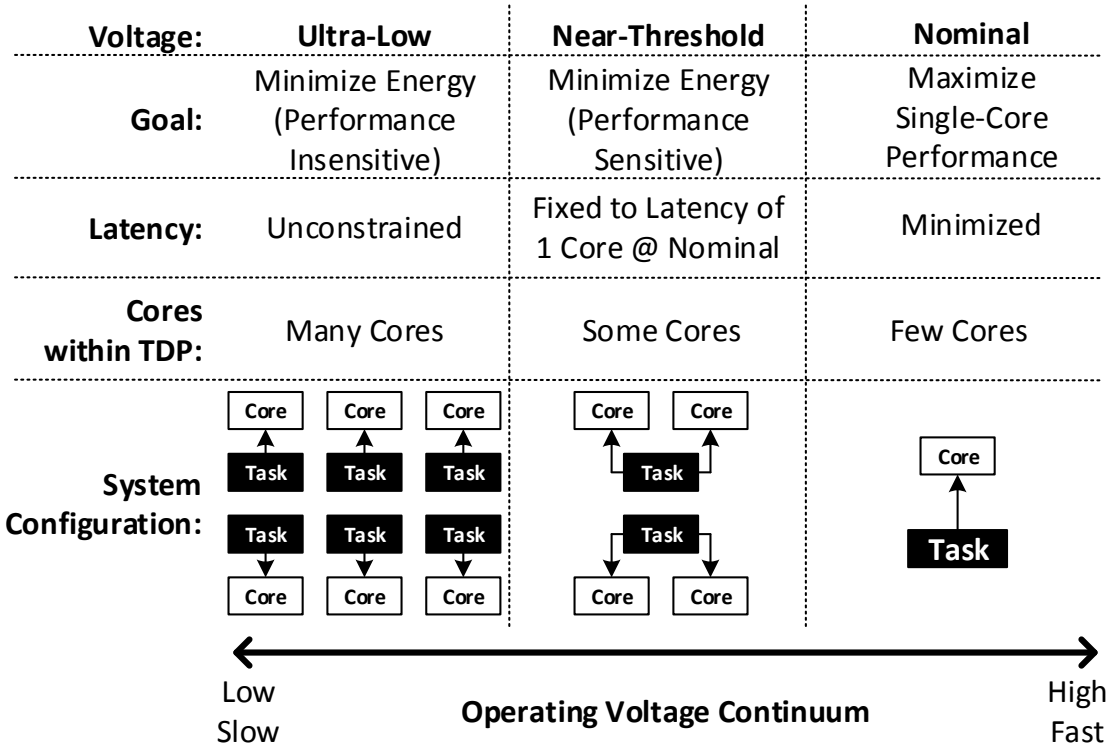


Table 7.1: **Voltage scaling operating scenarios, from ultra-low supply voltages to traditional nominal-voltage operation.**

Near-threshold (NT) balances ultra-low and nominal operating modes by parallelizing a task at low voltages to regain lost performance from clock frequency degradation [6]. In Pinckney et al. [10], we developed a systematic methodology for defining the near-threshold operating point and analyzed NTC energy and performance using transistor models of six industrial technology nodes from 180nm to 32nm. In order to consider performance sensitivity, latency is fixed to that of the task running on a single core at nominal voltage. As voltage is lowered to NT, clock frequency decreases and subsequently latency increases. However, this latency increase can be balanced through speeding up the task through parallelism (Table 7.1, middle). This is the definition of near-threshold we use in this work.

Achievable energy efficiency is limited by ease of parallelism and characteristics of the CMOS process [10]. The three key limiters to NTC scaling are: (1) *Leakage overhead*: as core voltage decreases, the obtainable clock frequency decreases. Consequently, leakage power is integrated over longer time periods for a given task, thus leakage energy dominates over dynamic energy at very low voltage. (2) *Amdahl overhead*: includes the algorithmic inefficiencies of parallelization on ideal hardware. (3) *Architectural overhead*: adds architectural-specific sources of inefficiency, such as non-ideal inter-core communications and memory hierarchy, including caches.

Pinckney et al. [10] also showed how the optimal NTC operating voltage (V_{opt}) tracks roughly $200 - 400$mV above threshold voltage for the benchmark suite. This constant supply voltage offset above V_t results in an energy gain that decreases progressively with each generation since supply voltage drops from technology to technology, reducing the proportional difference in voltage between near-threshold and nominal supplies. In 180nm the median energy gain from operating at NTC instead of nominal voltage was approximately $7.5\times$ across the benchmarks included, while maintaining throughput and latency. Scaling to newer technologies showed the downward trend in the benefits of operating at NTC, and in 32nm, the most recent technology node included in that study, median energy gain dropped to approximately $4\times$. Hence this study projected that NTC was becoming progressively less effective with process scaling.

Similarly, the number of cores needed to parallelize the application and maintain its latency in NTC decreased from 20 in 180nm to 12 in 32nm. In contrast, the number of avail-

able cores on a chip multiprocessor has increased from generation to generation, as transistor packing density has improved. Thus, processors are constructed with ever increasing number of cores while the useful amount of cores for energy-efficiency has decreased. If this trend continues, the realizable energy-efficiency gains will continue to decrease, and the problem of dark silicon will remain as many cores go unused, even with NTC.

However, at $22-16$nm and below, foundries are shifting to FinFET technologies, which differ from planar technologies by extending the transistor into three dimensions [62]. The transistor gate is controlled on three sides, instead of on one side as with planar technologies, leading to improved leakage, channel control, and packing density, among other advantages. Yet, differences between planar and FinFET within the NTC region have not been well investigated. The previous planar study [10] only included results from planar technologies as industrial FinFET models were unavailable to the authors at the time of publication. Without industrial models, it was unclear how near-threshold operation would affect FinFET, compared to planar. In this work, we investigate FinFET devices operating in the near-threshold regime, and our findings show that FinFETs exhibit much improved near-threshold performance and help mitigate dark silicon even in single-digit nanometer technology nodes. While FinFETs are praised for better channel control, drain-induced barrier lowering (DIBL), and subthreshold slope, the impact of these improved device characteristics on near-threshold operation has, to our knowledge, never been published.

7.2.2 Methodology

The methodology used in this work for evaluating near-threshold in FinFET is similar to the simulation framework proposed in the prior planar near-threshold studies [10, 11] to estimate energy gains and frequency loss. Our framework is split into two components: circuit characterization, to extract circuit delay and energy, and architectural models, to account for parallelism overheads. Predicting future technology nodes is difficult as many technological challenges have yet to be overcome. The International Technology Roadmap for Semiconductors (ITRS) provides estimates of many device parameters for high-performance, low-power, and SRAM transistors. However, ITRS reports are driven by future technology

requirements, and not necessarily representative of what is realizable. Therefore, ITRS tends to provide an optimistic outlook while industry estimates are more conservative [5].

Additionally, while circuit performance at nominal can be estimated with tabulated data from ITRS, voltage scalability cannot. Device models are essential to characterize performance while voltage scaling to near-threshold because ITRS does not include energy and delay curves as supply voltage is lowered. A well-known source of publicly available device models for multi-gate transistors is Arizona State University's Predictive Technology Models (PTM) [63], which are based on MOSFET scaling theory, ITRS, and other published data down to 7nm. However, these models being aligned with ITRS have very aggressive projections on dimension scaling and are not in line with industry trends [64–66]. For this work, ARM provided predictive technology models that include effects specific to FinFETs. The circuit simulations in this work use HSPICE BSIM Level 72 for 7nm, 10nm, and 14nm FinFET, and Level 54 for planar 20nm, 28nm, and 40nm models, of which all sets were developed by ARM based on published numbers, historical trends, and informed assumptions and calculations.

The canonical circuit simulated to characterize voltage scalability is a chain of thirty-one inverters, with a 15% activity factor, to emulate reasonably deep processor pipelines. Though actual critical paths are composed of more complex gates, we found inverters are reasonable for comparing performance and energy between operating voltages and technology. Circuit simulations also include back-end-of-line wire models and within-cell extracted parasitics, provided by ARM, where explicitly mentioned in later sections.

Prior work [10] architecturally evaluated near-threshold with the SPLASH-2 benchmark suite [67] using the gem5 cycle-accurate microarchitectural simulator [42]. SPLASH-2 consists of twelve scientific benchmarks intended to evaluate parallel systems, and was previously chosen because the benchmarks have been parallelized, so they readily scale to increased number of cores. Ease of parallelism varies by benchmark and, when fitted to Amdahl's equation [41], the Amdahl serial coefficient is up to 8%. For the purposes of this work, to illustrate parallelism sensitive voltage scaling, we picked an Amdahl serial coefficient of 2% which is higher than all but two of the SPLASH-2 benchmarks. For additional analysis of near-threshold on the serial coefficient, please refer to [10].

Custom MATLAB scripts combine circuit scaling to derive total energy from dynamic energy, static power, and parallelism overhead. The minimum energy point is found from the total energy, after overhead sources are included. Additionally, we expand on prior studies by constraining area in Section 7.5.2.

7.2.3 7nm FinFET Results

An initial 7nm FinFET predictive model was provided by ARM and run through the near-threshold characterization framework. The device's subthreshold leakage is approximately 0.6 nA/µm of channel width, which is essentially a low-standby power (LSP) device that foundries have shifted towards [68]. A comparison of the effectiveness of the 7nm device as compared to the previous planar nodes from [10] is shown in Figure 7.2, for performance-sensitive workloads (Amdahl serial coefficient $P_s = 2\%$) and performance-insensitive workloads. The results of the LSP 7nm device are initially unimpressive, with a 12% reduction in near-threshold energy-efficiency gain as compared to 32nm planar when performance sensitive. However, we observed a 130% increase in energy-efficiency gain for performance-insensitive workloads, deviating from the previous trend of effectiveness decreasing generation to generation.

Understanding why the 7nm FinFET device is better than recent planar nodes for performance-insensitive workloads is the primary motivation for this work, along with evaluating how to improve energy efficiency for performance-sensitive workloads. FinFET channel characteristics are much better than planar, namely FinFETs have much steeper subthreshold slope to allow lower V_t (thus lower V_{dd}) operation, better short-channel effects (DIBL), improved packing density, etc. However, how each of these characteristics impact near-threshold has not been explored, thus this work sets to analyze and understand how each individually effects voltage scalability. Using this understanding, we explain why FinFET is intrinsically better for near-threshold than planar, and how to modify the device for even better near-threshold performance, in the following sections.

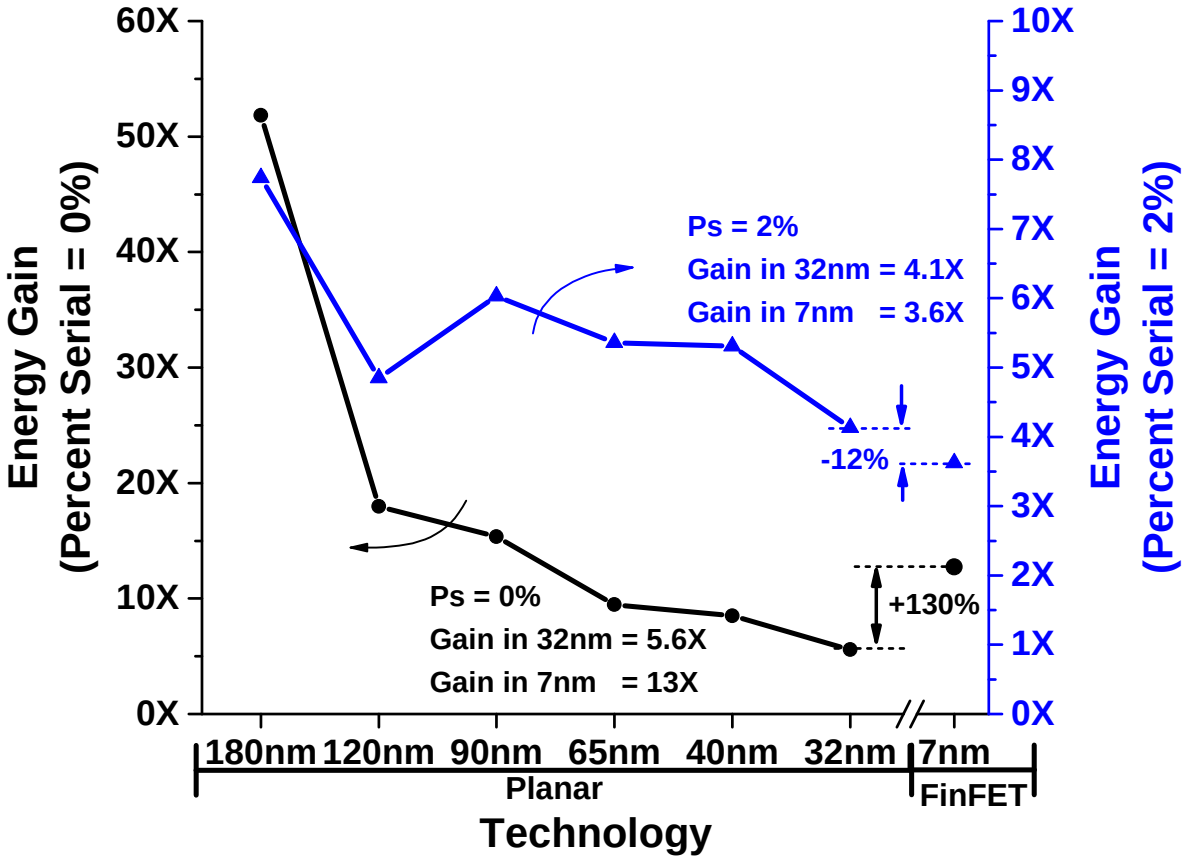


Figure 7.2: **Comparison of energy gain for performance sensitive and insensitive workloads, from previous planar study [10] to predicted 7nm FinFET device.**

7.3 Analytical Model

A simple analytical model is introduced to better understand underlying effects on device parameters. Though this model does not have high accuracy, especially for recent technology nodes, it is beneficial in understanding the effects of device parameters on NTC performance.

The energy of a task can be split up into two categories: dynamic and static, shown in Figure 7.3, and given by the equation:

$$E_{total} = E_{dynamic} + E_{static} \tag{7.1}$$

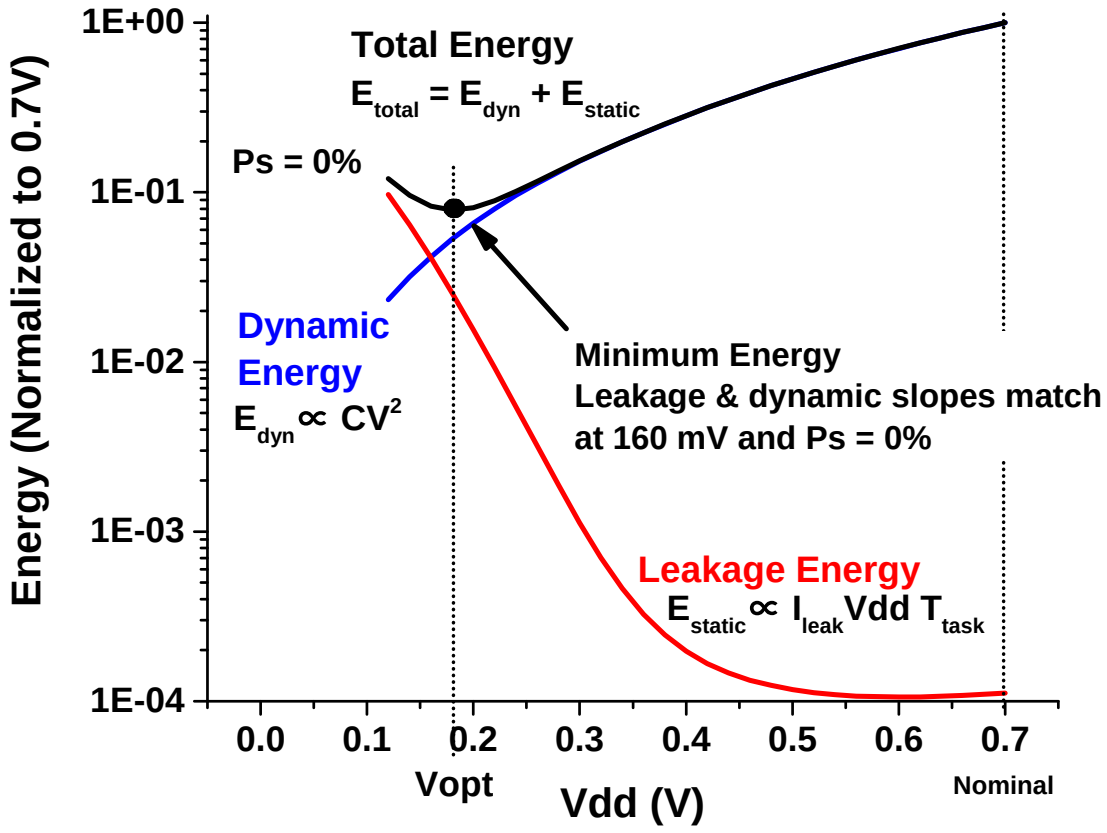


Figure 7.3: **Total energy of a circuit is composed of dynamic and leakage energies. Total energy is minimized when dynamic and leakage slopes match.**

Dynamic energy is the working energy needed to switch inputs of transistors and values of wires for calculations or communications during a task's execution. Dynamic energy can be modeled as a charge on a capacitor, and thus varies quadratically with supply voltage:

$$E_{dynamic} \propto C_{switch} V_{dd}^2 \tag{7.2}$$

Static energy is caused by leakages of a circuit regardless of if a task is executing or not. Static energy is usually dominated by subthreshold leakage through a transistor's source and drain and is dependent on the supply voltage and the period of time for a task to run.

$$E_{static} \propto I_{leak} V_{dd} T_{task} \tag{7.3}$$

The time for task completion depends inversely on the clock frequency of a core, which to first order, is inversely proportional to circuit delay (Figure 7.4):

$$T_{task} \propto 1/f \propto T_{logic\ delay} \tag{7.4}$$

Additional effects, such memory latency or peripherals, will change this relationship of completion time to clock frequency. While the behavior is not accurate for more complex systems, it is sufficient to understand voltage scaling behavior. Subthreshold leakage, when a transistor is in complete cutoff (gate-source voltage is 0V), can be modeled as [69]:

$$I_{leak} = I_{ds0} \exp \frac{-V_t}{nV_T} (1 - e^{-V_{dd}/V_T})$$

where I_{ds0} is the current at threshold V_t, n is a process-dependent constant, and V_T is the thermal voltage (kT/q, or approximately 26 mV at room temperature). Since supply voltage is typically much above V_T, even when near-threshold, we can simplify this equation as:

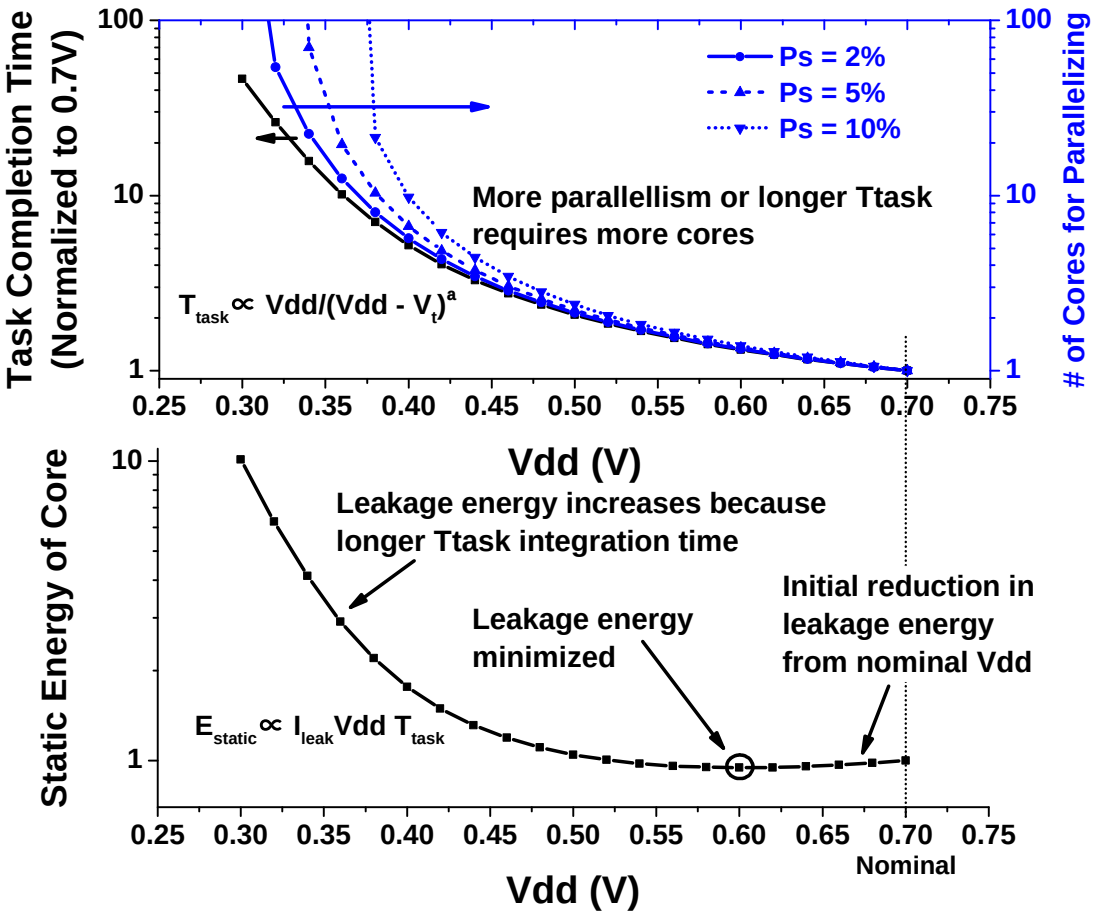


Figure 7.4: **Task completion time and leakage energy.** Task completion time increases as voltage is lowered, top. A larger serial coefficient requires more cores to maintain fixed latency. Leakage energy increases at low voltages since leakage power is integrated over longer periods of time (bottom).

$$I_{leak} = I_{ds0} \exp \frac{-V_t}{nV_T} \tag{7.5}$$

Thus to first-order, excluding effects such as drain-induced barrier lowering, leakage current is not a function of supply voltage, as long as supply voltage is reasonably high. Therefore, scaling of static energy with supply voltage is dependent on V_{dd} directly and the task completion time, but leakage current is fixed. As we discuss device characteristics in the next section, we will re-introduce drain-induced barrier lowering and other non-ideal I-V effects that impact static and dynamic energy consumption.

Logic delay is proportional to amount of charge needed to switch a transistor gate, $Q_{switch} = C_{switch}V_{dd}$, divided by the rate it can be discharged, which is modeled using the alpha power law of a transistor [29], $I_{dsat} \propto (V_{dd} - V_t)^{\alpha}$. Therefore, task completion time can be modeled as:

$$T_{task} \propto \frac{1}{f} \propto \frac{V_{dd}}{(V_{dd} - V_t)^{\alpha}} \tag{7.6}$$

From the above relationships, the dynamic energy monotonically decreases with supply voltage while the leakage energy initially decreases because I_{leak} and supply voltage drop. However, the task completion time rises exponentially at near-threshold voltages, and thus static energy increases as Vdd continues to be lowered. Energy is minimized when the margin cost of dynamic and static energy are in balance, in essence when the dynamic energy gain of scaling down voltage is equal to the marginal cost of static energy:

$$\frac{\delta E_{dynamic}}{\delta V_{dd}} = -\frac{\delta E_{static}}{\delta V_{dd}} \tag{7.7}$$

Marginal gain of dynamic energy is directly proportional to supply voltage through

$$\frac{\delta E_{dynamic}}{\delta V_{dd}} = 2C_{switch}V_{dd} \propto V_{dd} \tag{7.8}$$

while static energy consumption marginal cost is

$$\begin{aligned} -\frac{\delta E_{static}}{\delta V_{dd}} &= \frac{\delta}{\delta V_{dd}}[-V_{dd}T_{task}] \\ &= \frac{\delta}{\delta V_{dd}}[-V_{dd}\frac{V_{dd}}{(V_{dd}-V_t)^{\alpha}}] \\ &= \frac{V_{dd}}{(V_{dd}-V_t)^{\alpha+1}}(2V_t+(\alpha-2)V_{dd}) \end{aligned}$$

Assuming the use of an older technology, where α is close to 2, this simplifies to

$$-\frac{\delta E_{static}}{\delta V dd} \propto \frac{V_t V_{dd}}{(V_{dd}-V_t)^3} \tag{7.9}$$

At high voltages, $Vdd \gg V_t$, therefore Equation 7.9 further simplifies to $-\delta E_{static}/\delta V_{dd} \propto V_t/V_{dd}^2$. Dynamic energy dominates at nominal voltages, as long as switching activity factor is reasonably high. The slope of dynamic energy is proportional to V_{dd} (Equation 7.8) while for static energy it is very shallow because it is threshold voltage (a portion of V_{dd}) divided by the square of V_{dd}. As voltage is scaled to near-threshold, dynamic energy's slope becomes increasingly shallower while static energy initially increases its slope. Eventually static energy's slope becomes very steep near V_t because of the denominator $(V_{dd}-V_t)^3$, and subsequently the total energy rises causing a rapid loss of energy efficiency. When the slopes are of equal magnitude then total energy is minimized.

Up until now in this chapter we have been considering the energy of a task without regards to energy required to maintain task latency if it is performance sensitive. Parallelism overhead of a program can be modeled through Amdahl's law [41], where the speedup of a parallelized program is given by

$$Speedup = \frac{n}{1-P_s+P_s n} \tag{7.10}$$

where n is number of cores parallelized over and P_s is the percent serial coefficient of the workload. A perfectly parallelizable program has a $P_s = 0\%$, while higher percent serials indicate less of the code is parallelizable, up until $P_s = 100\%$ implying the workload is completely parallelizable. In this work we consider a fixed latency constraint when performance

sensitive, so that the speedup through parallelism has to balance any performance loss from longer circuit delay as a consequence of scaling to near-threshold:

$$Speedup = \frac{T_{task,NTC}}{T_{task,nominal}} \tag{7.11}$$

For a perfectly parallelizable task, the switching capacitance per core can be equally divided among the cores:

$$C_{switch,parallel} = \frac{1}{n} C_{switch,original}$$

As a concrete example of why capacitance per core for a task is not fixed, consider $E_{switch} = \alpha N C V_{dd}^2$ where N is the number of clock cycles to execute the task, C is the switching capacitance of a core, and α is the switching activity factory [31]. Consider that the task is now parallellized across two cores. Since it is perfectly parallelizable, the number of cycles to run the task is now $N/2$ while the amount of cores double so switching capacitance is now $2C$. The switching energy for the task is now

$$E_{switch} = \alpha(N/2)(2C)V_{dd}^2 = \alpha N C V_{dd}^2$$

Thus, the switching energy per task is constant and per core is now halved, since the task is running across two cores. In the case of a perfectly parallelizable task, $Speedup = n$. Therefore, in this case, the task dynamic energy is constant regardless of amount of parallelism:

$$\begin{aligned} E_{dynamic} &= nC_{switch,parallel}V_{dd}^2 \\ &= n\frac{1}{n}C_{switch,original}V_{dd}^2 \\ &= C_{switch,original}V_{dd}^2 \end{aligned} \tag{7.12}$$

For a task that does have parallelism overhead, and is not perfectly parallel, the energy is derated by a factor of $(Speedup/n)$ compared to the perfectly parallelizable baseline. To arrive at this derating factor, first assume that the switching capacitance can be divided

among cores as if it was perfectly parallelizable ($n = Speedup$):

$$C_{switch,parallel} = \frac{1}{Speedup} C_{switch,original} \tag{7.13}$$

Next, since the task is not perfectly parallelizable, but has parallelism overheads $P_s > 0$ and thus the number of cores is greater than the achieved speedup ($n > Speedup$), the dynamic energy is then:

$$\begin{aligned} E_{dynamic,parallel} &\propto nC_{switch,parallel}V_{dd}^2 \\ &= \frac{n}{Speedup} C_{switch,original} V_{dd}^2 \\ \implies E_{dynamic,parallel} &= \frac{n}{Speedup} E_{dynamic,original} \end{aligned} \tag{7.14}$$

By this definition, the ratio of $n/Speedup > 1$ provides a derating factor on dynamic energy. Because Amdahl's law shows an asymptotic speedup with number of cores, the ratio of cores to speedup is monotonically increasing as cores are added. Therefore, unlike a perfectly parallelizable workload, dynamic energy increases as voltage is lowered when parallelism overhead is included.

Leakage energy when parallelizing is similar and, for a performance insensitive workload, remains unchanged as cores are added. This may seem unintuitive at first, since as the number of cores increases the total subthreshold leakage also increases proportionally. However, the task completion time T_{task} decreases as cores are added. Therefore, for a perfectly parallelizable task, the amount of cores added and the reduction in task completion time cancel out, subsequently E_{static} is unchanged. When parallelism overhead is added, static energy increases as

$$\begin{aligned} E_{static,parallel} &\propto (nI_{leak,original})V_{dd}\frac{T_{task}}{Speedup} \\ &= \frac{n}{Speedup} I_{leak,original} V_{dd} T_{task,original} \\ \implies E_{static,parallel} &= \frac{n}{Speedup} E_{static,original} \end{aligned} \tag{7.15}$$

Again, an $n/Speedup$ factor can be used to derate leakage energy for tasks that are not perfectly pralellizable. To summarize, the total energy when parallelizing is

$$\begin{aligned} E_{total,parallel} &= E_{dynamic,parallel} + E_{static,parallel} \\ &\propto \frac{n}{Speedup} E_{total,original} \end{aligned} \tag{7.16}$$

which for a perfectly parallelizable workload ($Speedup = n$) is identical to the performance insensitive result, otherwise the energy is derated by the ratio of cores to speedup, which increases as supply voltage drops.

Figure 7.5 demonstrates increasing parallelism overhead increasing the minimum energy and decreasing voltage scaling's efficacy. With a perfectly parallelizable program ($P_s = 0\%$), the minimum energy is 8% of the energy at nominal. A serial coefficient of $P_s = 2\%$ raises minimum energy to 28% of the energy of nominal for this example technology.

Since our analysis minimizes energy subject to a latency constraint, the latency of a task is fixed when voltage scaling, but the absolute throughput at nominal will change with varying device characteristics. For example, if threshold voltage is reduced, the fanout-of-4 circuit delay at nominal will improve. In order to evaluate a device characteristic's impact on absolute system performance, we use an aggregate throughput metric. In our analysis, we assume a task repeats as soon as it finishes, thus the throughput is the inverse of task latency. Furthermore, the power budget of the representative system is fixed so additional tasks are run until the power constraint is met. If a task consumes $1/X$ of the fixed power budget, its throughput is multiplied by X to be the aggregate throughput of the system, and our goal is to maximize aggregate throughput. Figure 7.5 illustrates aggregate throughput, normalized to nominal voltage, as compared to total energy.

7.4 Device Characteristics

Transistor devices have a multitude of interrelated characteristics, but we focus on a few key parameters relevant to FinFET. Using the analytical model from the previous section, we first examine the effects of three basic device characteristics impacting near-threshold

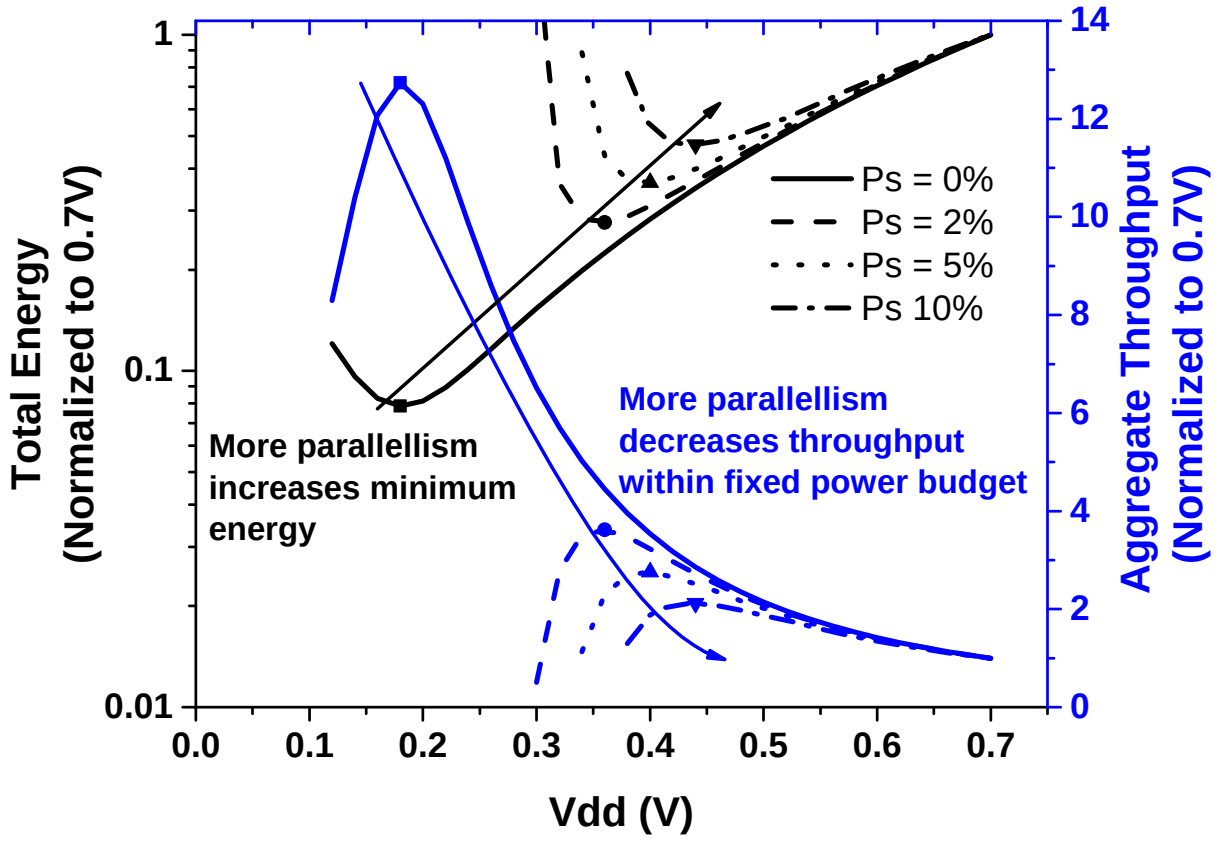


Figure 7.5: **Total energy of a task increases with a higher serial coefficient (P_s) since parallelism overheads limit voltage scalability as task latency is fixed.** Aggregate throughput for a fixed power budget is improved as energy per task is reduced.

performance: drain-induced barrier lowering, subthreshold slope, work function (effectively threshold voltage). Then, we expand this analysis into five more device or process characteristics: gate capacitance, source/drain parasitic resistance, within-cell and back-end-of-line parasitics, FinFET fin height, and lastly planar channel length.

7.4.1 Basic Device Characteristics

Work function

Work function changes the transistor's threshold voltage V_t, with lower threshold voltages exhibiting increased leakage. If leakage is a significant portion of the total energy, a lower threshold voltage negatively impacts voltage scalability since static energy is more significant and therefore V_{opt} is higher. However, threshold voltage has a significant impact on clock frequency scaling through changing transistor ON current, $T_{task,NTC} \propto 1/(V_{dd} - V_t)^{\alpha}$. Figure 7.6, top, shows normalized FO4 circuit delay (i.e. $T_{task,NTC}$) for five transistor threshold voltages. The 0.6 nA/µm leakage device is the low-standby power device (LSP) presented in Section 7.2, and exhibits the worst circuit delay voltage scalability. For instance, at $V_{dd} = 360$ mV the FO4 delay of the LSP device is 10× higher than at the nominal voltage of $V_{dd} = 700$ mV. The subsequent higher leakage devices (6, 40, 350, and 2800 $mathrmnA/$µm) have lower threshold voltages, therefore can scale lower in supply voltage for the same degradation in FO4 delay.

Better FO4 delay scalability allows a task to operate at a lower voltage and still maintain adequate performance, as less parallelism is needed for a fixed latency constraint. The aggregate task throughput within a fixed power budget for the five transistor voltages is shown in Figure 7.6, bottom. The LSP device has peak throughput at 360 mV, as below this voltage parallelism overhead becomes significant. The 6 and 40 nA/µm scale lower and have better throughput within the power constraint, since they are able to operate at a lower voltage with better FO4 delay degradation. Though the circuit delay scalability of the 350 and 2800 nA/µm devices are the best, leakage is very significant in these devices so that throughput decreases, even though V_{opt} is at very low voltages. The high leakage of these devices limits the energy efficiency gain for scaling from nominal supply voltages to

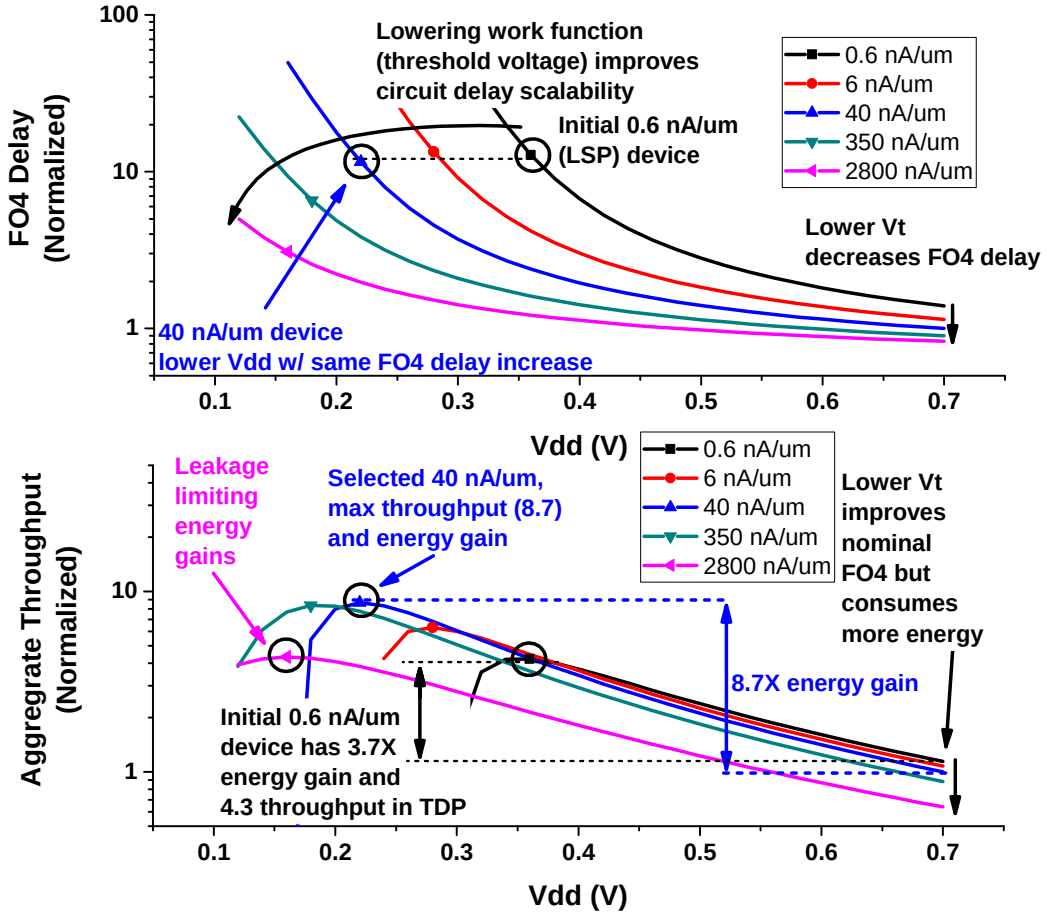


Figure 7.6: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) for varying threshold voltage in 7nm FinFET.**

near-threshold, since dynamic energy savings comes at the cost of higher static energy, even despite the devices being faster at nominal V_{dd}.

For the 2% serial coefficient workload, the 40 nA/µm device achieved the best throughput, with an energy gain of 8.7× at a $V_{opt} = 220$mV, decidedly better than the 3.6× gain of the original LSP device. Devices with around 50 nA/µm are colloquially referred to as *high-performance* (HP) transistors within the semiconductor industry. Analysis in the subsequent sections uses the HP device as a baseline in which to compare.

Drain-induced barrier lowering

Drain-induced barrier lowering (DIBL) is a short-channel effect that reduces threshold voltage as V_{ds} increases. This can be modeled through [69]:

$$V_t = V_{t0} - \eta V_{ds} = V_{t0} - \eta V_{dd}$$

where V_{t0} is the threshold voltage with no drain-source potential and η is the DIBL coefficient (typically around 100 mV/V [69]). As V_{dd} is lowered, DIBL causes V_t to increase and therefore the transistor overdrive voltage, $V_{ov} = V_{dd} - V_t$, rapidly collapses and severely limits voltage scalability. This directly affects task completion time $T_{task,NTC} \propto 1/(V_{dd} - V_t)^{\alpha} = 1/V_{ov}^{\alpha}$ and therefore the *Speedup* needed to maintain a latency constraint. As the DIBL coefficient η increases, $T_{task,NTC}$ degrades, shown in Figure 7.7. DIBL 1 and DIBL 2 are progressively worse DIBL coefficients from the baseline while each device is tuned to match both I_{off} and the ON current at nominal supply voltage of the baseline device.

For workloads that are sensitive to performance, the poor voltage scalability in clock frequency translates to limited energy gains and an increasing V_{opt} as more parallelism is required (Figure 7.7, bottom). Therefore, an improved DIBL coefficient directly improves near-threshold energy gains and performance in near-threshold for performance constrained applications.

For performance insensitive workloads, energy and V_{opt} do not change significantly, and, in fact, leakage can be slightly reduced at low voltages because of increased threshold voltages, $I_{leak} \propto \exp((-V_{th0} + \eta V_{dd})/(nV_t))$.

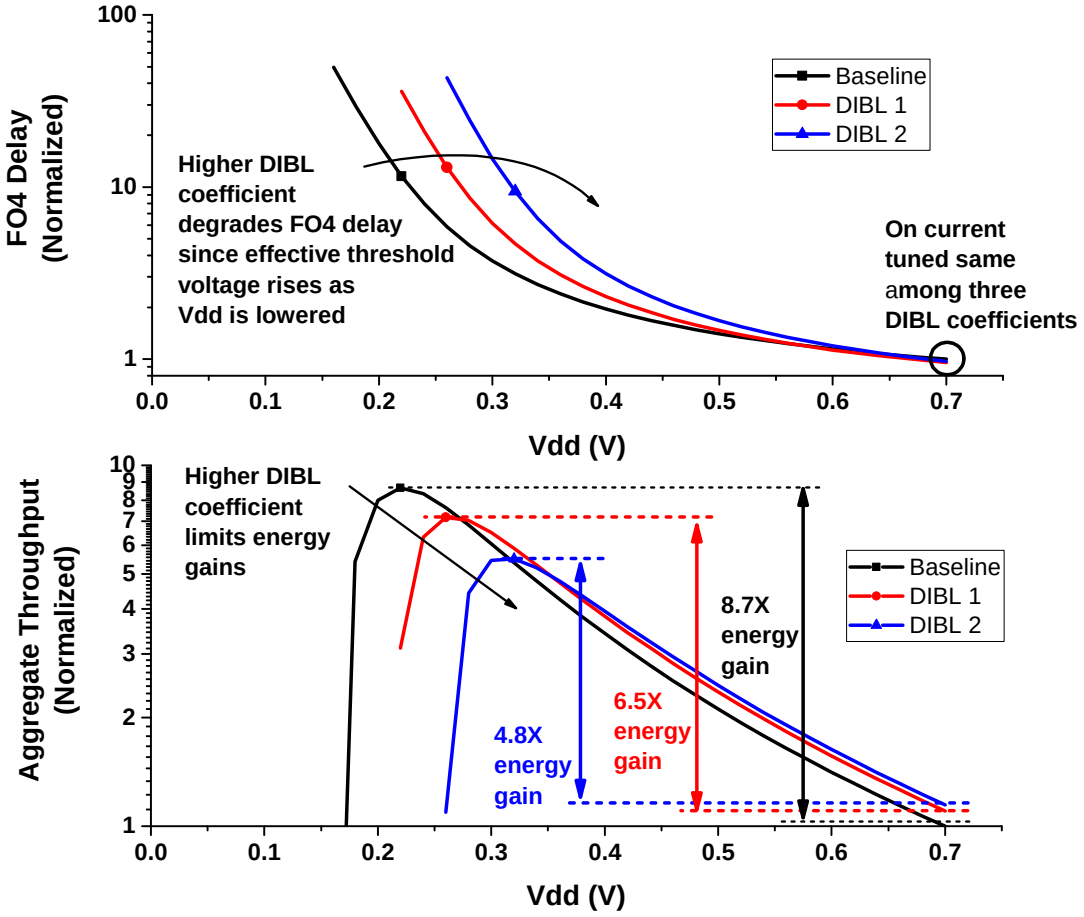


Figure 7.7: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) as DIBL coefficient increases in 7nm FinFET.**

Subthreshold slope

Subthreshold slope is the change of magnitude of drain-source current for a corresponding change in threshold, which can be modeled as [69]:

$$I_{leak} = I_{ds0} \exp(\frac{-V_t}{nV_T})$$

The denominator nV_T sets the subthreshold slope of the device. Subthreshold slope is typically given in units of mV/dec, and a steeper slope (smaller mV/dec) allows for a smaller threshold voltage to achieve the same leakage, as less mVs are required to decimate source-drain leakage currents. Two things happen as subthreshold slope increases (becomes less steep): (1) for the same threshold voltage, leakage increases; and, (2) the current drivability of the transistor improves (i.e. the transistor is better able to drive a load at lower voltages). The increased effective current improves $T_{task,NTC}$ scaling with V_{dd}, as shown in Figure 7.8, top, with worsening subthreshold slope.

Despite higher leakage with worse subthreshold slope, causing total energy at near-threshold to increase, circuit delay scaling improves because of better drivability. These two effects (higher leakage and better drivability) oppose each other for performance sensitive workloads, thus V_{opt} stays relatively constant (Figure 7.8, bottom). However, for performance insensitive workloads, improved circuit delay scaling has no impact on energy and thus increases both V_{opt} and total energy, limiting achievable energy efficiency gains.

7.4.2 Additional Process Characteristics

The previous process characteristics are the main contributors to improvements of near-threshold in FinFET. However, additional process effects may impact voltage scalability, which are presented below.

External resistance and Gate Capacitance

Two significant delay sources in transistors are the parasitic source drain resistance and gate capacitance. Parasitic source drain resistance, primarily contact resistance (R_{cont}), contributes to poorer drivability of a transistor. This decreases FO4 delay of transistors, at

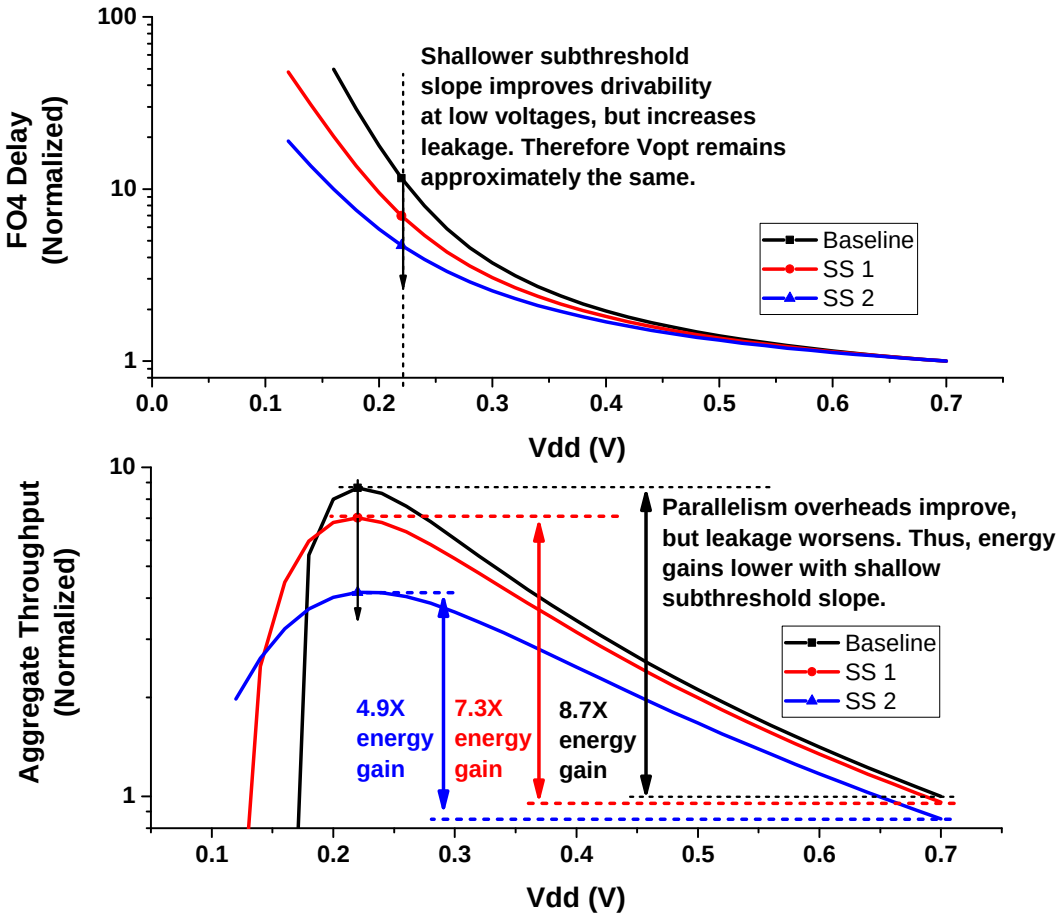


Figure 7.8: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) as subthreshold slope becomes less steep in 7nm FinFET.**

both nominal and near-threshold, but does significantly change the circuit delay scalability of the device, shown in Figure 7.9 with baseline HP device and two improving R_{cont} parameters.

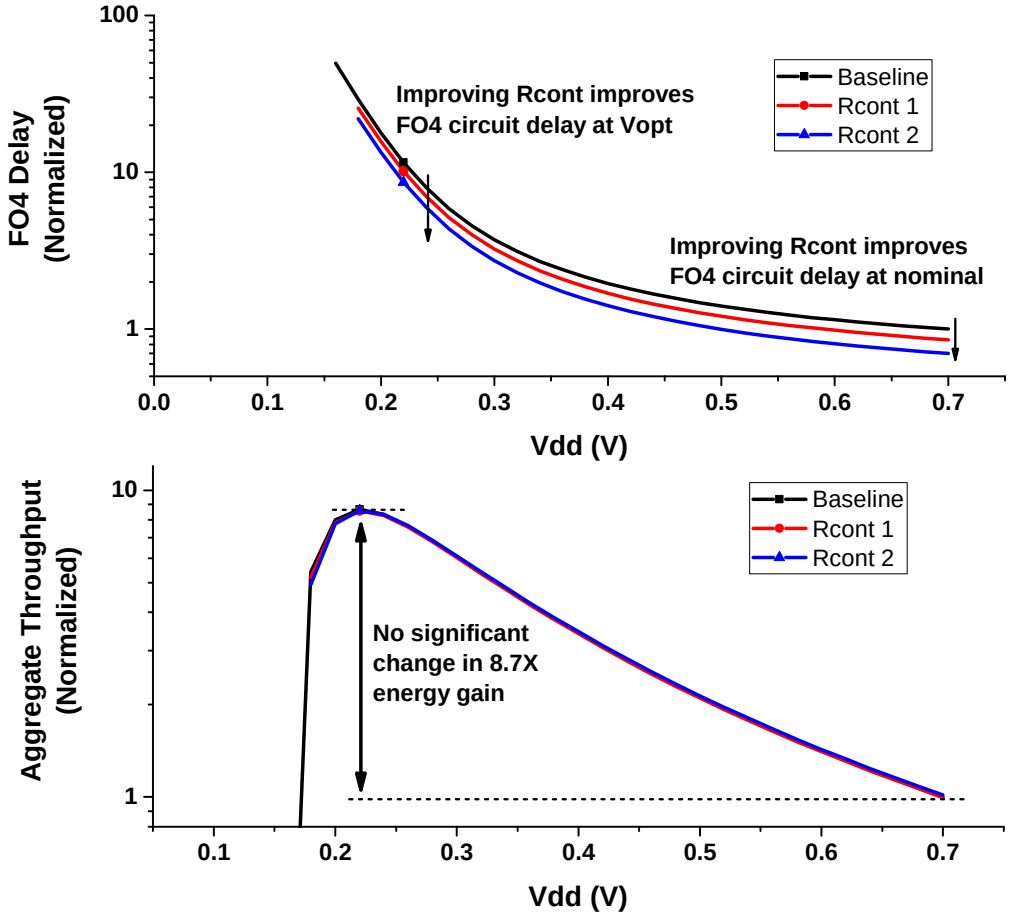


Figure 7.9: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) with parasitic source/drain resistance decreasing in 7nm FinFET.**

Decreasing gate capacitance C_{gg} improves circuit delay and decreases dynamic energy consumption, as less total capacitance is switching every cycle. Figure 7.10 shows the HP baseline device compared to a device with smaller gate capacitance. The FO4 circuit delay improves, as is expected from less gate capacitance but similar drivability (i.e. lower C and constant R). We also observe energy gain improving from 8.6× in the baseline to 9.3× in the lower gate capacitance device because of slightly better delay scaling.

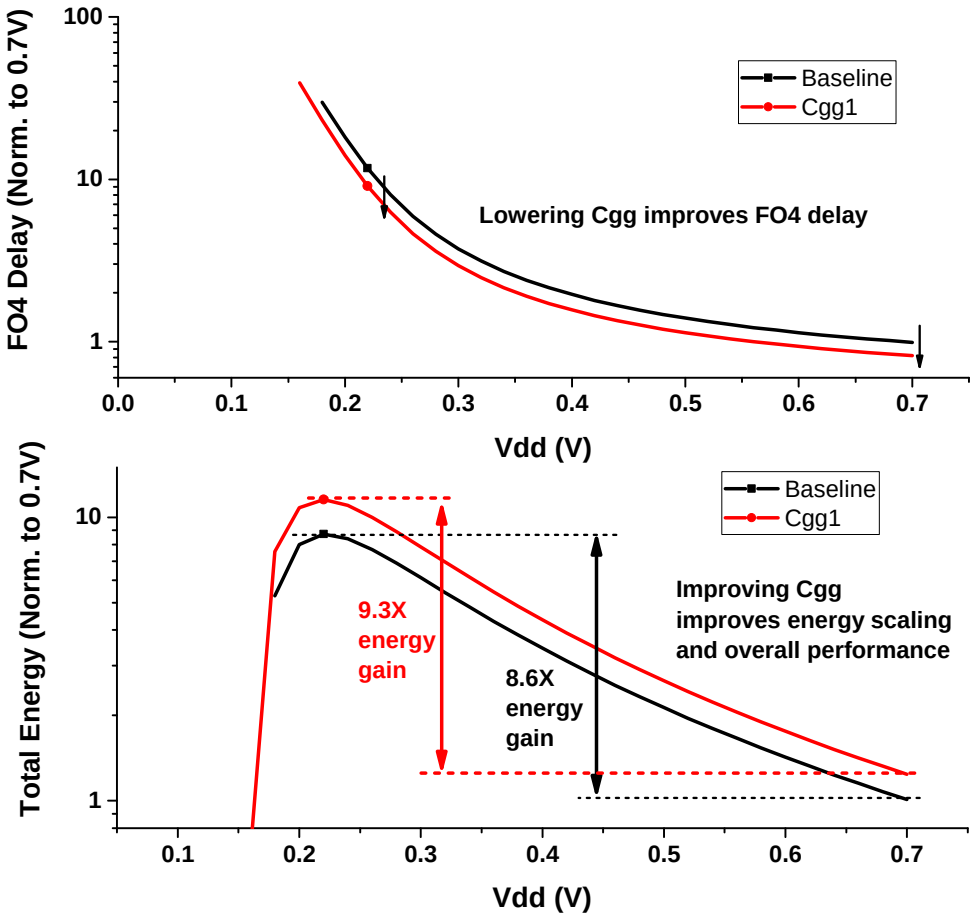


Figure 7.10: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) with gate capacitance decreasing from baseline in 7nm FinFET.**

Back-End-of-Line and Within-Cell Parasitics

Transistors are interconnected through wires and vias, thus back-end-of-line parasitic capacitance and resistance needs to be considered when analyzing voltage scaling performance. Within-cell parasitics are added to the characterization simulations in this study by extracting representative standard cell layouts of a 1X inverter in predictive 7nm, 10nm, and 14nm FinFET technologies, as provided by ARM. These parasitic models include source, drain and gate resistance due to trench contacts and local interconnects introduced at sub 20nm nodes and the corresponding coupling capacitances between input and output pins and the pins to power-rails. It has been observed that within cell parasitics can contribute up to half of the total gate delay.

Wire parasitics are modeled in our HSPICE simulations through π-models [70] of predicted resistance and capacitance per unit length of a low-level metal wire with minimum width and spacing. The wire length was swept across multiples of minimum track pitch, from 150 tracks (labeled 150TR) to 1200 tracks (labeled 1200TR), shown in Figure 7.11. Fanout-of-4 circuit delay was measured, and though the absolute delay increases as within-cell and wire load is added, energy-efficiency gain is nearly identical across the different wire lengths. Unlike the prior device characteristics, where aggregate was normalized to a baseline at 0.7V, the BEOL sweeps are normalized to itself at 0.7V for each wire length, as longer wire lengths will always be slower than shorter wire lengths.

Since throughput and V_{opt} are nearly the same across varying wire lengths, BEOL parasiticts do not significantly impact voltage scaling analysis in 7nm FinFET. In Section 7.5.2 we revisit BEOL across technologies and find that BEOL lowers energy gains slightly for some technologies.

FinFET fin height

As fins are discretized, transistor width cannot be as accurately selected as with planar. Therefore, careful selection of fin height is crucial for near-threshold operations. The 7nm FinFET device was simulated with fin heights of 15nm, 26nm, 30nm, 34nm, 38nm, 42nm, and 70nm. Our baseline device in the prior sections had a fin height of 42nm as that is

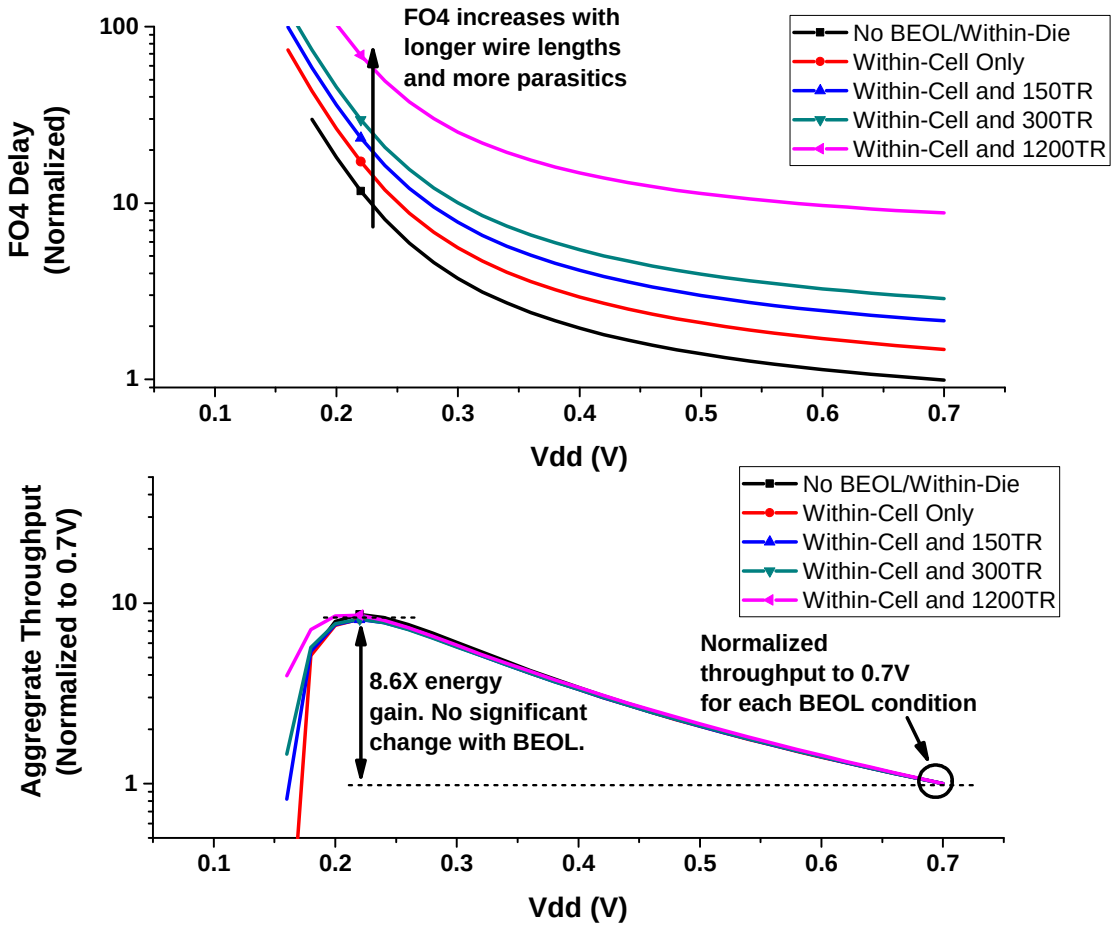


Figure 7.11: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) for within-cell parasitics and back-end-of-line parasitics from varying wire lengths in 7nm FinFET.** Throughput is normalized for each BEOL to 0.7V.

closest to industry projects. Substhreshold leakage was kept constant among all fin heights by modifying the work function. Because parasitics add a fixed capacitive load which can change the optimum fin height, we simulated three scenarios: without within-cell or wire parasitics, within-cell parasitics only, and within-cell plus a 300 track wire. The 300TR wire was selected as it is close to the average wire length observed for a representative placed-and-routed ARM processor at sub-28nm technologies. Fanout-of-4 circuit delay and NT energy-efficiency gain for these sweeps is shown in Figure 7.12.

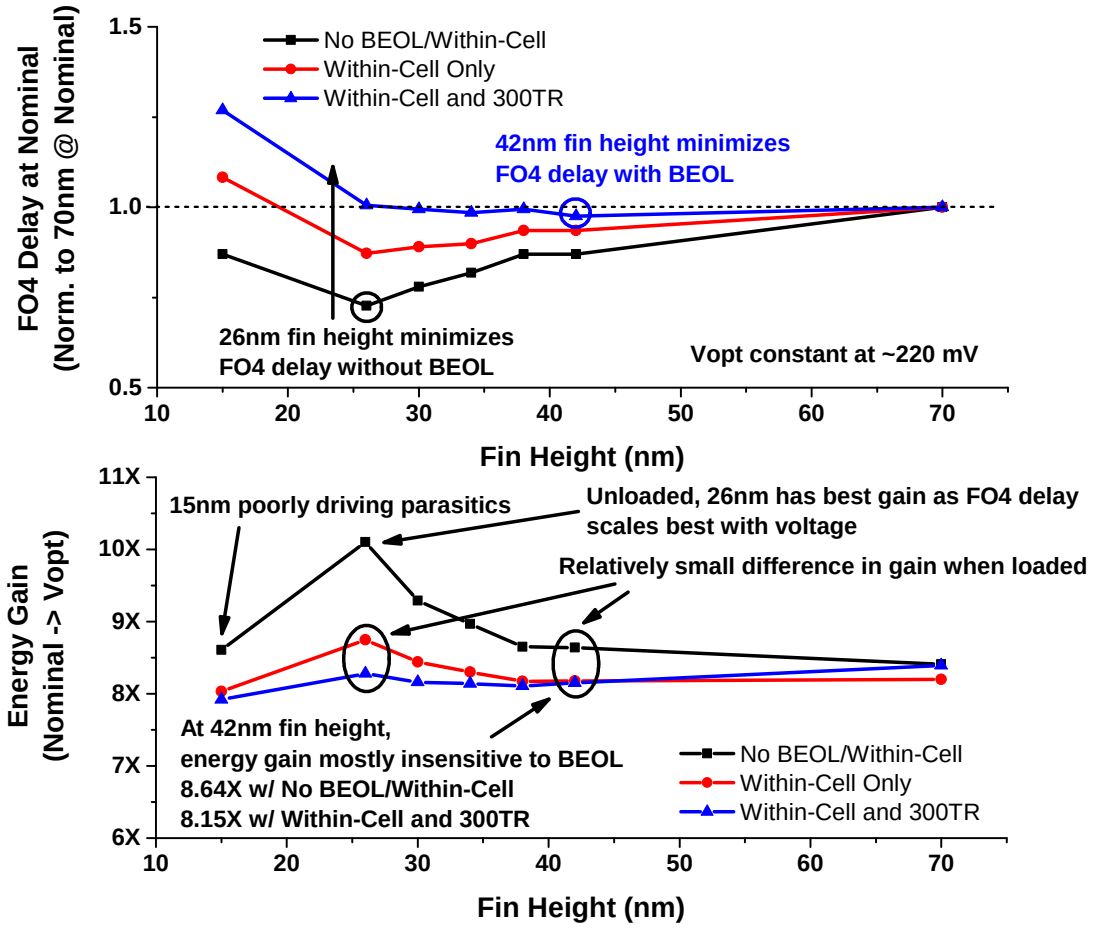


Figure 7.12: **Circuit delay scaling (top) and energy efficiency gain (bottom) for varying fin heights and different wire lengths in 7nm FinFET.**

Without considering within-cell or BEOL parasitics, decreasing fin height reduces capacitance and drive strength simultaneously as the physical fin dimensions become smaller. However, drive strength decreases at a slower rate than gate capacitance, therefore FO4 delay improves and the fin height for minimizing FO4 delay is 26nm, Figure 7.12, top. However,

BEOL and within-cell parasitics add a fixed load that remains unchanged with fin height. When including within-cell parasitics and the 300TR length wire, the 42nm has marginally improved FO4 delay more than the other fin heights. Of course, this also comes at the cost of increased dynamic energy since more gate capacitance is switching every cycle.

The optimal near-threshold voltage V_{opt} was relatively constant between $200 - 220$ mV for all the combinations studied. Subthreshold leakage is the same across all devices, so larger fin heights have a slightly bigger ratio of dynamic to static energy, and therefore favor slightly lower voltages as static energy is marginally less significant. However, energy-efficiency gain deviated with back-end-of-line parasitics, especially at with 26nm fin height. When unloaded, with no within-cell or wire parasitics, 26nm fin height had a peak energy gain of 10.1× compared to 8.7× of the baseline 42nm device because of improved circuit delay scalability. However, when loaded, this effect is diminished, and 26nm fin heights have a gain between $8.2 - 8.8\times$ depending on if wire load is included or not, respectively.

Depending on expected wire loads, a fin height smaller than the 42nm height baseline (e.g. 26nm fin height) is preferable since fanout-of-4 is faster for smaller wire loads and dynamic energy is reduced. However, for our analysis we kept with the 42nm fin height as it was the most representative of expected fin heights in 7nm FinFET. When including parasitics, a smaller fin height does not greatly change voltage scalability analysis but would result in less overall energy than the 42nm height we used.

Planar channel length

A common technique to reduce leakage and short-channel effects in planar is to increase channel length, l_g, to be longer than the minimum length supported by the technology, at the cost of poorer drivability and gate capacitance. We swept channel length from 40 nm to 100 nm when simulating our 40nm planar CMOS models. Fanout-of-4 delay and aggregate throughput for a fixed TDP across this sweep is plotted in Figure 7.13.

Increasing channel length to $l_g = 100$ nm slows FO4 delay by 3.2×. A slower FO4 penalizes aggregate throughput since tasks run slower at nominal supply voltage. However, since the clock frequency is slower, more tasks can fit within the same power budget, as dynamic power $P \propto CV_{dd}^2 f$. For channel lengths of 50 and 60 nm the aggregate throughput

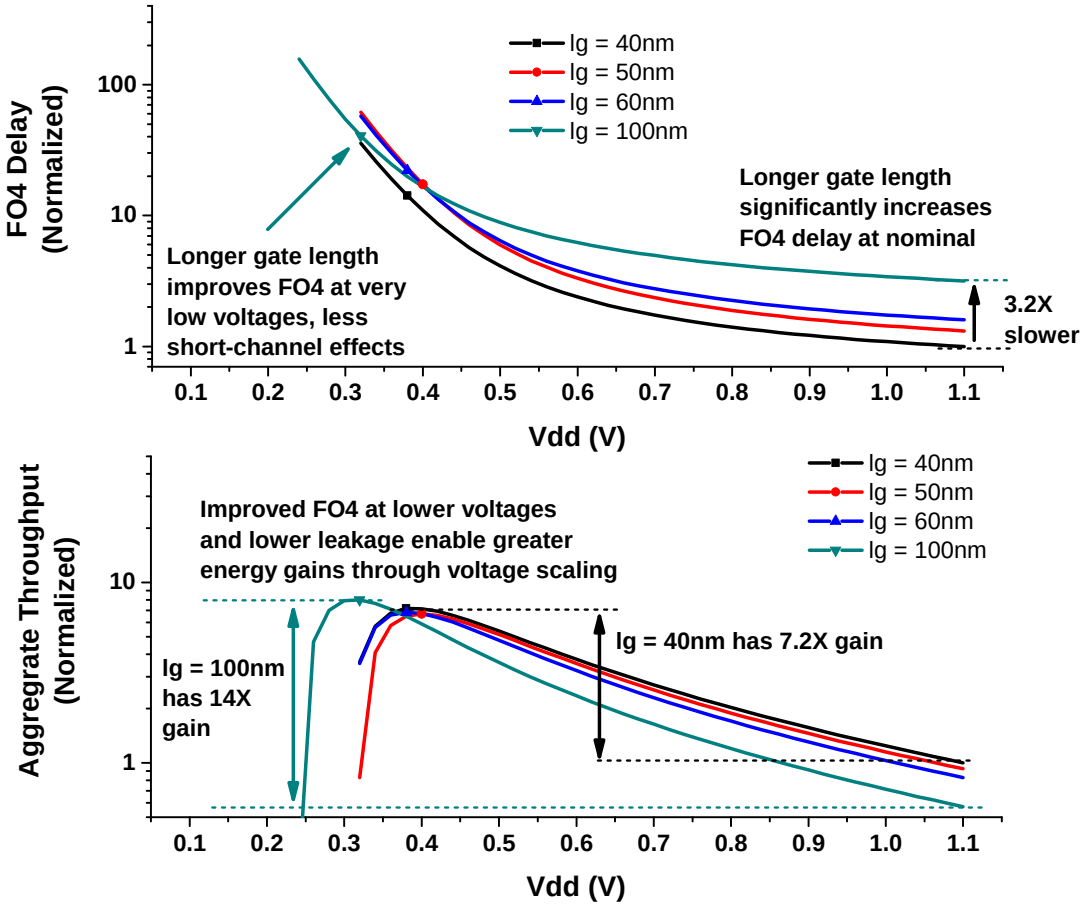


Figure 7.13: **Circuit delay scaling (top) and aggregate throughput for a fixed TDP (bottom) with varying channel lengths in a 40nm planar technology.**

is roughly the same as with a gate length of 40nm. However, $l_g = 100$ nm can scale to lower voltages while still improving energy, as leakage and short-channel effects (thus parallelism overhead) are mitigated. Peak throughput of a $l_g = 100$ nm channel length is 11% higher than $l_g = 40$ nm channel length, at the cost of 43% slower performance at nominal supply voltage. Note that the energy gain difference is much higher, 14× for $l_g = 100$ nm versus 7.2× for $l_g = 40$ nm, but the slower fanout-of-4 delay offsets the relative throughput gains from improved voltage scaling.

7.5 Technology Trends

The previous sections examined the individual device characteristics that impact voltage scaling and near-threshold energy-efficient operation. In this section we revisit the 7nm FinFET and expand the analysis to older FinFET and planar technology nodes.

7.5.1 Co-Optimized 7nm FinFET Device

The 7nm FinFET device from Section 7.2 was re-targeted for improved near-threshold operation by lowering threshold voltage to 40 nA/µm as found in Section 7.4.1. The device is essentially a low-V_t (LVT) or high performance (HP) transistor flavor. Subthreshold slope, DIBL, fin height, parasitic source drain resistance, and gate capacitance were targeted to match predictions for a LVT/HP FinFET device in 7nm. Figure 7.14 shows a comparison of this device to the original low-standby power (LSP) device from Section 7.2. The NT-targeted device consumes 60% less energy in near-threshold, because of improved circuit delay scalability, at the cost of 14% higher energy at nominal supply voltage. Since this device has a lower threshold voltage than the original, fanout-of-4 delay is improved for both nominal and near-threshold operation.

The NT-targeted 7nm device, along with two older FinFET technologies (10nm and 14nm), is compared with planar technology nodes (20nm, 28nm, 40nm) below.

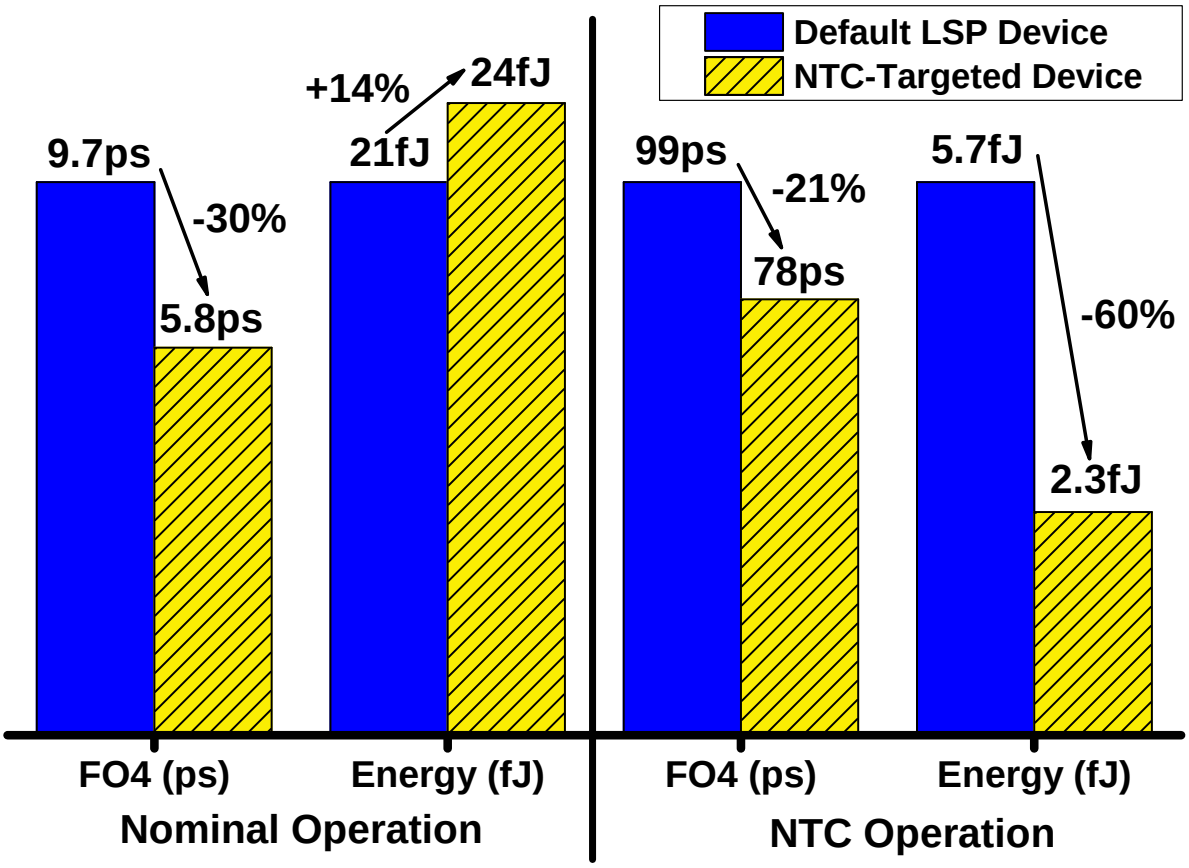


Figure 7.14: **Comparison of 7nm FinFET predicted device before and after NTC optimizations.**

7.5.2 Comparison to Planar

All six technologies provided by ARM were re-targeted for consistency to that of the HP/LVT (40 nA/µm) device from the prior subsection. We then compared the 7nm FinFET to five older technologies, 10nm and 14nm FinFET, and 20nm, 28nm, and 40nm planar. By targeting all technologies for the same transistor threshold, better consistency is obtained compared to [10] which used disparate technology models from different foundries. Additionally, the analysis in this work extends the 180nm to 32nm analysis in [10] by analyzing 40nm and below.

The energy gain across the six technology nodes is shown in Figure 7.15 both without additional (within-cell or wire) parasitics and with within-cell and 300TR wire parasitics. Within-cell extracted parasitics are not included since they were unavailable across all technology nodes. However, based on findings from Section 7.4.2 the relative voltage scaling gains should not be impacted greatly from within-cell parasitics. Of the planar nodes, 40nm has the best energy gain at $6.2-7.2\times$ and this reduces in 28nm and 20nm to $3.5\times$ and $3.3\times$, respectively, confirming the trends seen in [10]. Energy gain is diminished in these newer nodes because of stagnated V_t but lower V_{dd} (thus a reduction in headroom) and increased short-channel effects, such as DIBL, causing poor circuit delay scaling.

Transitioning to FinFET in 14nm shows much better energy gains of $9.3-10.7\times$ because threshold voltage has dropped by approximately 210 mV, with the same leakage characteristics, and DIBL coefficient has improved from 173 mV/V in 20nm to 31 mV/V in 14nm. In successive FinFET technologies the energy gain decreases, not because of worse DIBL but because the nominal voltage is dropping by 50 mV per generation while threshold voltage is mostly constant (deviating 3 mV per generation). Thus, the dynamic range between nominal supply voltages and near-threshold is reduced with each generation.

The optimal near-threshold supply voltage to maximize energy gain, V_{opt}, is shown across the six technology nodes in Figure 7.16, both with and without BEOL parasitics. The nominal supply voltage and threshold voltage for each technology is also shown in the figure. For the 2% serial coefficient studied, V_{opt} is approximately $40-80$ mV above the threshold

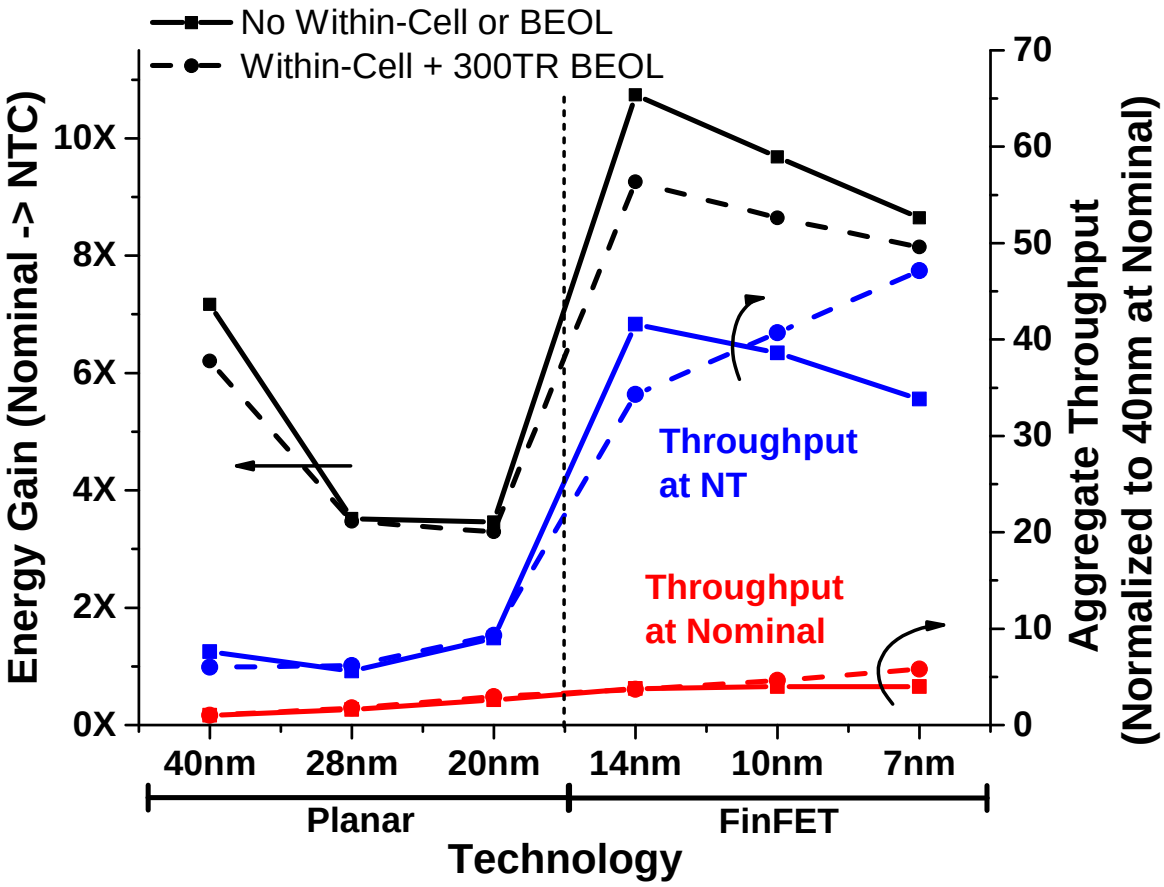


Figure 7.15: **Energy gain and throughput across technology.**

voltage. Workloads with higher serial coefficients would have higher V_{opt} because of increased parallelization overheads. BEOL wire loads do not significantly change V_{opt}.

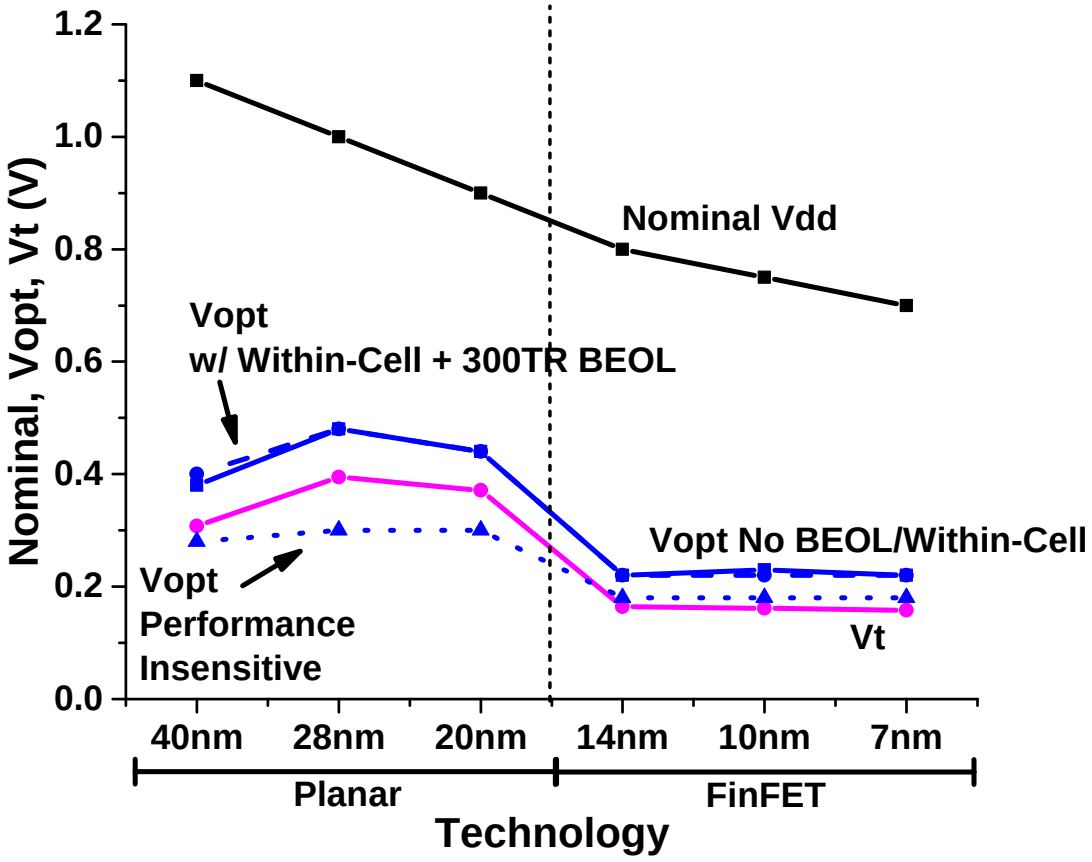


Figure 7.16: **Vopt across technology.**

7.6 Additional Observations

Beyond the initial technology comparison in the last section, we look at additional across technology effects, including variability and area constraints. These effects add additional constraints and change the achievable energy-efficiency gain.

7.6.1 Variability

Global (die-to-die and wafer-to-wafer) variation and local (across-die or within-die) variation both impact voltage scaling analysis. In near-threshold, circuit delay is more sensitive to changes in threshold voltage than at nominal voltage, and thus delay variation is exacerbated at near-threshold. This must be accounted for when evaluating near-threshold for energy efficiency, since operating at a higher voltage may be worse for energy with ideal transistors, yet more practical as it increases yield due to reduced variation.

Global Variation

Global variation shifts the performance or energy of all transistors on a chip, and is traditionally evaluated through typical (TT), fast (FF), and slow (SS) fixed corners. The typical corner models an average transistor in a process, while slow and fast model 3-sigma extremes in performance. For the six technology nodes in this chapter, we simulated the FF and SS corners in addition to the typical (TT) corners used in previous sections. Global variation is correlated, so, for example, a 10% increase in delay of a single inverter equally increases the delay for a chain of inverters by 10% (assuming rising and falling transitions are impacted equally).

Binning is a common technique used by processor manufactures to designate different models of a product based on tested performance. In essence, a slow chip is sold for a different price than a fast chip, even though the designs are identical. The majority of chip designs cannot be binned, and instead must meet a minimum performance specification. Fast transistors correspond with a lower threshold voltage, so leakage is higher at the fast process corner. When binning is not available, a chip with fast transistors is still run at the slow clock speeds, thus static energy increases. We evaluated near-threshold for the two cases, when binning is or is not available.

Binning allows for fast transistors to operate at a different clock speed than slow transistors. In this case, near-threshold operation has a normalizing effect on maximum throughput, Figure 7.17. Since the FF corner's threshold voltage is lower than the SS and TT corners, it allows for a slightly better circuit delay scaling and lower V_{opt}. In earlier planar nodes, the

difference in energy gain and V_{opt} between the fixed corners is more pronounced than in the FinFET nodes.

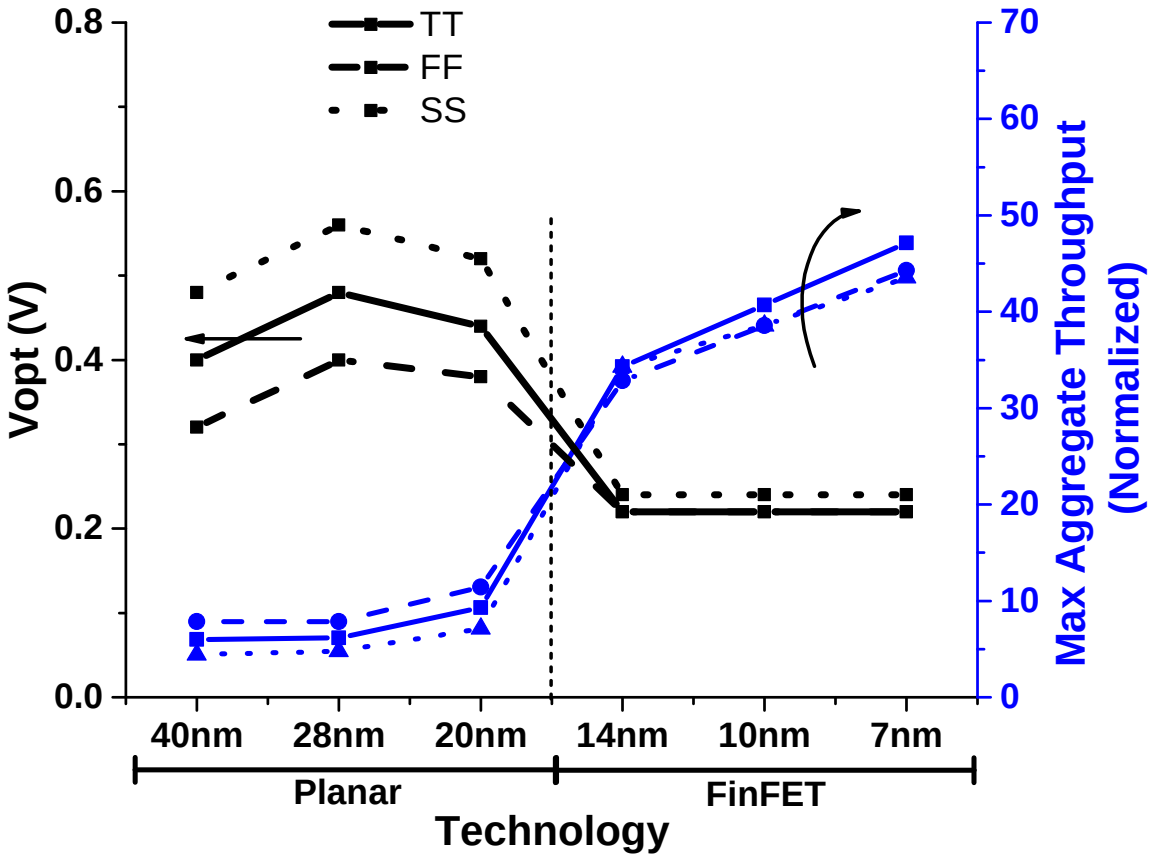


Figure 7.17: V_{opt} **and maximum aggregate throughput across technology nodes when considering global variation and binning is available.** Within-cell and 300TR wire parasitics included.

When binning is not available, all corners are limited to the clock speeds of the SS corner. The TT and FF corners have progressively worse maximum aggregate throughput for a fixed power budget, since TT and FF have higher static energy but do not benefit from faster clock frequencies, Figure 7.18. This causes V_{opt} to be higher for the FF corner.

Local Variation

Local variation affects the threshold voltage (random dopant fluctuations) and channel length (line edge roughness) but, unlike global variation, local variation causes differences

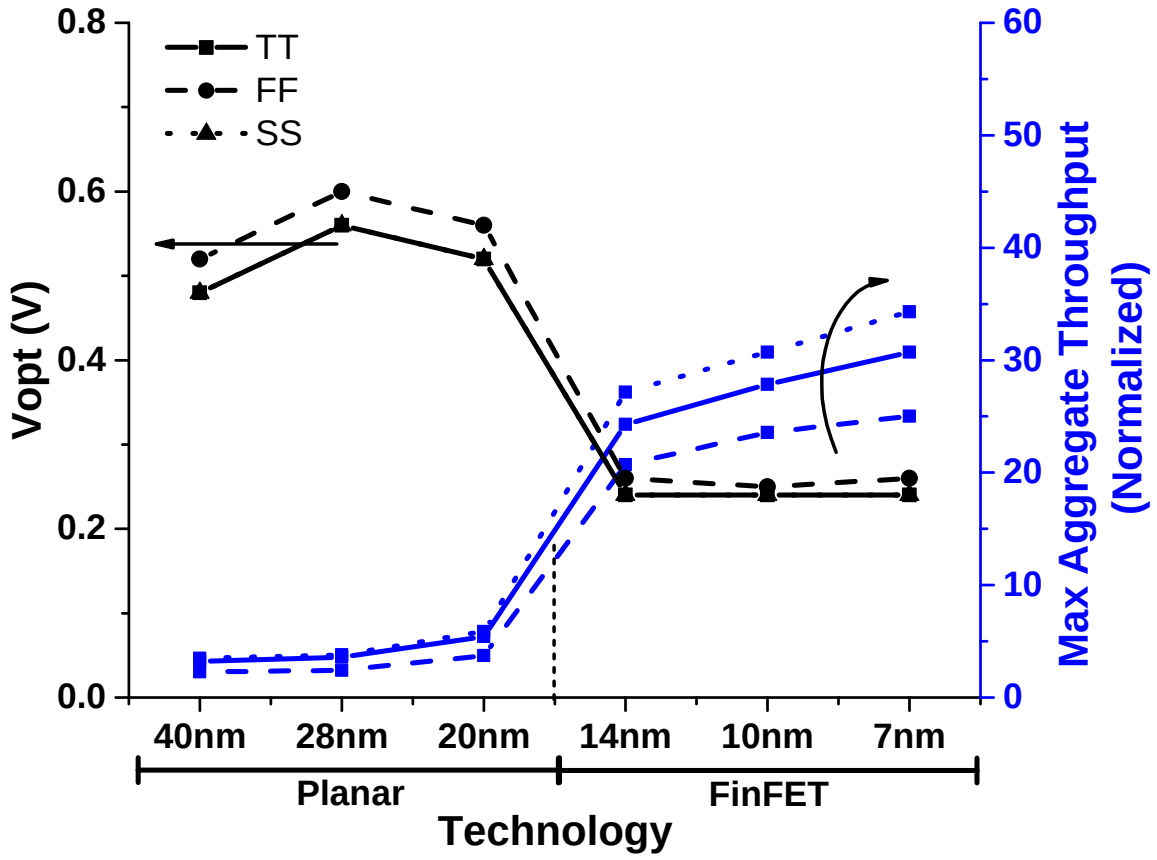


Figure 7.18: V_{opt} **and maximum aggregate throughput across technology nodes when considering global variation and binning is not available.** Within-cell and 300TR wire parasitics included.

between transistors within a chip. In digital designs, short paths and clock skew are particularly sensitive as local variation averages out by the law of large numbers for long logic paths because its impact is uncorrelated.

Local variation's effect on fanout-of-4 delay across supply voltage was included by using conventional Monte Carlo simulations ($N = 1000$) of statistical transistor models. The 3-sigma (3σ) over mean (μ) delay variation of a single inverter gate, normalized to nominal supply voltage, is shown in Figure 7.19. At 220 mV in 7nm FinFET the $3\sigma/\mu$ variation in circuit delay is 3.3× worse than 3-σ/μ at nominal voltage: for example, if there is $3\sigma/\mu = 10\%$ variation in FO4 delay at nominal and near-threshold $3\sigma/\mu = 33\%$. The planar technologies $3\sigma/\mu$ saturates at a higher voltage, primarily because threshold voltage V_t is higher in these technology nodes.

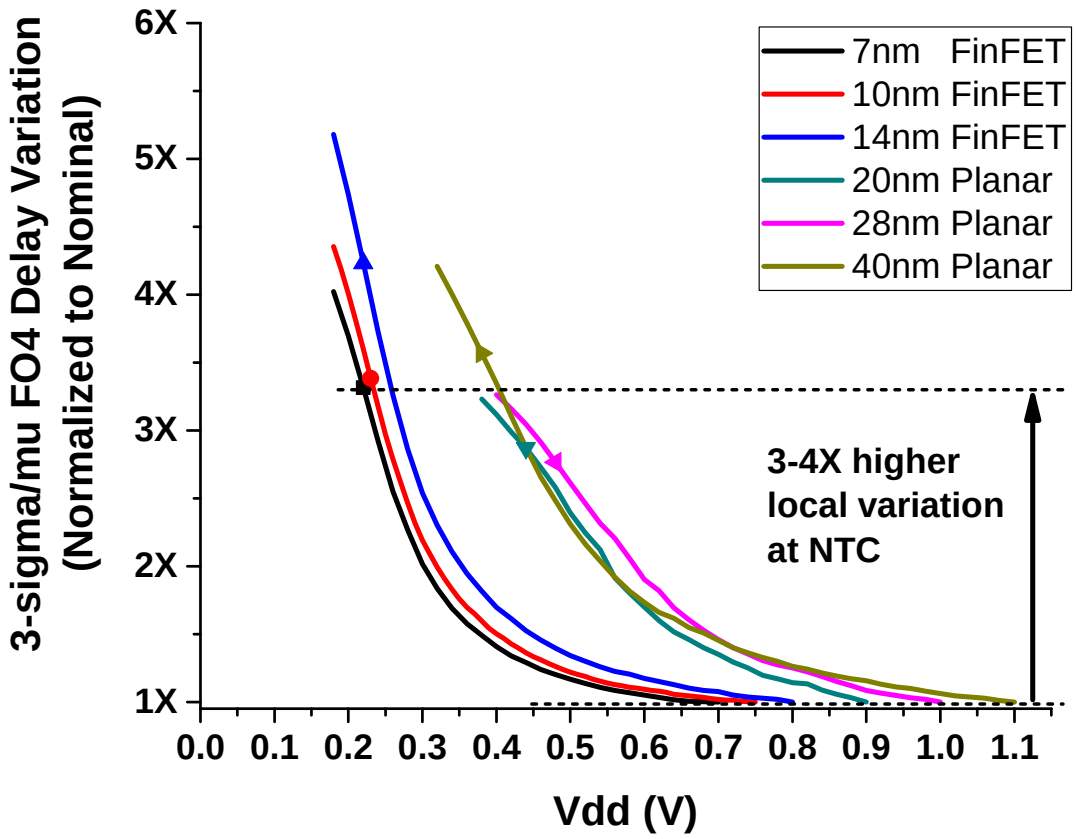


Figure 7.19: **Relative increase in circuit delay variation normalized to nominal voltage in each node.**

We penalize low-voltage operation by derating energy to account for local variation's effects on setup and hold time margining. Increased hold time margins at low voltage are accounted for by approximating that 10% of energy at nominal is used for hold time buffering, and then increasing this percentage as supply voltage is scaled using the $3\sigma/\mu$ variation in Figure 7.19. This coarsely models energy spent on increased hold time buffering, assuming that clock skew and variation through short circuit paths is proportional to $3\sigma/\mu$ variation. In practice, this amount is not directly linear as increased buffers have variations themselves, so we used a voltage-dependent multiplier of up to 1.1× at near-threshold on top of the energy percentage to further derate hold time energy.

When including within-cell and 300TR wire BEOL parasitics with hold time margining, energy gain in 7nm drops from 8.2× to 6.5×, shown in Figure 7.20. Setup time was similarly included by assuming 10% of clock period is for setup time margin at nominal voltage, and then increasing this percentage proportional to $3\sigma/\mu$ delay variation as we did with hold time margin. The energy gain in 7nm further drops from 6.5× to 6.3×, after including setup time margin. Thus, hold and setup time margins are substantial enough that they should be included in voltage scaling analysis. Time borrowing or in-situ correction and detection [71] could mitigate setup time margins, while improved flip-flops [72] and clock buffering [73] may improve hold time margins at low voltage, and therefore achieve higher energy efficiency gains. However, when these margins are included, the 6.3× energy-efficiency gain in 7nm FinFET is still over double the energy gain of 20nm planar of 2.65×.

7.6.2 Area Analysis

Maintaining a fixed latency constraint in near-threshold requires parallelism across cores which costs area, yet the impact of area overhead was not evaluated in [10]. In this section we introduce an area budget while maximizing throughput through voltage scaling, shown in Figure 7.21. Based on technology scaling predictions from Section 7.1, we fix the area to that of a single core in 40nm planar to give a range for achievable near-threshold energy gains with and without an area constraint. As technology scales, transistor density increases to 2.2, 4.0, 5.8, 9.9, and 19 cores for 28nm, 20nmm, 14nm, 10nm, and 7nm, respectively. The

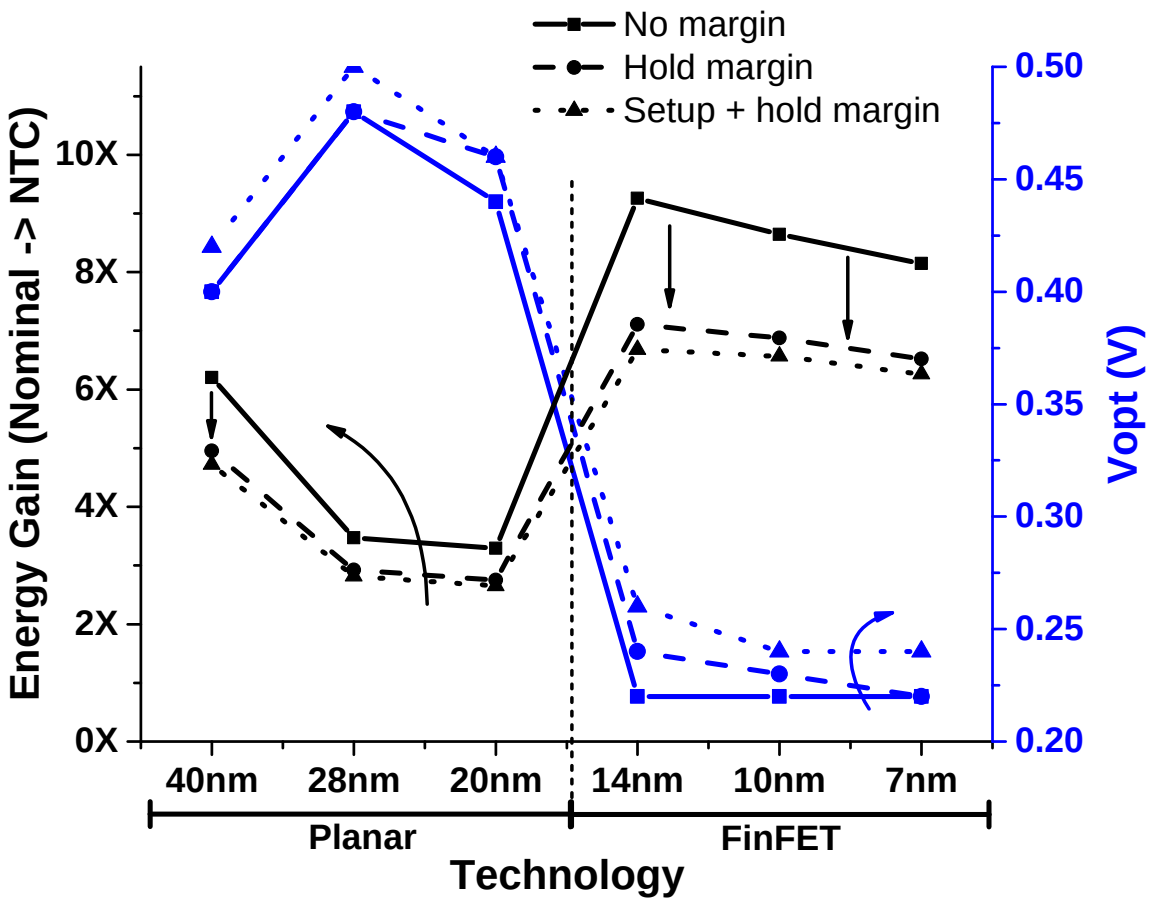


Figure 7.20: **Energy efficiency gain and optimal near-threshold voltage when including hold and setup time margining along with within-cell parasitics and 300TR wire loads.**

effects of BEOL parasitics from a 300TR wire length are also included, while throughput is normalized to 40nm at nominal voltage.

The dashed lines in Figure 7.21 represent throughput at the area constraint and the solid lines are throughput at the power constraint. The solid power-constraint lines are as originally presented in Section 7.3. The dashed area constraint lines start highest on the right side of the graph (nominal voltage) and rapidly decrease as the maximum clock frequency of a processor slows as voltage is scaled. When the dashed line for a technology is above its solid line, the system is area constrained, and vice-versa. Where the two lines cross are when the design is consuming all available power and area.

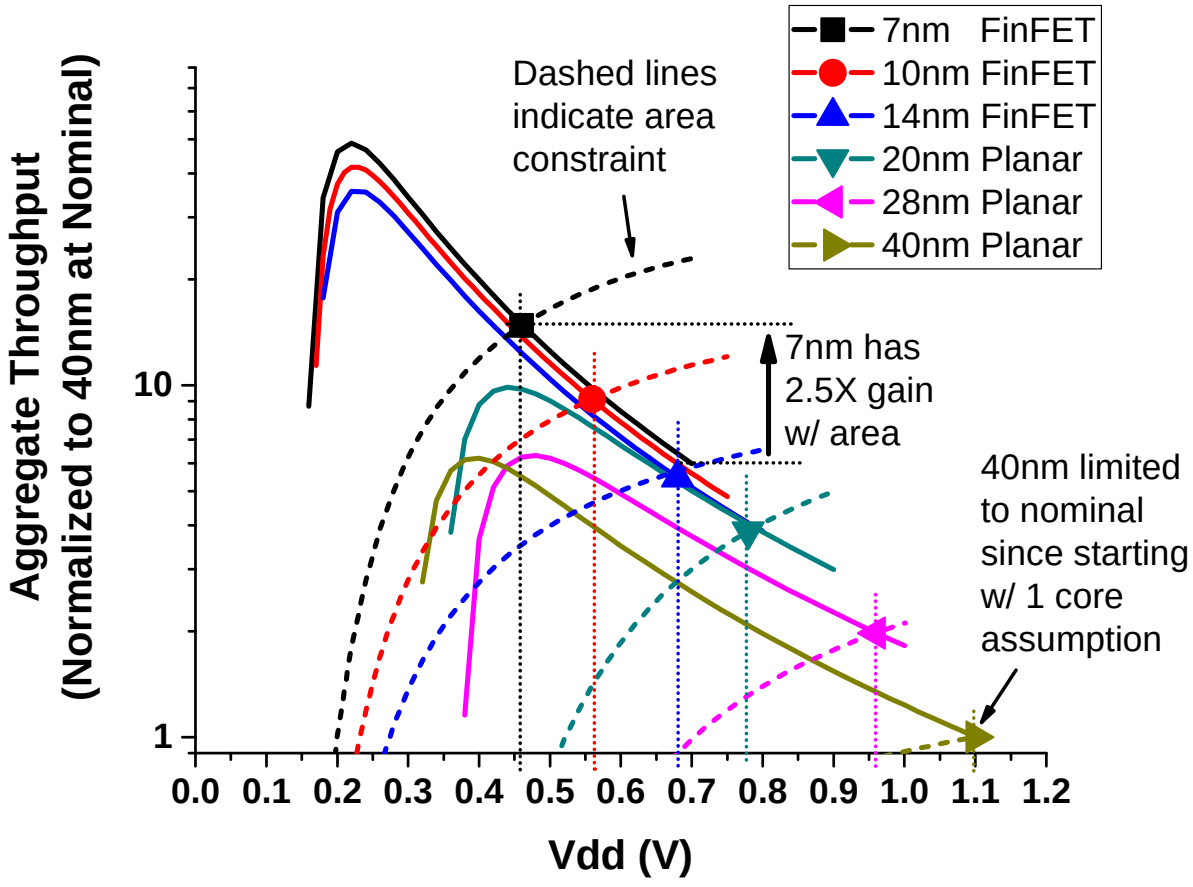


Figure 7.21: **Aggregate throughput across technology when power constrained (solid line) and area constrained (dashed line) with varying supply voltage.** Within-cell and BEOL parasitics of a 300TR wire are included.

The minimum of the solid and dashed lines in Figure 7.21 shows the achievable throughput when considering both power and area budgets for each technology. For example, 7nm FinFET has a maximum throughput of 15 at 460 mV, below which the core is area-limited and above which is power-limited. Without area constraints, a throughput of 49 is obtainable at 220 mV. Instead of an 8.2× energy efficiency gain, only 2.5× is achievable within the 19 core area budget. The three FinFET technologies have progressively less throughput from 7nm to 14nm when area is constrained, despite similar maximum throughput when area is unconstrained. The 10nm FinFET node has a maximum throughput of 9.1 at 560 mV and 14nm FinFET has a maximum throughput of 5.5 at 680 mV.

Energy-efficiency gain and NT voltage V_{opt} is shown in Figure 7.22 for both area constrained and unconstrained. Area constraint dominates achievable energy gains, indicating that near-threshold is more practical for smaller cores that are more power-limited than area-limited. Planar nodes are extremely limited because of the starting single core constraint in 40nm. In reality, chip multiprocessors commonly have multiple cores, even at 40nm, but the area constraint provides a conservative bound for evaluating the practically of near-threshold when area is constrained.

7.7 Conclusions

Near-threshold computing (NTC) has received much interest for overcoming power dissipation limits through improving energy efficiency. However, NTC is observed to be less effective in recent planar technology nodes [10]. In this work we evaluated the impact of device characteristics on voltage scaling and showed how to target a FinFET technology for improved near-threshold operation. FinFET enables significant voltage scaling improvements over planar nodes because of improved channel characteristics, namely less drain-induced barrier lowering (DIBL) and steeper subthreshold slopes. Additionally, continued area density improvements generation to generation contribute to the practically of voltage scaling. A comparison of aggregate throughput of a task at nominal supply voltage and near-threshold, with and without an area constraint, is shown in Figure 7.23.

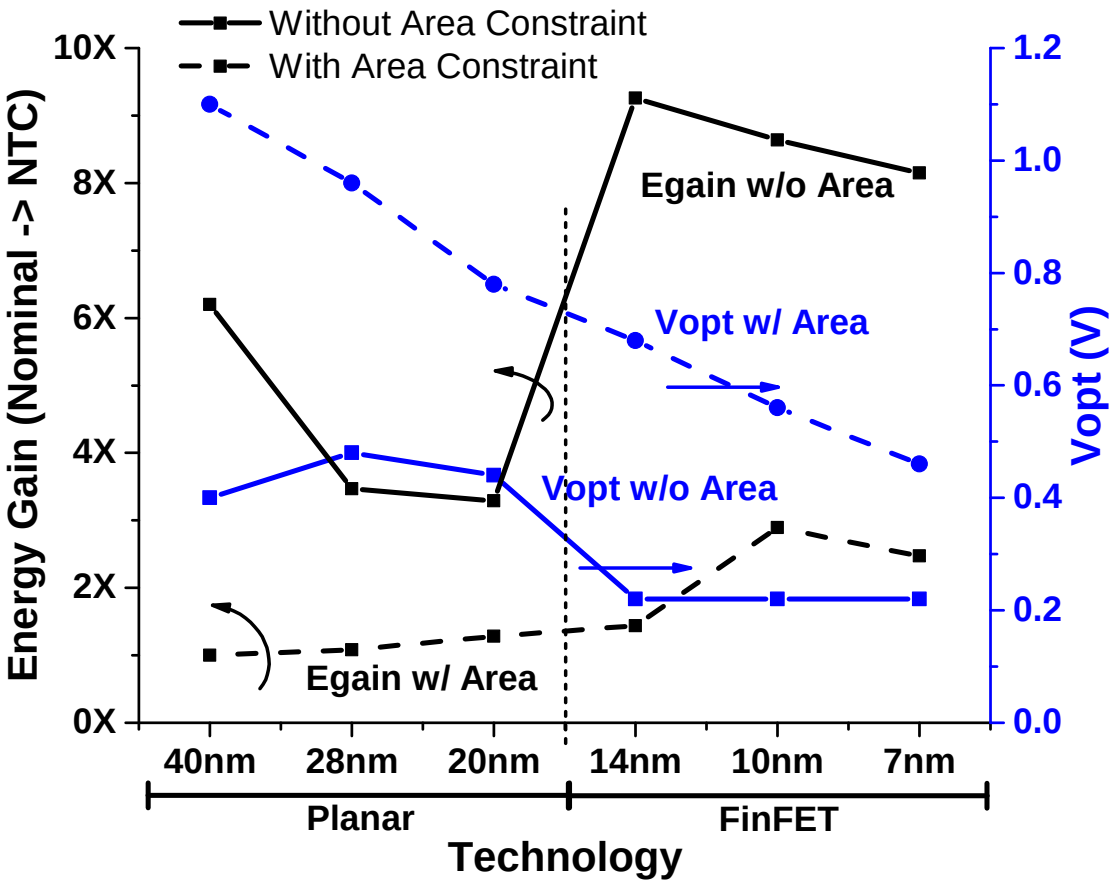


Figure 7.22: **Energy efficiency gain and V_{opt} with and without an area constraint across technologies.** Within-cell and 300TR wire load parasitics included.

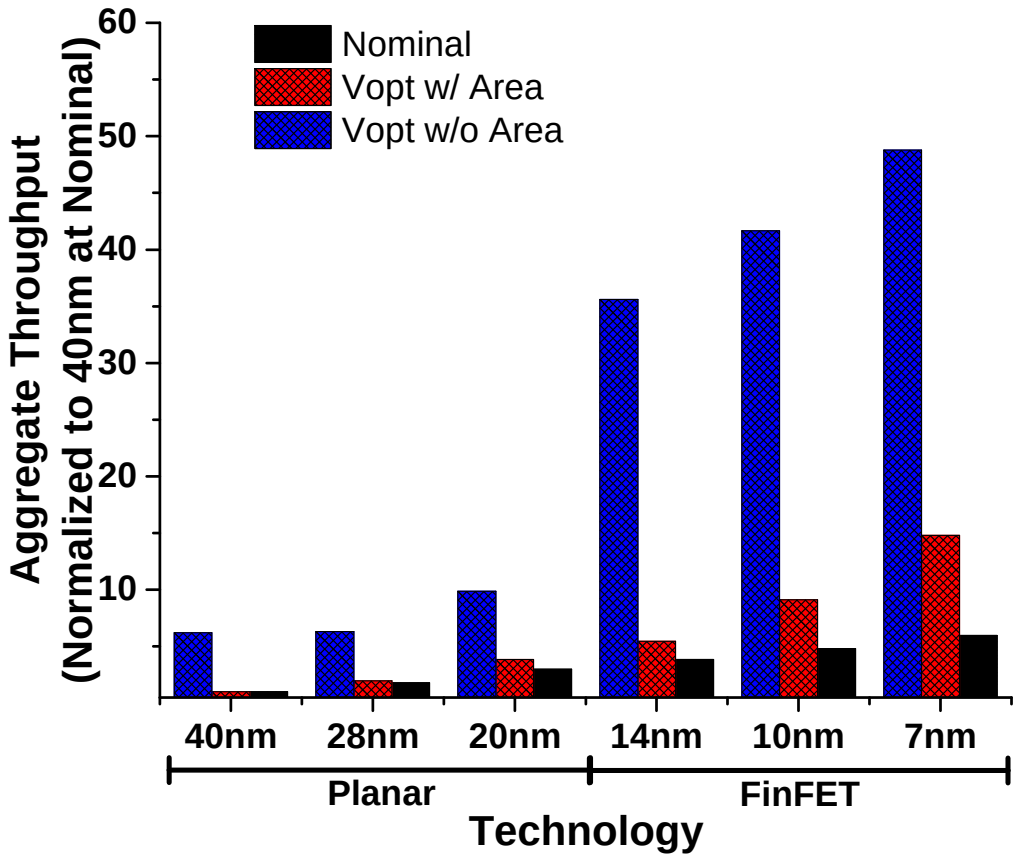


Figure 7.23: **Comparison of aggregate throughput at nominal and near-threshold supply voltages across technologies.** Within-cell and 300TR wire load parasitics included.

FinFET allows for more effective near-threshold operation than ever before because it achieves better voltage scalability and higher area densities. Continued work on circuit techniques to mitigate variation is needed to maximize energy gains. Performance-sensitive NT operation requires algorithms to readily parallelize over more cores, requiring further research on efficient architectures and systems to improve existing algorithms.

CHAPTER 8

Near-Threshold Computing in FinFET Technologies: An Architectural Study

8.1 Motivation

In this chapter, we study computational elements of a futuristic system for wide-area motion imaging (WAMI) to understand how to architect in 7nm FinFET CMOS. The WAMI system is composed of a large application space, from data-parallel pixel processing to high-level object detection and decision making, suggesting a heterogeneous architecture of specialized accelerator units, throughput units, and general-purpose CPUs. We translate the application space to voltage scaling constraints and objectives, in order to understand how near-threshold techniques should influence the architectural design. With these definitions we examine how to maximize throughput in six technology nodes, three FinFET (7nm, 10nm, 14nm) and three planar (20nm, 28nm, 40nm), using transistor models developed by ARM. Unlike previous near-threshold studies, we also include area constraints in our analysis to understand what is achievable within a reasonable area budget.

We find FinFET has significant voltage scaling advantages over planar technologies that allow us to improve energy efficiency, and has a profound impact on architectural design. In 7nm, 2.6−8.9× gains are possible for pixel-level processing elements compared with 1.3−4.9× in 20nm, depending on area constraints. Single tasks that require the highest performance possible have traditionally operated at the nominal voltage of a process, given reliability and

cooling constraints. Increasing single task throughput in recent planar technology (20nm and 28nm) was not possible due to poor circuit delay scaling from short channel effects. We show that in 7nm, because of FinFET's improved channel characteristics, up to 20% throughput increase is possible for single tasks in our WAMI application, with the same power budget.

8.2 WAMI: An Example Application

While near-threshold in FinFET can be applied to a wide range of applications, we choose wide-angle motion imaging (WAMI) in unmanned aerial vehicles (UAVs) as an example in this work. Improving the energy efficiency of WAMI allows more processing to be done on-board the vehicle which decreases communication to satellite or ground stations, thereby reducing the probability of detection or interception. A WAMI processing pipeline starts with high-resolution camera images, up to 1.8 Gigapixel, taken every second. These are analyzed in real-time to identify objects, tracks, events, and threats [74]. A summary of the different tasks in WAMI are shown in Figure 8.1. The bottom of the pyramid is composed of tasks that process enormous amounts of data, for example billions of pixels in an image, but require relatively few computations for each unit processed. Moving up the pyramid to track, event, and threat levels, data coalesces into fewer and fewer units, though more computation is required per unit.

Pixel-level processing includes algorithms such as deblurring or debayer, which must accommodate very high data throughput (1.8Gigapixels/s), so efficiency is crucial at this level. Fortunately, pixel-level algorithms are highly data parallel, which voltage scaling accommodates best. Feature extraction is also applied at this level to recognize objects within the scene, such as a car. Data from the pixel level is provided to the track level to follow movements of objects through optical flow algorithms, which at the next level are classified into events, such as a car executing a U-turn. Finally, threat level uses machine learning algorithms to identify threats from patterns in the event data.

The goal of this work is to understand the efficacy of voltage scaling at each WAMI pyramid level, and use this knowledge to increase energy efficiency when architecting a low-power, high-performance, embedded system. Pixel-level algorithms require different computational

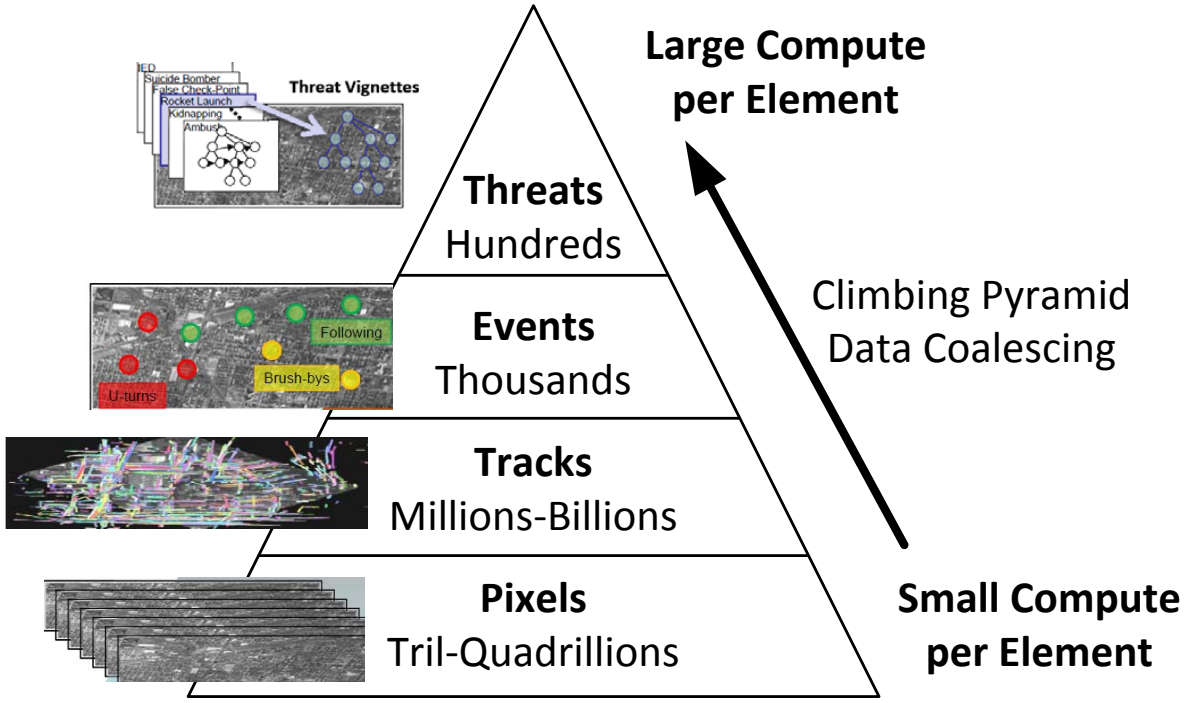


Figure 8.1: **Workload pyramid for our wide-area motion imaging (WAMI) application.** Low levels (pixels) process lots of data in parallel, while higher levels are single tasked.

blocks than higher-level algorithms. For example, at the bottom of the pyramid, kernel algorithms such as deblur can be mapped to specialized hardware accelerators like a Fast-Fourier Transform (FFT) unit. Midway up the pyramid the algorithms require more control flow, and subsequently can be mapped to general purpose throughput processors, similar to general-purpose graphic processing units (GPGPUs). At the top of the pyramid the workloads exhibit higher percentages of serial code and significant amounts of branch divergence, so more conventional multi-processors are desired.

Each of these levels has different performance requirements and parallelism characteristics. Latency of a kernel is less important at the bottom of the pyramid since overall code size in each kernel is small and, additionally, parallelizes well. General-purpose throughput processors also parallelize well, but latency should be maintained for these applications because of larger code size. Finally, the top of the pyramid is highly serialized and bottled-necked, so minimizing latency is critical.

WAMI includes many algorithmic kernels, and characterizing their parallelism behavior is ongoing. For the purposes of this work, we use a methodology similar to Pinckney et

al. [10] which models workload scaling using Amdahl's law [41], thus we use three Amdahl serial coefficients for three levels of the WAMI pyramid. Because the bottom of the pyramid, pixel-level, is very data parallel we estimate a serial coefficient of 2%, which is similar to the serial percentage seen in SPLASH-2 scientific benchmarks [10, 75]. The workloads above pixel-level are undergoing development, but are more serialized than pixel level. We picked a 5% Amdahl serial coefficient to demonstrate track-level. For event-level we show a range from a pessimistic 25% case to an optimistic 10% case.

Exact serial coefficient values are not critical to understanding our methodology, but were picked for illustration purposes and values for WAMI workloads may be revised in future work as the algorithms are better characterized. The same concepts used in this work can be readily applied to other application spaces if workload behavior is known.

8.3 Circuit Analysis

8.3.1 Methodology

This analysis uses a similar framework to that used by Pinckney et al. [10,11] for estimating energy and performance when voltage scaling in planar technologies. Circuit simulations use HSPICE BSIM Level 72 models for 7nm, 10nm, and 14nm FinFET, and Level 54 models for 20nm, 28nm, and 40nm planar, of which all sets were developed by ARM based on published numbers, historical trends, and informed assumptions and calculations. The canonical circuit simulated to model circuit effects is a chain of thirty-one inverters, which emulates reasonably deep processor pipelines. Though actual critical paths are composed of more complex gates, we found inverters are reasonable for comparing performance and energy between operating voltages and technologies. Within our circuit model, we also included back-end-of-line (BEOL) parasitics, which are additional capacitors and resistors from wires that interconnect gates on a chip. Back-end-of-line is important to model as achievable energy gains when BEOL is included are lower than with ideal wires. Lastly, impact from across die mismatch variation is accounted for by derating a percentage of the total energy

and minimum clock period, proportional to increases in variation, to penalize low voltage operation.

Architectural effects and overheads from scaling our target WAMI algorithms across multiple cores was modeled using Amdahl's law [41], with a percent serial coefficient of 2% for pixel-level workloads, 5% for track-level, and 10%/25% for event-level. These are not exact values but provide sufficient results to estimate efficacy of voltage scaling. A percent serial coefficient of 2% is used to describe the voltage scaling scenarios, but we provide a table of all three sets of serial coefficients at the end of this section.

Final energy and performance estimates were calculated by combining circuit and architectural data using MATLAB. The final energy and performance numbers are used to generate throughput estimates under different voltage scenarios.

8.3.2 Power and Area Scaling

The number of active cores on a die can be limited by power or area constraints. Thermal design power (TDP) is the allowable power that can be readily cooled by a system, and has maintained relatively constant generation-to-generation. The area scaling, or transistor packing density, has been improving allowing more cores within a fixed die area for each new generation.

Figure 8.2 shows power and area scaling predictions. Power scaling is estimated by simulating the ARM transistor models, while area scaling is derived from publicly available foundry data. For this analysis we evaluated core logic only. Caches, with their significantly lower activity rate, are less power dense. Therefore, within the same power budget, it is easier to increase the cache size on a microprocessor than add additional cores as technology scales.

For a TDP that can support a single core in 40nm, 4.7 cores can be powered in 7nm. While power has improved by a factor of 4.7×, area scaling has improved much faster between 7nm and 40nm. Within a single core's area in 40nm, approximately 19 cores can fit in 7nm. This is less than scaling predicted geometrically by squaring the differences in linear feature size, $(40\text{nm}/7\text{nm})^2 = 33$, but still much greater than the 4.7× power scaling. Therefore,

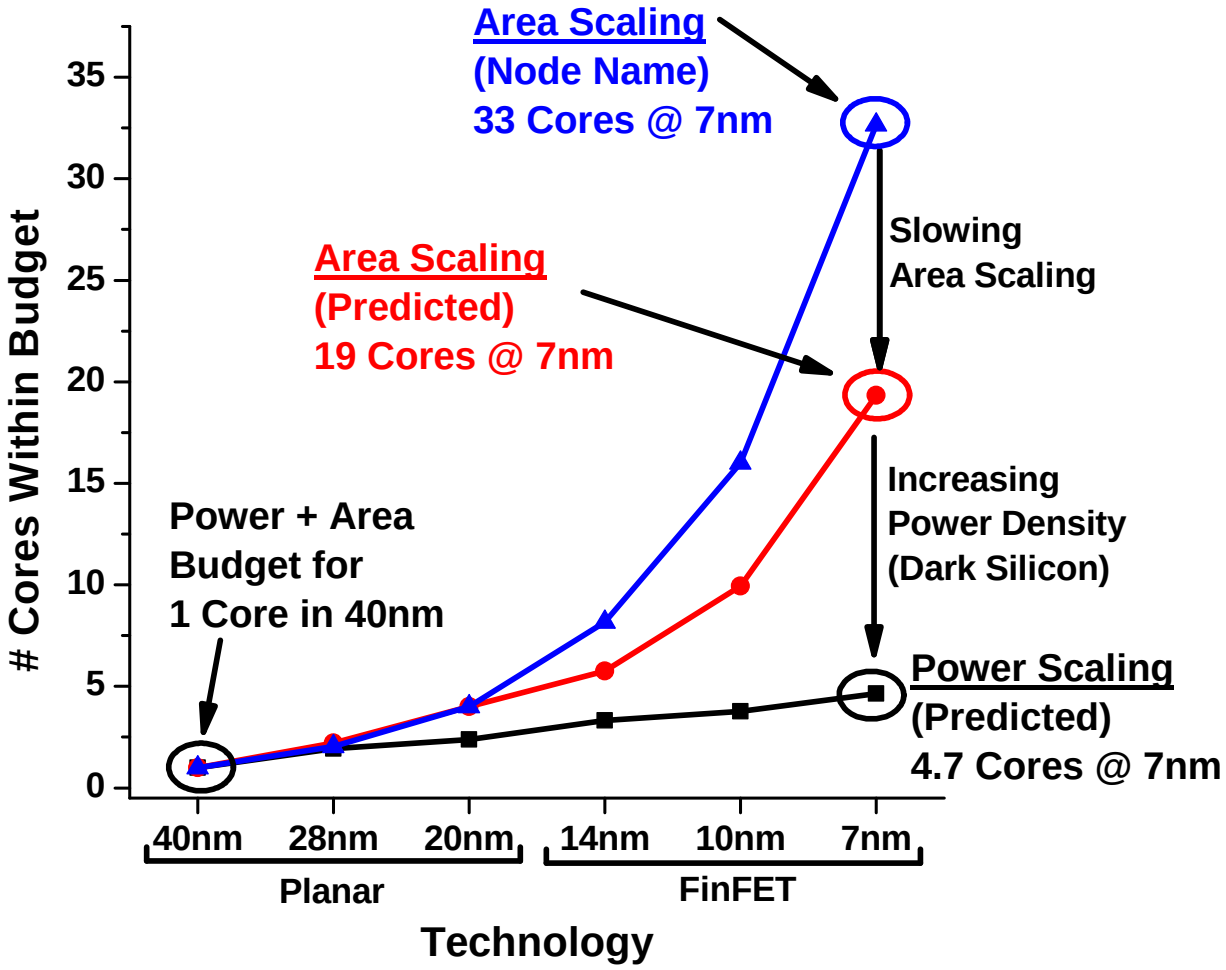


Figure 8.2: **Power and area scaling normalized to 40nm.** Area scaling has outpaced power scaling, leading to a rise in power density.

technology scaling has allowed for a large increase in cores within a fixed area, but only a small increase of cores within a fixed power budget. Previously, Dennard scaling [3] allowed for area and power to track, therefore the ratio between power and area (power density) remained constant generation-to-generation. However, Figure 8.2 demonstrates that power density in recent technology nodes is increasing because area density is rising at a faster rate than power is improving.

This mismatch between power improvements and area improvements has led to the term dark silicon [5], where we can fit many cores on a chip but not power them. Voltage scaling exploits this mismatch by increasing the energy efficiency of cores, thereby reducing power, at the cost of greater area.

8.3.3 Circuit Delay Scaling

Maximum clock frequency of a processor is limited by its critical circuit path, by definition. Circuit delay for a technology is traditionally measured with fanout-of-4 (FO4) delay, the delay of a logic gate driving four copies of itself, since it is insensitive to absolute transistor sizing and representative of an optimized circuit path. For this work, fanout-of-4 delay of inverters was measured in simulation to characterize circuit performance when voltage scaling, but we observed similar results for other common CMOS gates, such as NANDs and NORs.

A plot of clock frequency relative to supply voltage (Vdd) is shown in Figure 8.3. Supply is also normalized to maximum supply voltage of each technology. Clock frequency, as estimated using inverter FO4 delay, was also normalized to maximum clock frequency of each technology to show degradation in speed as voltage is lowered. A shallower slope reflects better circuit delay scaling.

Prior work [10] found NTC tracked 200-400mV above transistor threshold voltage, thus Figure 8.3 highlights the minimum and maximum of this range across technologies as a percent of nominal Vdd (43% to 85% of nominal). We denote the clock frequency degradation at the bottom end of this range (43% of nominal). To the first order, energy savings is quadratic with supply voltage (CV^2). Therefore, a 57% reduction in Vdd (43% of nominal

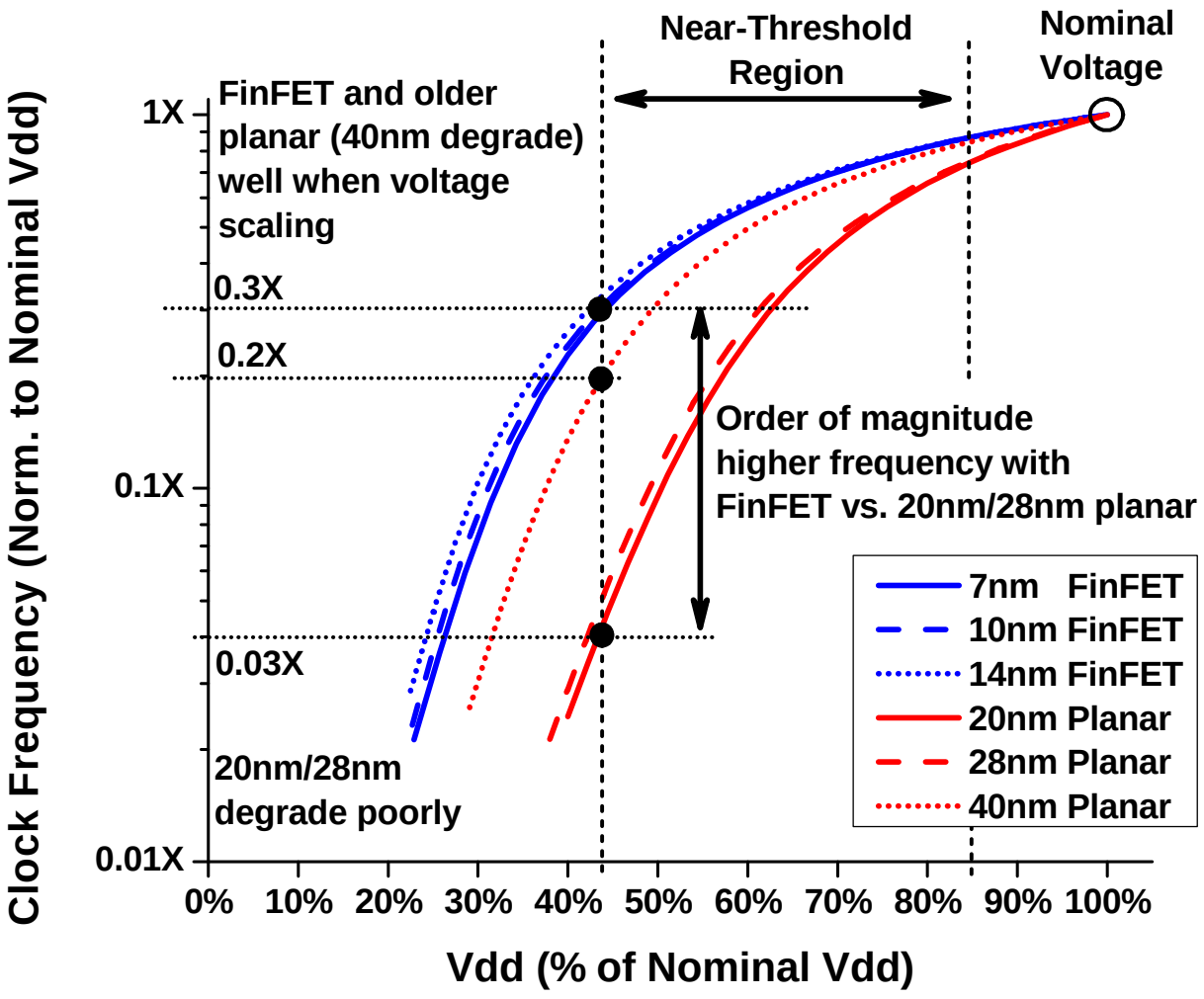


Figure 8.3: **Voltage scaling's impact on clock frequency of a circuit.** FinFET scales better than planar.

Vdd) very roughly translates to a 82% energy reduction. Less clock frequency degradation mitigates amount of needed parallelism to maintain performance, and thus reduces overheads associated with parallelizing, thereby enabling performance-sensitive tasks to operate at a lower voltage and realize higher energy efficiency gains.

Of the three planar nodes included in this study, 40nm clock frequency scales the best with voltage (5× slower clock speed at 43% of nominal Vdd). Successive planar generations, 28nm and 20nm, become progressively worse at voltage scaling (33× slower clock at 43% of nominal Vdd), because of increased short channel effects and poor subthreshold slope. FinFET has much better channel characteristics, thus exceeds planar voltage scaling despite possessing small lithographic feature sizes and good area density. At 43% of nominal Vdd 7nm, 10nm, and 14nm FinFET's clock speed is 3× slower than nominal, an order of magnitude higher than 20nm planar. Therefore, FinFET differs from planar by degrading in performance the least when voltage scaling, contributing to reduced parallelism to achieve the same performance. Additional technology factors, such as improved subthreshold device leakage, also allow FinFET to scale lower than planar technologies.

8.3.4 Voltage Scaling Scenarios

When considering the performance of a near-threshold system, three voltage scaling scenarios are evaluated depending on the prioritization of task latency versus overall system throughput. The three scenarios are summarized in Table 8.1. To simplify analysis, all tasks running on a system are assumed to be identical when analyzing throughput, though for our system-level analysis in the next section we partition units based on the task.

The first voltage scaling scenario is when all processors in a system are utilized by a single task to maximize speedup, and no other tasks are run on the system. Absolute maximum single task performance is achieved, however because adding additional cores to a task may only marginally improve performance, the overall energy efficiency is impacted. The second scenario considers the aggregate throughput of multiple tasks running the system, but the tasks are performance sensitive, which is accounted for by constraining a task's latency to that of the task running on a single core at maximum supply voltage and frequency.

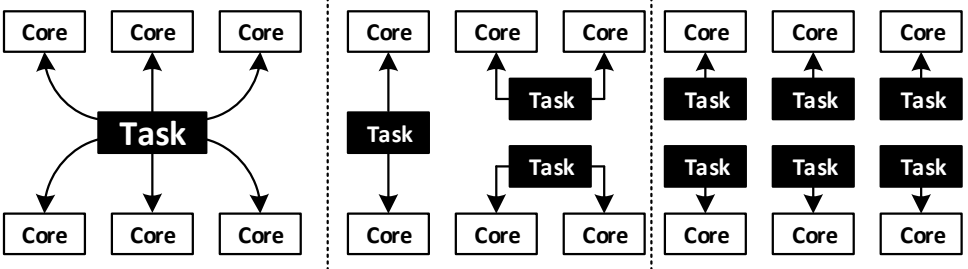
Scenario:	#1	#2	#3
Goal:	Maximize Single-Task Performance	Balance Single/Many-Task Performance	Maximize Many-Task Performance
Latency:	Minimized	Fixed to Latency of 1 Core @ Nominal	Unconstrained
System Configuration:			

Table 8.1: Voltage scaling scenarios when latency is minimized, fixed, or unconstrained. Traditional computing is scenario one, while ultra-low voltage is scenario three and recent near-threshold definitions scenario two.

As supply voltage is lowered and clock frequency degrades, tasks are parallelized across more cores until the fixed latency constraint is met. This definition was used to define the near-threshold region by Pinckney et al. [10]. The final scenario again considers aggregate throughput of many tasks, but each task is assigned to a single core and latency is completely unconstrained. In this scenario tasks may take very long to finish, but run very efficiently by minimizing energy consumption.

Each of these voltage scaling scenarios is described in detail within the following subsections. A percent serial of 2% is used to initially explain each scenario's behavior, but we conclude this section by expanding analysis to 5%, 10%, 15%, and 25% serial.

Minimizing Latency

The relative throughput of the single-task scenario is shown in Figure 8.4 while supply voltage is swept across technologies. To include effects of dark silicon, the starting constraint is with a power and area budget sufficient to run one core in 40nm, and subsequent technologies scale as mentioned in Section 8.3.2. Long and short dashed lines indicate maximum throughput within the power and area budgets, respectively. Solid lines denote throughput satisfying both area and power constraints. Each color represents a different CMOS technology, from 40nm to 7nm. The area constraint lines are highest at nominal voltage, and

rapidly decrease as voltage is lowered since clock frequency degrades. The power constraint lines feature a peak, representing maximized energy efficiency above which dynamic energy dominates, and below which leakage energy or parallelization overheads dominate. For 10nm and 14nm, the short dashed area budget lines bisect the power budget line, indicating best case efficiency gain when area is considered.

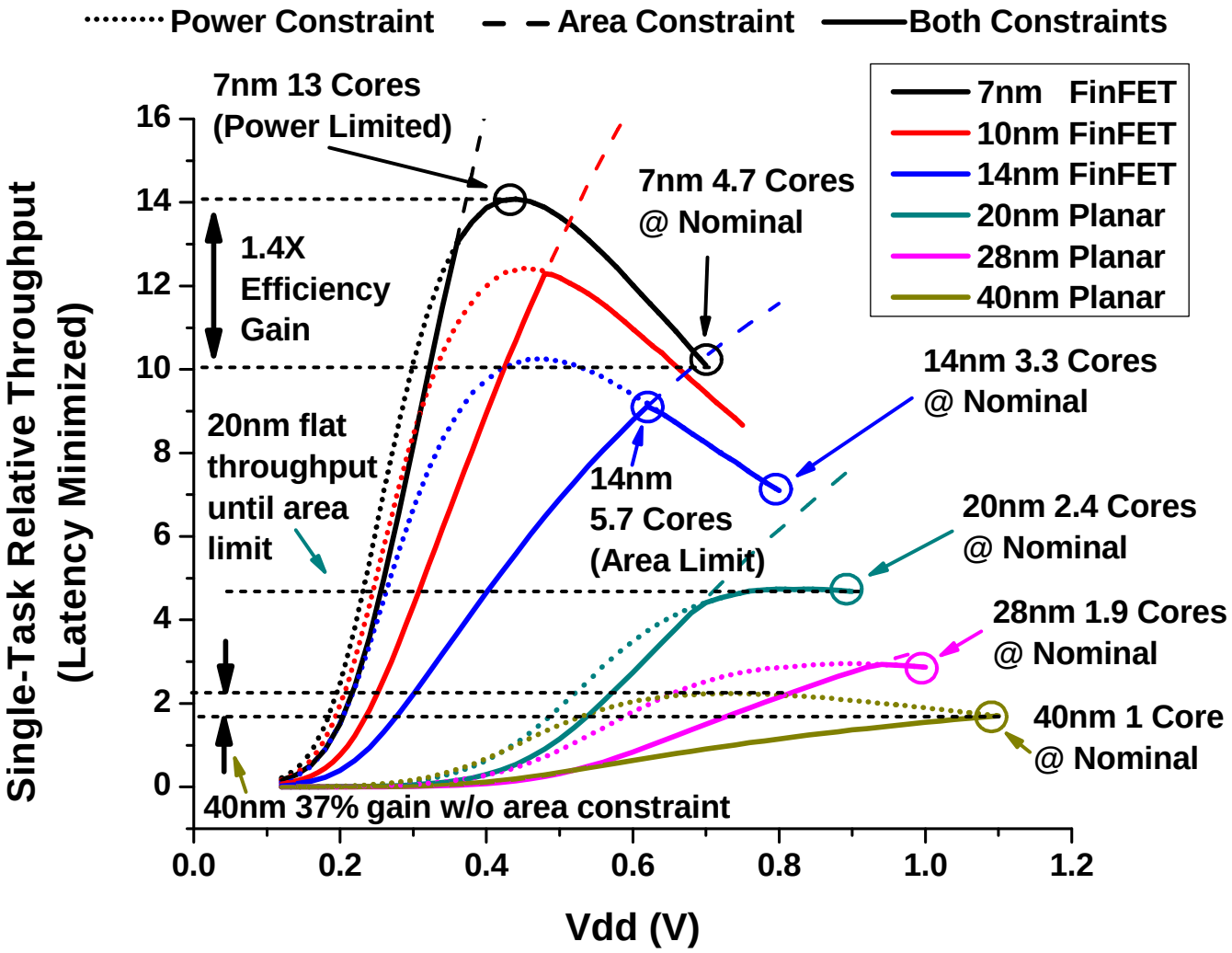


Figure 8.4: **Throughput of single-task scenario (minimizing latency) across six technologies when power and area are constrained.** FinFET scaling and area density allow for improved single-task performance, which was not possible in older CMOS technologies.

Area is extremely limited in planar nodes and, in this example, can only support one core in 40nm, two cores in 28nm and four cores in 20nm. Along with poor circuit delay scaling, area limits best-case energy efficiency gains in planar nodes. Without constraining

area, 40nm is able to increase single-task throughput by 37% by operating at 0.7V instead of 1.1V. However, newer planar nodes suffer from poorer circuit delay scalability (degradation in clock frequency at low voltages), exemplified by 20nm and 28nm having negligible throughput increase when voltage scaling.

FinFET technologies exhibit better circuit delay scalability, and are much less area-limited, thus are able to improve single task performance by 30% in 14nm, 40% in 10nm, and 40% in 7nm, even while constraining area. The power budget allows for 4.5 cores at nominal voltage (700 mV) in 7nm FinFET, despite having area for 19 cores. Lowering the voltage in 7nm to 440 mV maximizes throughput by allowing 13 cores to operate within the power budget. Below 440 mV, throughput reduces because of degrading clock frequency.

Despite low voltage operation's conventional use only during periods of minimal processor load, FinFET's superior circuit delay scalability and the shift to dark silicon, has introduced a new opportunity for voltage scaling to improve energy efficiency even for high-performance applications, so long as a task is sufficiently parallelizable.

Fixed Latency Constraint

The second scenario is when aggregate throughput of many tasks is maximized, subject to latency and power constraints, shown in Figure 8.5 without (top) and with (bottom) area constraints. Latency is constrained to that of the task running at nominal supply voltage on a single core. As voltage is lowered the clock frequency degrades, however latency can be maintained by parallelizing. If parallelism overhead is sufficiently small, energy efficiency can be improved while maintaining fixed latency for the task. In other words, for energy savings, the energy overhead needed to parallelize the application across more cores should be less than the energy gain from running at the lower voltage. Recall that power is proportional to energy times rate (rate inversely proportional to latency in this example). Because energy for a task improves, while latency remains constant, additional tasks can be run within the same power budget.

Without an area constraint, 40nm planar had the best throughput gains across the three planar nodes, and gains become progressively worse in 28nm and 20nm (4.8× gain in 40nm, 2.8× in 28nm, and 2.7× in 20nm). FinFET exceeds all planar nodes in throughput, with

gains of 6.8×, 6.7×, and 6.4× for 14nm, 10nm, and 7nm, respectively. Including an area constraint further limits planar nodes to a maximum gain of 1.3× in 20nm, while FinFET can achieve 2.5× in 7nm because of increased number of cores that fit within the area budget. A practical implementation will fall between the area constrained and unconstrained estimates ($2.5 - 6.4\times$ energy efficiency gain in 7nm) depending on the size of the core and room dedicated to cores versus other peripherals on an SoC.

Unconstrained Latency

The final scenario is when latency is unconstrained and aggregate throughput is maximized, shown in Figure 8.6. In this scenario the throughput gains from voltage scaling without an area budget are much larger than the previous scenario, because tasks always run on a single core, so that any power savings from running at a lower voltage directly translates into the ability to run additional tasks on previously inactive cores.

With the area budget of one core in 40nm, all technologies have approximately the same gains as that of the fixed-latency scenario, since the amount of parallelism is relatively small in the fixed-latency scenario (no more than a couple of cores per task), thus overheads from running on multiple cores are negligible. However, without an area constraint the voltage is pushed lower, achieving gains of 8.9×, 9.6×, and 9.8× in 7nm, 10nm, and 14nm FinFET, and gains of 4.9×, 5.7×, and 8.3× in 20nm, 28nm, and 40nm planar, respectively. Since latency is unconstrained, circuit delay scalability with voltage has less of an impact than the previous scenario, therefore higher throughput gains are achievable.

8.3.5 Sensitivity to Parallelism Overheads

The previous results were for a relatively parallel task (Amdahl percent serial = 2% or less), equivalent to estimates for pixel-based workloads in WAMI. The event-level and track-level workloads aggregate more data, and therefore are less parallelizable. This reduces achievable efficiency gains, as more parallelism is needed for the same speedup at lower voltage if latency of a task is fixed or minimized. To account for higher serial percentage we swept percent serial for the three different scenarios and show results for 2%, 5%, 10%,

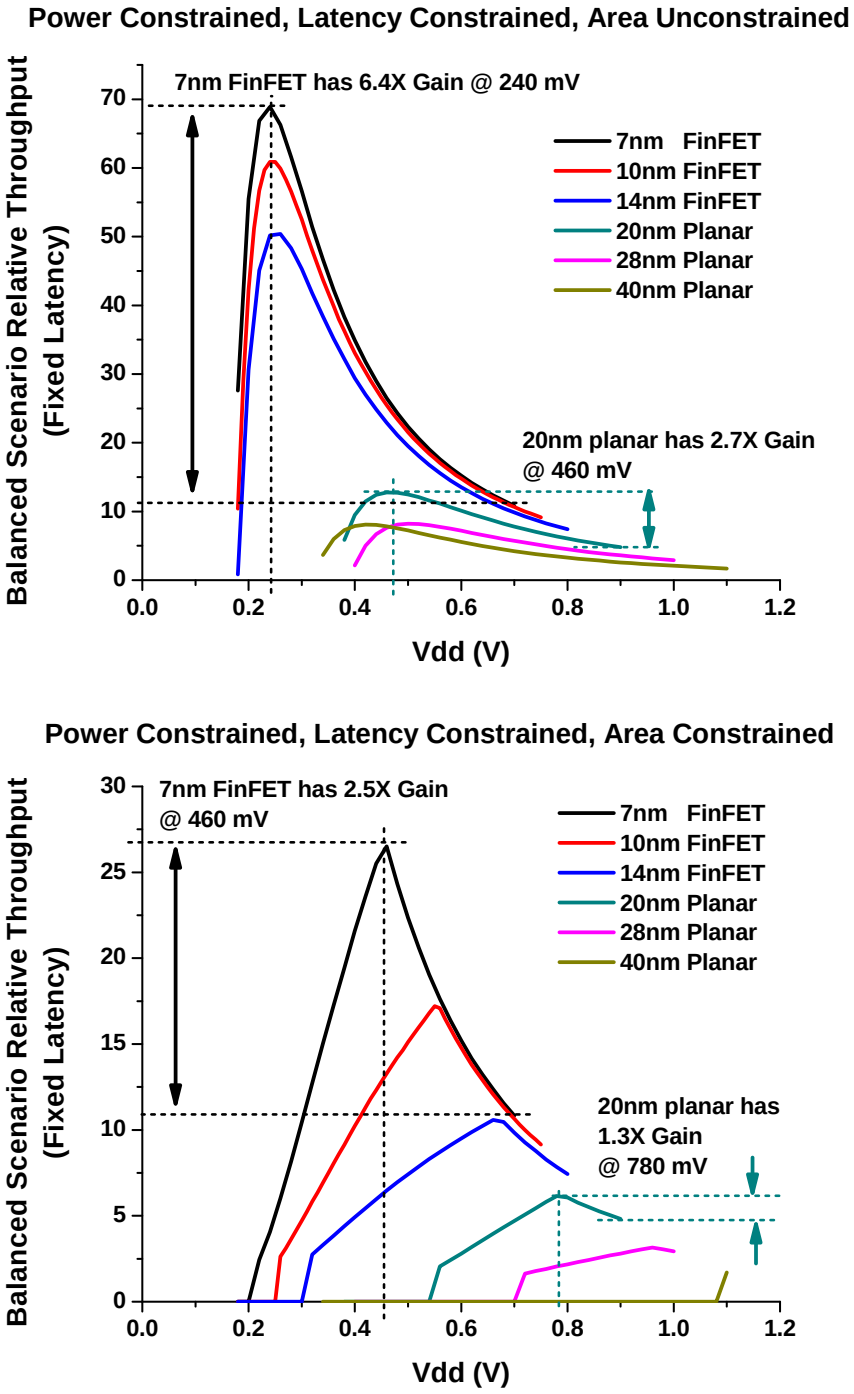


Figure 8.5: **Throughput of many tasks when latency constrained, without an area budget (top) and with an area budget (bottom).** FinFET is drastically better than planar for both energy gains and absolute throughput. An area constraint limits achievable gains, but newer technologies scale better because of higher packing densities.

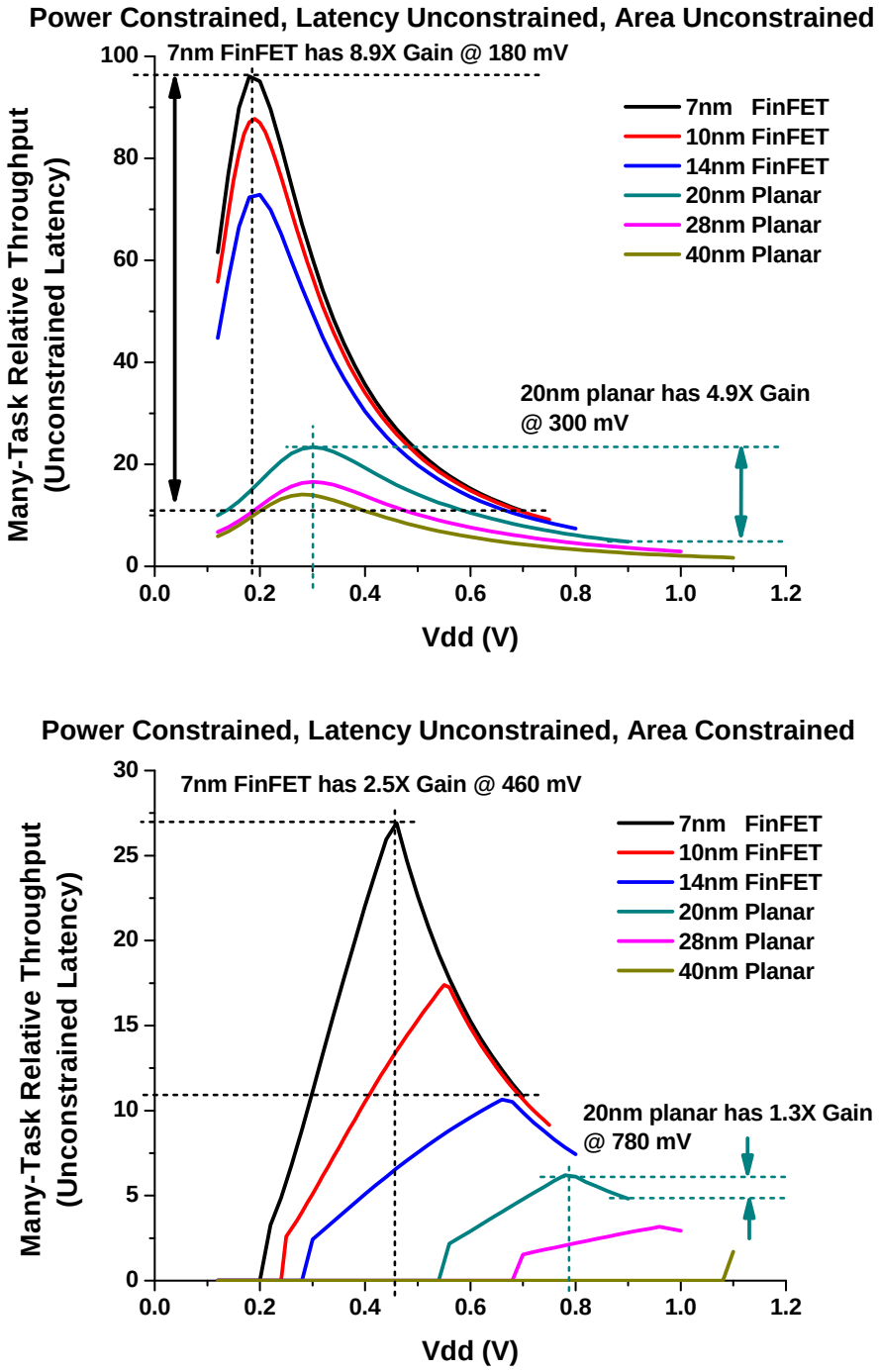


Figure 8.6: **Throughput of many tasks when task latency is unconstrained, without an area budget (top) and with an area budget (bottom).** When area constrained results are similar to fixed latency results, since needed parallelism was small. Without an area constraint higher energy efficiency can be achieved.

25% in Table 8.2 for 7nm FinFET and 20nm planar. With and without area constraints are included, to compare achievable efficiency gains for both large and small cores, respectively. In our scaling example, 20nm only has area budget for four cores and power budget for 2.4 cores at nominal Vdd, versus area for 19 cores and power for 4.7 cores in 7nm. The relative throughput, number of cores per task, and number of total cores is also included in the table, to compare performance across scenarios and technologies, and to gauge which configurations are practical.

Throughput Gain, Maximum Relative Throughput, and Number of Cores												
7nm FinFET												
Scenario	**Single-Task (Minimize Latency)**				**Balanced (Fixed Latency)**				**Many-Task (Unconstrained Latency)**			
	Gain	**T-Put**	**Cores /Task**	**Cores Total**	**Gain**	**T-Put**	**Cores /Task**	**Cores Total**	**Gain**	**T-Put**	**Cores /Task**	**Cores Total**
% Serial	**Area Unconstrained**											
2%	1.4×	14	13		6.4×	69	11	320	8.9×	96	1	1,500
5%	1.2×	11	11		4.9×	53	9.1	210	”	”	”	”
10%	1.1×	9	8		3.7×	40	5.9	100	”	”	”	”
25%	1.0×	6	5		2.2×	24	3.0	31	”	”	”	”
% Serial	**Area Constrained (19 Cores Max)**											
2%	1.4×	14	13		2.5×	27	1.6	18	2.5×	27	1	18
5%	1.2×	11	11		2.4×	26	1.6	18	”	”	”	”
10%	1.1×	9	8		2.3×	25	1.7	18	”	”	”	”
25%	1.0×	6	5		2.0×	22	2.0	18	”	”	”	”
20nm Planar												
Scenario	**Single-Task (Minimize Latency)**				**Balanced (Fixed Latency)**				**Many-Task (Unconstrained Latency)**			
	Gain	**T-Put**	**Cores /Task**	**Cores Total**	**Gain**	**T-Put**	**Cores /Task**	**Cores Total**	**Gain**	**T-Put**	**Cores /Task**	**Cores Total**
% Serial	**Area Unconstrained**											
2%	1.0×	5	3		2.7×	13	14	85	4.9×	23	1	1,700
5%	1.0×	5	3		2.1×	10	7.0	35	”	”	”	”
10%	1.0×	4	2		1.8×	8	4.2	18	”	”	”	”
25%	1.0×	4	2		1.3×	6	2.0	5.8	”	”	”	”
% Serial	**Area Constrained (4 Cores Max)**											
2%	1.0×	5	3		1.3×	6	1.3	4	1.3×	6	1	4
5%	1.0×	5	3		1.3×	6	1.3	4	”	”	”	”
10%	1.0×	4	2		1.2×	6	1.4	4	”	”	”	”
25%	1.0×	4	2		1.1×	5	1.4	4	”	”	”	”
Summary of 7nm FinFET vs. 20nm Planar												
Scenario	**Single-Task (Minimize Latency)**				**Balanced (Fixed Latency)**				**Many-Task (Unconstrained Latency)**			
	Gain	**T-Put**	**Cores /Task**	**Cores Total**	**Gain**	**T-Put**	**Cores /Task**	**Cores Total**	**Gain**	**T-Put**	**Cores /Task**	**Cores Total**
Average in 7nm:	**1.2**×	10	9.3		**3.3**×	36	4.5	92	**5.7**×	62	1	760
Average in 20nm:	**1.0**×	4.5	2.5		**1.6**×	8	4.1	20	**3.1**×	15	1	850

Table 8.2: Summary of throughput (energy efficiency) gains from voltage scaling in 7nm FinFET and 20nm planar. Relative throughput and number of cores are also listed to compare across scenario or technology.

Increasing percent serial decreases the achievable gains, especially for the minimum latency scenario, since serial code can never be sped up through parallelism. In 7nm, a percent serial of 25% shows no gain in the single-task scenario. Area constrained and unconstrained is nearly identical in 7nm for a single task, since the number of cores is always less than the area budget. Because of poor circuit delay scaling in 20nm planar, a single-task can never be improved through voltage scaling.

Larger percent serial also reduces gains in the fixed-latency scenario when area unconstrained. However, this effect is marginalized when area constrained, since in this case the number of cores parallelized across is relatively small. Varying the percent serial does not impact the unconstrained latency scenario, as a task always runs on a single core and is never parallelized.

The relative throughput numbers help compare performance across technology. For instance, when latency unconstrained 7nm has a throughput of $96/8.9 = 11$ units at nominal Vdd, compared to $23/4.9 = 4.7$ units in 20nm. Therefore, at nominal Vdd, 7nm has an $11/4.7 = 2.3\times$ increase in throughput compared to 20nm, despite three generations of process improvements. Thus, even a $2\times$ gain through voltage is significant compared to gains from technology improvements. The bottom of Table 8.2 includes a summary comparing 7nm FinFET to 20nm CMOS planar, by averaging across all values for each scenario.

The sensitivity of gain to percent serial, when voltage scaling a single task to minimize latency, is shown in Figure 8.7. Area was unconstrained as few cores are consumed in this scenario. A perfectly parallelizable task (0% serial) has a 65% gain in 7nm FinFET, compared to negligible gain ($< 1\%$) in 20nm planar. Single task gains drop rapidly as percent serial increases; above 10% serial the gain is $< 10\%$. This is because at nominal Vdd the task is already parallelized across 4.7 cores in 7nm (the power budget described in prior sections) and parallelizing across any additional cores leads to very marginal improvements in latency.

Our predicted transistor models show a small improvement in FO4 delay from 20nm to 7nm, translating to a clock frequency decrease of only roughly 5%. Therefore, improvements in single task performance must come from architectural or circuit techniques, such as voltage scaling, and not from generational process improvements. Also included in Figure 8.7 is the optimal Vdd when voltage scaling a single task to minimize latency. Nominal voltage in

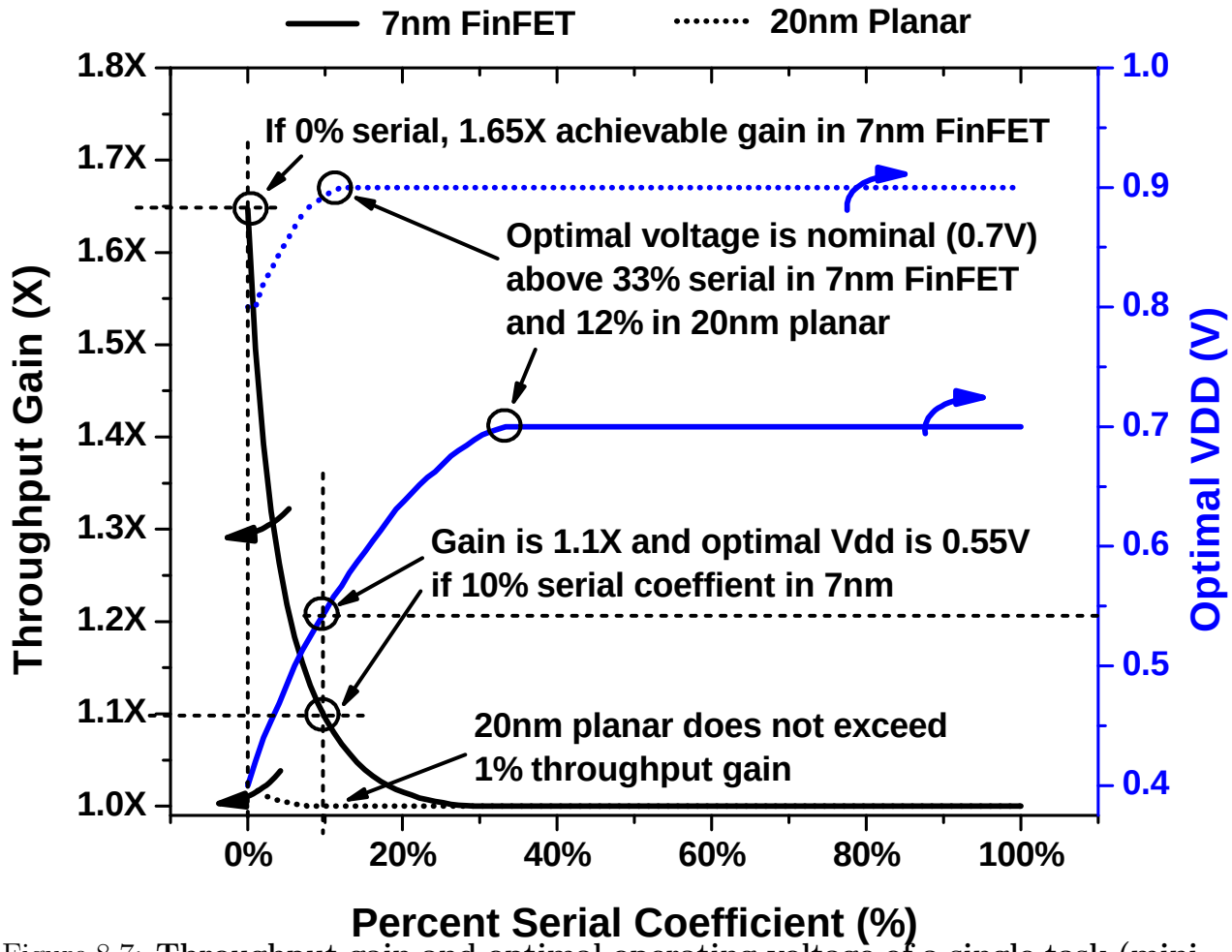


Figure 8.7: **Throughput gain and optimal operating voltage of a single task (minimize latency) in 7nm FinFET and 20nm Planar as percent serial is swept.** Nominal voltage is optimal above 33% serial.

20nm is 900 mV and in 7nm is 700 mV. At 0% serial, the optimal voltage is 400mV in 7nm and increases with higher percent serial until reaching nominal voltage at 33% serial. This suggests that fine-grained voltage and frequency adjustment is useful to optimize single task performance.

We used the above numbers as a guide for how to design the WAMI architecture, in order to understand how to best apply voltage scaling to system components and quantify achievable efficiency gains. Our system is composed of a mix of the above scaling scenarios, and how much area to dedicate to each core type is a free variable in our analysis, therefore no one scenario embodies the entire WAMI system. FinFET is well within the predicted dark silicon regime, where power density has increased to the point of limiting the majority of cores from operating simultaneously at full voltage and frequency. However, FinFET also offers substantial flexibility in architecture design than planar could not offer. Because of FinFET's improved circuit delay scaling and higher area density, designers may realize energy efficient heavily parallelized systems that work over a wide range of workloads. Traditionally, high single-task performance has been accomplished by running cores at maximum frequency and Vdd, yet in FinFET even single-task performance can be improved through voltage scaling.

8.4 System Analysis

8.4.1 Compute Units

The architecture we envision to handle the WAMI class of workloads is heterogeneous and targets the differing amounts and styles of parallelism that WAMI exhibits. For this study we examined WAMI, though the approach can be generally applied to any high performance system by understanding the computational requirements, goals, and ease of parallelism. Frames of an image in WAMI are assigned to small, independent *compute units*, Figure 8.8, consisting of specialized accelerators (e.g. FFTs), throughput accelerators units (e.g. GPGPUs) and general-purpose processors (CPUs), to handle different algorithm kernels from pixel-level to event-level in the WAMI pyramid. Threat-level machine learning detection is still done outside of the system. We envision a complete system will include many clusters

of compute units, with interconnects and shared memory (both off-chip DRAM and on-chip L2 cache), in a 3D stack such as proposed by Fick et al. [14]. However, in this chapter we focus on the achievable maximum energy efficiency (GFLOPS/W) of a compute unit, as interconnect and cache will directly contribute to power requirements and not computation, thereby decreasing the final GFLOPS/W.

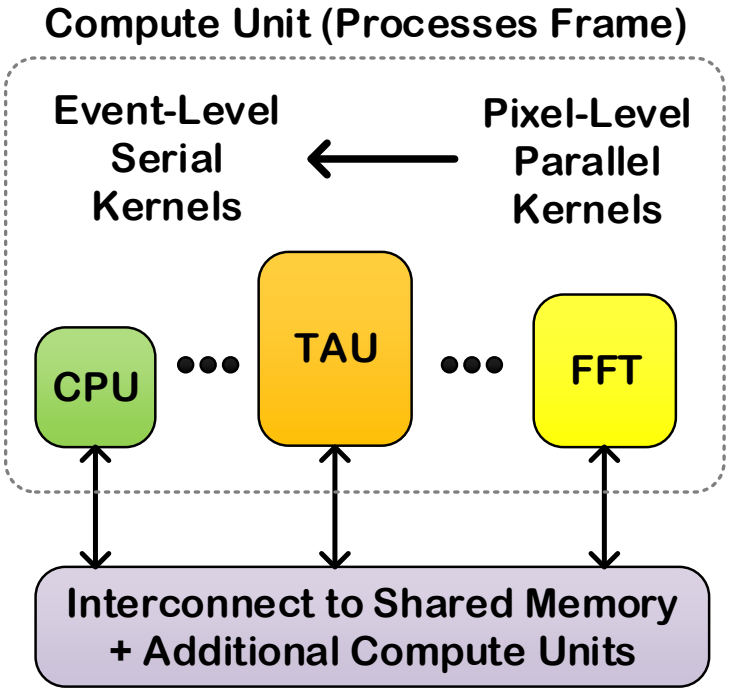


Figure 8.8: **Proposed high-performance compute unit to handle heterogeneous WAMI workloads.**

Specialized Accelerators (FFT)

WAMI pixel-level workloads include a deblurring algorithm which can be mapped to a dedicated Fast Fourier Transform (FFT) accelerator. While FFT accelerators have limited applications, and cannot be applied to event- or track-level processing, deblurring processes all pixels for each frame of an image, thus is large enough to justify a dedicated hardware unit. For our analysis we use the Jeon et al. design of a 16-bit 1024-point FFT accelerator with a low-power first-in first-out (FIFO) pipeline [76]. The accelerator specifically includes energy saving techniques, such as super pipelining, to improve low voltage operation. Published

results for their FFT accelerator are from a 65nm CMOS process, which we scale to 40nm CMOS for this work.

Throughput Accelerator Unit (TAU)

An important ingredient of our ongoing effort to achieve high power/performance for WAMI workloads is a programmable "throughput processor" that handles data in 32-wide integer words or 16-wide floating point words. We include this in our scaling analysis as an example unit to demonstrate how these type of workloads scale, but the energy efficiency gain results could be applied to similar vector processors.

The throughout accelerator unit (TAU) is a straightforward SIMD processor with 32/16 lanes. Each lane is a simple in-order pipeline with its own 32-entry register file. The ISA conforms to the ARM v8 definition, which makes it compatible with a wide range of programming models, and therefore leverages the entire ARM ecosystem. Simulations of the TAU unit, beyond this chapter's scope, show a maximum improvement 25× in peak GFLOPS/W over a complex out-of-order ARM Cortex-A57 core in the same technology for pixel-level WAMI kernels.

General-Purpose Processor

WAMI pixel-level workloads are highly parallel, but event- and track-level become increasingly serial and require more flow control. Therefore, our compute unit also includes a conventional general-purpose processor. For our example scaling, we estimated the power and performance of an in-order, dual-issue ARM Cortex-A53 processor, as we found event- and track-level WAMI kernels did not significantly benefit from out-of-order processors. Again, as with the previous units, these are shown for illustration purposes, and the general voltage scaling principles and methodologies could be applied to other processors and workloads.

8.4.2 Results

Mapping Voltage Scaling Scenarios

We map each level of WAMI to processor units, voltage scaling scenarios, and ease of parallelism, Table 8.3. For instance, at the bottom of the WAMI pyramid, pixel-level requires an FFT accelerator (for deblur algorithm), TAU accelerator (de-bayer algorithm, short-term change detection, and noise removal), and CPU (geographic registration). Through analysis of the kernel algorithms, we found that they are largely data parallel, with tight spatial locality, and thus parallelize well with a Amdahl serial coefficient of roughly 2%. As WAMI is intended for real-time analysis, latency for pixel-level algorithms is important, but is not absolutely critical to minimize as code size is small. Also included in pixel level is 'geographic registration,' which periodically aligns the image with GPS coordinates, requiring a general-purpose core. Geographic registration is not processed per frame, so quick latency is not important and the unconstrained latency scaling scenario applies to this algorithm.

For event- and track-level workloads, ongoing work is to characterize these WAMI kernels and refine estimates for how well they parallelize, thus the estimated percent serial may change after the kernels are fully understood. For event-level we show a range from 10% to 25% serial as parallization scaling for these kernels is not well understood.

Pre-Scaling Assumptions

For illustrating voltage scaling in the futuristic 7nm FinFET process, we begin by estimating the area, power, and performance of each compute unit component in 40nm when running at nominal supply voltage, shown in Table 8.4. These values were estimated through published data for a 65nm FFT [76] scaled to 40nm, publicly available specifications for an ARM Cortex-A53 CPUs, and internal predictions based on gate-level synthesis results of our throughput accelerator unit.

Efficiencies of the CPU and throughput accelerator are limited to 2.2 and 5.2 GFLOPS/W, respectively. However, the efficiency of our FFT is orders of magnitude higher than the CPU and TAU, because FFT hardware accelerators feature much higher efficiency than software FFTs running on conventional processors. Of course FFT accelerators have the disadvantage

Unit	Percent Serial	Goal
Event-Level Workload		
CPU Pessimistic	25%	Maximize Single-Task (Minimize Latency)
CPU Optimistic	10%	Maximize Single-Task (Minimize Latency)
Track-Level Workload		
CPU	5%	Maximize Single-Task (Minimize Latency)
Pixel-Level Workload		
CPU	2%	Maximize Many-Task (Unconstrained Latency)
TAU	2%	Balanced (Fixed Latency)
FFT	2%	Balanced (Fixed Latency)

Table 8.3: WAMI workload mapping to voltage scaling scenarios.

of being limited to running a single algorithm. Other applications may require a different set of processing units, with different efficiencies and area requirements, but the relative scaling from 40nm to our futuristic 7nm technology will be similar.

Block	Area (mm^2)	Efficiency (GFLOPS/W)	Power (mW)
CPU	2.2	2.2	330
TAU	12	5.2	1,200
FFT	3.3	170	7,300

Table 8.4: Pre-scale estimates of compute unit components running in 40nm at nominal voltage.

7nm Nominal Voltage Estimates

The compute components were scaled from 40nm planar VDD to 7nm FinFET, both targeting nominal operating voltage (1.1V in 40nm to 0.7V in 7nm), based on our technology scaling predictions from Section 8.3.2. Estimates are shown in Table 8.5. Area and power of each block decreased by 19× and 4.7×, respectively, while efficiency increased by 6.4×.

Block	Area (mm^2)	Efficiency (GFLOPS/W)	Power (mW)
CPU	0.11	14	71
TAU	0.62	33	258
FFT	0.17	1,100	1,570

Table 8.5: Components of compute unit in futuristic 7nm FinFET technology at nominal operating voltage.

7nm Voltage Scaling Estimates

The final area, efficiency, and power of computer unit components after voltage scaling to near-threshold in 7nm FinFET is shown in Table 8.6. Voltage scaling improvements are from Table 8.2. Area of blocks increased from nominal operating voltage because we assume blocks were replicated so that they match the power draw of nominal voltage operation, but with much higher energy efficiency and thus higher throughput (GFLOPS). In essence, voltage scaling allows us to achieve higher throughput with the same power budget by increasing area and lowering voltage.

The FFT and TAU units are able to achieve between $2.5 - 6.5\times$ efficiency gain, by operating between 460 mV to 240 mV, depending on the area limitations of the final system. Similarly, the pixel-level CPU is able to achieve $2.5 - 8.9\times$ gains at 460 mV to 180 mV because it does not have a strict latency requirement. Event- and track-level require latency to be minimized, and so gain up to 20% for track-level (at 480 mV) and up to 10% (at 540 mV) for event-level.

8.5 Related Work

The problem of increasing power density has been referred to as *dark silicon* by Esmaeilzadeh et al. [5], since in this regime it is not possible to run all cores on a processor simultaneously at maximum frequency and voltage. Taylor [15] further looks into issues of dark silicon as it impacts architecture, and discusses four solutions, including heterogeneous architectures and voltage scaling.

Block	Area (mm^2)	Efficiency (GFLOPS/W)	Power (mW)
Event-Level Workload			
CPU Pessimistic	0.11	14	71
CPU Optimistic	0.91	15	71
Track-Level Workload			
CPU	11	17	71
Pixel-Level Workload			
CPU	$2.1 - 170$	$35 - 125$	71
TAU	$11 - 200$	$83 - 212$	258
FFT	$3.1 - 55$	$2,707 - 6,931$	$1,570$

Table 8.6: 7nm estimates after voltage scaling and increasing number of units until power matches nominal Vdd.

Leveraging voltage scaling to regain energy is not new, and traditional dynamic voltage and frequency scaling (DVFS) is used extensively in processors. Early low-power subthreshold architectures were presented by Chandrakasan et al. [36] and Wang et al. [31]. More recent low-voltage work [6,12,48,50] shifts to using voltage scaling (near-threshold or NT) for overcoming dark silicon. A key distinction to prior work is that near-threshold is proposed under normal processor load, not just during periods of idleness. Multicore architectures, used to parallize workloads, were also included as parts of these works. Azizi et al. [77] shows that voltage scaling is an effective technique for trading off performance and power, and that a large energy-performance design space can be encompassed using a small core, large core, and voltage scaling. Pinckney et al. [10, 11] provided a methodical definition of near-threshold by defining it as the point where energy is minimized subject to a fixed latency constraint, and examined it across six planar technology nodes (180nm to 32nm). Circuit challenges and solutions in near-threshold were examined by Kaul et al. [23].

Most recently there has been increased interest in wide-voltage scaling circuits, that are meant to scale gracefully from low to high supply voltages. Intel has published a wide-voltage scaling processor [16], network-on-chip [17], and register file [8]. From academia, Centip3De [14] is a proposed 3D processor that can dynamically reconfigure between many low-Vdd processors or few high-Vdd processors, depending on workload requirements.

This work differs from prior work primarily by examining differences between FinFET and planar. Compared to prior NT studies, such as [10], we analyze across technologies using a set of models that have been consistently tuned with similar transistor threshold voltage types. We also combine the impact of wire loading, and mismatch variation, directly into the energy optimization, both of which are especially significant in recent technologies. Finally, this work proposes three definitions for voltage scaling scenarios, and includes area constraints, which [10] does not address.

8.6 Conclusions

Power and performance improvements in process technology has slowed and systems now, more than ever, need to be co-designed with circuit techniques, such as voltage scaling. We analyzed voltage scaling as it applies to a variety of workloads, by study a wide-area motion imaging (WAMI) application. Three specific voltage scaling scenarios are examined: (1) single-task throughput where latency is minimized; (2) many-task throughput where latency is fixed to that of nominal supply voltage; and (3) many-task throughput when latency is completely unconstrained. For each of these scenarios, we estimated efficiency gains with differing amounts of parallelism and area budgets. We then provided sample specifications of processing units for a WAMI system and show how they scale to near-threshold in 7nm FinFET. By leveraging voltage scaling we are able to achieve significantly higher throughput numbers, especially for easily parallelized kernels of the WAMI workloads. However, voltage scaling of mid-level single-task workloads also achieve gains, which was previously not possible in recent planar technologies.

We show that FinFET offers important advantages over planar CMOS technologies, namely less degradation in circuit performance at low voltages, which translates into sizable energy efficiency gains (Figure 8.9). Combined with the ability to pack more cores within the same die area, FinFET offers architects unique opportunities to design futuristic system that leverage voltage scaling for achieving high-performance, even when latency minimization is critical.

It is important to note that many additional factors would impact the efficiency and performance of a complete system, namely interconnects, caches, memory interfaces, and peripherals. Memory and interconnects voltage scale differently than core logic, since they are generally dominated by wire loads and leakage power. Of course, they will also impact the scalability of tasks and further co-optimization is needed. Our estimates show very high GFLOPS/W are possible at the pixel-level, for instance exceeding a 75 GFLOPS/W target for WAMI systems set by the DARPA-funded 'PERFECT' project, though further work is needed on system-level components to demonstrate this target is realizable. Nevertheless, understanding voltage scaling benefits and limitations is essential in designing futuristic architectures in our post-Dennard scaling world with FinFET.

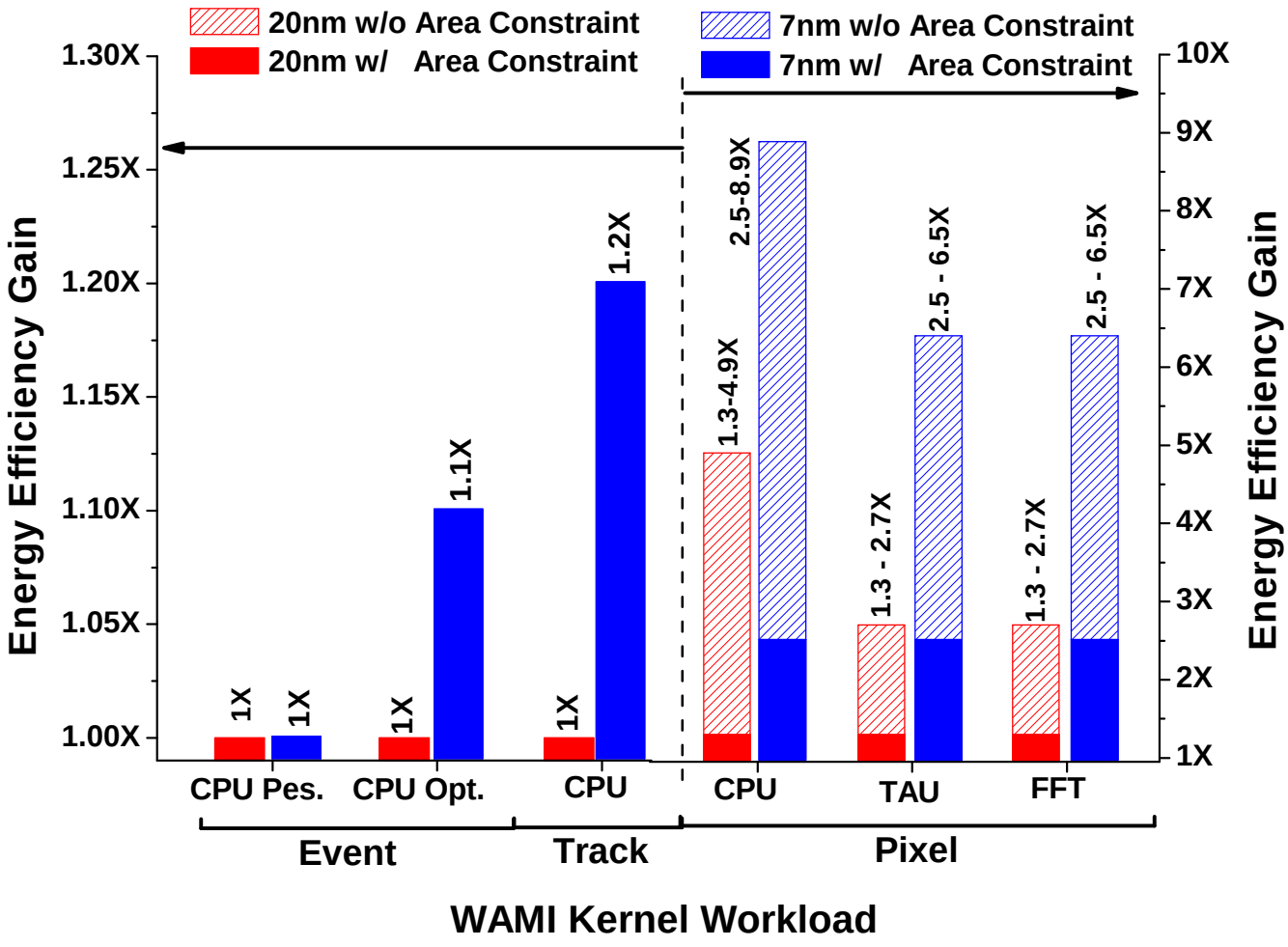


Figure 8.9: **Summary of energy efficiency gains of WAMI kernels in futuristic 7nm FinFET compared to 20nm planar.**

CHAPTER 9

Summary and Future Work

Supply voltage scaling has stagnated in recent technology nodes, leading to so-called *dark silicon* and increasing interest in near-threshold computing. This dissertation carefully examines near-threshold's past, present, and future, at device, circuits, and architectural levels.

9.1 Contributions

First, in chapters 2 and 3 we defined near-threshold computing, observing tradeoffs and trends across previous planar technology nodes (180nm to 32nm). Specifically, near-threshold is defined as the minimum energy point of a workload when accounting for parallelism overheads (algorithmic and architectural) to maintain a fixed latency of the workload running on a single core at nominal voltage. By constraining latency voltage scaling is evaluated for performance sensitive workloads, to better understand how close to threshold is practical for many workloads and then using this definition to examine trends across technology nodes. This iso-latency analysis is workload-dependent and parallelization overheads, arising from algorithmic and architectural sources, are assessed through system-level simulations of the SPLASH-2 benchmark suite [75]. A key finding is that, across the scientific benchmarks studied, the near-threshold region tracked roughly $200 - 400$ mV above threshold voltage. Across the SPLASH-2 benchmarks in 32nm, parallelism across 12 cores is needed on average.

Additionally, NT energy gain was decreasing from 8× in 180nm to 4× in 32nm, and therefore is becoming less effective as planar technologies reach end of life.

Chapters 4 and 5 continued with comparing fast voltage boosting techniques with heterogeneous cores, and proposed a fast voltage boosting architecture. The technique, called *Shortstop*, uses three external supplies to quickly raise the voltage of a core, within 10's of nanoseconds, without inducing supply droop. This is achieved by leveraging parasitic inductance of a package similar to a boost converter arrangement. The 28nm wirebonded demonstration chip was able to raise the voltage of a core 1.7× than PMOS headers and with $3.5-6\times$ less droop. Proposed modifications to Shortstop for flip-chip architectures is given in Chapter 6.

Finally, chapters 7 and 8 looked towards the future near-threshold, specifically evaluating the performance of near-threshold in recent and upcoming FinFET technologies. Chapter 7 examined effects of device characteristics on near-threshold and found that FinFET's superior channel characteristics, namely DIBL and subthreshold slope, contribute to much improved voltage scalability and near-threshold energy efficiency gain. Chapter 8 expanded analysis to consider three voltage scaling scenarios: minimizing lantecy, fixed latency, and unconstrained. A predicted 7nm FinFET technology has an energy efficiency gain of $2.6-8.9\times$, depending on area and latency constraints, a marked improvement from recent planar technology. Efficiency improvements of up to 20% are possible for tasks in which latency is minimized.

9.2 Future Work

Near-threshold holds promise for improving energy efficiency of modern processors as process scaling continues to slow. Increased transistor packing densities will enable advanced voltage scaling techniques. Initial near-threshold designs have shown promise, yet many obstacles remain before NTC can be widely adopted. Variability remains one of the biggest challenges for low-voltage operation but variation tolerant techniques, such as soft clocking and in-situ monitoring, can help mitigate these issues. Improved topologies for blocks that traditionally scale very poorly because of sensitivity to mismatch, such as SRAMs, demon-

strate that low-voltage operation is possible even under extreme variation [78–80]. Circuit techniques will aid in realizing efficient, voltage scalable systems, yet radical new CAD tools and methodologies must also be developed to alleviate needs on designers for quickly and practically implementing voltage scalable systems.

The near-threshold studies presented in chapters 2, 3, 7, and 8 are based on high-level models of voltage scaling, deriving near-threshold energy and performance from transistor models and architectural simulations. However, processors do not always port directly to newer technologies and are constrained by complex critical paths composed of both logic and wire delays. Additionally, circuit structures other than core logic paths (with high activity factors) may scale differently than the paths modeled in the above chapters. For instance, SRAM caches are composed of large array of bitcells with low activity factors, which favor high-threshold transistors because leakage is more dominant than in core logic. Thus, design and implementation of fully-realized architectures and systems utilizing voltage scaling may reveal caveats and limitations not exposed in this work.

Designs and proof-of-concepts of the Shortstop fast supply boosting technique were presented in chapters 5 and 6 but the studies have a few limitations. Neither wirebond nor flip-chip designs include a fully functioning processor, instead core modeling was limited to emulation of a processor through virtual rail capacitive and current loading. Also, integration of caches with multiple switched power domains was not explored, and may require level shifters and synchronizers costing power and area.

Another key problem with Shortstop is the need to accurately actuate all power switches, with little skew, to avoid causing short-circuit currents between power supplies. The enable signal for distributed on-chip power switches are conventionally daisy chained to reduce metallization utilization. However, daisy-chaining requires significantly more time to activate than if a buffered high-fanout network were used. In practice, this may also be a benefit as to not overtax the power supply network. Because Shortstop requires accurate actuation of PMOS and NMOS power switches, a low-skew clock-like network is needed for the switch enable networks, thereby requiring always-on buffering and higher metallization usage over core voltage areas. Centralizing power switches eases design and overhead of the switch enable network, but may degrade the power distribution network as centralized switches

may have higher IR drop and worse mutual inductance. Therefore, further work is needed to understand how to efficiently balance header and footer positioning between the two extremes of completely centralized and completely distributed power switches.

9.3 Related Publications

- Chapter 2: N. Pinckney, K. Sewell, R.G. Dreslinski, D. Fick, T. Mudge, D. Sylvester, and D. Blaauw. Assessing the performance limits of parallelized near-threshold computing. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 1143-1148, June 2012.

- Chapter 3: N. Pinckney, R.G. Dreslinski, K. Sewell, D. Fick, T. Mudge, D. Sylvester, and D. Blaauw. Limits of parallelism and boosting in dim silicon. *Micro, IEEE*, 33(5):30-37, Sept 2013.

- Chapter 4 is unpublished, but based off of a University of Michigan EECS 570 report from N. Pinckney and E. Lu.

- Chapter 5: N. Pinckney, M. Fojtik, B. Giridhar, D. Sylvester, and D. Blaauw. Shortstop: An on-chip fast supply boosting technique. In VLSI Circuits (VLSIC), 2013 Symposium on, pages C290-C291, June 2013.

- Chapter 6 is unpublished, but submission is planned once chip testing is complete.

- Chapter 7: N. Pinckney, L. Shifren, B. Cline, S. Sinha, S. Jeloka, R. Dreslinski, T. Mudge, D. Sylvester, and D. Blaauw. Near-Threshold in FinFET Technologies: Impact of Process on Voltage Scalability. *Pending submission.*

- Chapter 8: N. Pinckney, L. Shifren, B. Cline, S. Sinha, S. Jeloka, R. Higgins, G. Ray, J. Ballast, W. Snapp, C. Chakrabarti, T. Mudge, D. Sylvester, D. Blaauw, and R. Dreslinski. Near-Threshold Computing in FinFET Technologies: An Architectural Study. *Submitted for publication.*

- Sections of this thesis are also based on N. Pinckney, D. Blaauw, and D. Sylvester. Low-Power Near-Threshold Design: Techniques to Improve Energy Efficiency. In *Solid-State Circuits Magazine, IEEE*, 7(2):49-57, Spring 2015.

BIBLIOGRAPHY

[1] G.E. Moore. No exponential is forever: but "forever" can be delayed! [semiconductor industry]. In *Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC. 2003 IEEE International*, pages 20–23 vol.1, Feb 2003.

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

[3] R.H. Dennard, F.H. Gaensslen, V.L. Rideout, E. Bassous, and A.R. LeBlanc. Design of ion-implanted MOSFET's with very small physical dimensions. *Solid-State Circuits, IEEE Journal of*, 9(5):256–268, Oct 1974.

[4] S. Sawant, U. Desai, G. Shamanna, L. Sharma, M. Ranade, A. Agarwal, S. Dakshinamurthy, and R. Narayanan. A 32nm Westmere-EX Xeon enterprise processor. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2011 IEEE International*, pages 74–75, Feb 2011.

[5] Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, and Doug Burger. Dark silicon and the end of multicore scaling. In *Proceedings of the 38th Annual International Symposium on Computer Architecture*, ISCA '11, pages 365–376, New York, NY, USA, 2011. ACM.

[6] R.G. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge. Near-Threshold Computing: Reclaiming Moore's Law through energy efficient integrated circuits. *Proceedings of the IEEE*, 98(2):253–266, Feb 2010.

[7] L. Chang, D.J. Frank, R.K. Montoye, S.J. Koester, B.L. Ji, P.W. Coteus, R.H. Dennard, and W. Haensch. Practical strategies for power-efficient computing technologies. *Proceedings of the IEEE*, 98(2):215–236, Feb 2010.

[8] A. Agarwal, S. Hsu, S. Mathew, M. Anders, H. Kaul, F. Sheikh, and R. Krishnamurthy. A 32nm 8.3ghz 64-entry x 32b variation tolerant near-threshold voltage register file. In *VLSI Circuits (VLSIC), 2010 IEEE Symposium on*, pages 105–106, June 2010.

[9] R.G. Dreslinski. *Near Threshold Computing: From Single Core to Many-Core Energy Efficient Architectures*. PhD thesis, University of Michigan, 2011.

[10] N. Pinckney, K. Sewell, R.G. Dreslinski, D. Fick, T. Mudge, D. Sylvester, and D. Blaauw. Assessing the performance limits of parallelized near-threshold computing. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 1143–1148, June 2012.

[11] N. Pinckney, R.G. Dreslinski, K. Sewell, D. Fick, T. Mudge, D. Sylvester, and D. Blaauw. Limits of parallelism and boosting in dim silicon. *Micro, IEEE*, 33(5):30–37, Sept 2013.

[12] R.G. Dreslinski, Bo Zhai, T. Mudge, D. Blaauw, and D. Sylvester. An energy efficient parallel architecture using near threshold operation. In *Parallel Architecture and Compilation Techniques, 2007. PACT 2007. 16th International Conference on*, pages 175–188, Sept 2007.

[13] D. Fick, R.G. Dreslinski, B. Giridhar, Gyouho Kim, Sangwon Seo, M. Fojtik, S. Satpathy, Yoonmyung Lee, Daeyeon Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Sylvester, and D. Blaauw. Centip3de: A 3930dmips/w configurable near-threshold 3d stacked system with 64 arm cortex-m3 cores. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, pages 190–192, Feb 2012.

[14] D. Fick, R.G. Dreslinski, B. Giridhar, Gyouho Kim, Sangwon Seo, M. Fojtik, S. Satpathy, Yoonmyung Lee, Daeyeon Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Blaauw, and D. Sylvester. Centip3de: A cluster-based ntc architecture with 64 arm cortex-m3 cores in 3d stacked 130 nm cmos. *Solid-State Circuits, IEEE Journal of*, 48(1):104–117, Jan 2013.

[15] M.B. Taylor. Is dark silicon useful? harnessing the four horsemen of the coming dark silicon apocalypse. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 1131–1136, June 2012.

[16] S. Jain, S. Khare, S. Yada, V. Ambili, P. Salihundam, S. Ramani, S. Muthukumar, M. Srinivasan, A. Kumar, S.K. Gb, R. Ramanarayanan, V. Erraguntla, J. Howard, S. Vangal, S. Dighe, G. Ruhl, P. Aseron, H. Wilson, N. Borkar, V. De, and S. Borkar. A 280mv-to-1.2v wide-operating-range ia-32 processor in 32nm cmos. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, pages 66–68, Feb 2012.

[17] G. Chen, M.A. Anders, H. Kaul, S.K. Satpathy, S.K. Mathew, S.K. Hsu, A. Agarwal, R.K. Krishnamurthy, S. Borkar, and V. De. 16.1 a 340mv-to-0.9v 20.2tb/s source-synchronous hybrid packet/circuit-switched 16x16 network-on-chip in 22nm tri-gate cmos. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, pages 276–277, Feb 2014.

[18] M. Severson, K. Yuen, and Yang Du. Not so fast my friend: Is near-threshold computing the answer for power reduction of wireless devices? In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 1160–1162, June 2012.

[19] Fang-Li Yuan, Tsung-Han Yu, and D. Markovic. A 500mhz blind classification processor for cognitive radios in 40nm cmos. In *VLSI Circuits Digest of Technical Papers, 2014 Symposium on*, pages 1–2, June 2014.

[20] David Meisner and Thomas F. Wenisch. Dreamweaver: Architectural support for deep sleep. In *Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS XVII, pages 313–324, New York, NY, USA, 2012. ACM.

[21] T.N. Miller, Xiang Pan, R. Thomas, N. Sedaghati, and R. Teodorescu. Booster: Reactive core acceleration for mitigating the effects of process variation and application imbalance in low-voltage chips. In *High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on*, pages 1–12, Feb 2012.

[22] Ulya R. Karpuzcu, Krishna B. Kolluru, Nam Sung Kim, and Josep Torrellas. Variusntv: A microarchitectural model to capture the increased sensitivity of manycores to process variations at near-threshold voltages. In *Proceedings of the 2012 42Nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, DSN '12, pages 1–11, Washington, DC, USA, 2012. IEEE Computer Society.

[23] H. Kaul, M. Anders, S. Hsu, A. Agarwal, R. Krishnamurthy, and S. Borkar. Near-threshold voltage (ntv) design - opportunities and challenges. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 1149–1154, June 2012.

[24] Kyungseok Kim and V.D. Agrawal. Minimum energy cmos design with dual subthreshold supply and multiple logic-level gates. In *Quality Electronic Design (ISQED), 2011 12th International Symposium on*, pages 1–6, March 2011.

[25] Sadagopan Srinivasan, Li Zhao, Ramesh Illikkal, and Ravishankar Iyer. Efficient interaction between os and architecture in heterogeneous platforms. *SIGOPS Oper. Syst. Rev.*, 45(1):62–72, February 2011.

[26] S. Tavarageri and P. Sadayappan. A compiler analysis to determine useful cache size for energy efficiency. In *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International*, pages 923–930, May 2013.

[27] D.M. Harris, B. Keller, J. Karl, and S. Keller. A transregional model for near-threshold circuits with application to minimum-energy operation. In *Microelectronics (ICM), 2010 International Conference on*, pages 64–67, Dec 2010.

[28] Yasuhisa Omura, Azuma Yu, Yoshimasa Yoshioka, Kyota Fukuchi, and Daishi Ino. Proposal of preliminary device model and scaling scheme of cross-current tetrode soi mosfet aiming at low-energy circuit applications. *Solid-State Electronics*, 64(1):18–27, 2011.

[29] T. Sakurai and A.R. Newton. Alpha-power law mosfet model and its applications to cmos inverter delay and other formulas. *Solid-State Circuits, IEEE Journal of*, 25(2):584–594, Apr 1990.

[30] Mingoo Seok, S. Hanson, Yu-Shiang Lin, Zhiyoong Foo, Daeyeon Kim, Yoonmyung Lee, N. Liu, D. Sylvester, and D. Blaauw. The phoenix processor: A 30pw platform for sensor applications. In *VLSI Circuits, 2008 IEEE Symposium on*, pages 188–189, June 2008.

[31] A. Wang and A. Chandrakasan. A 180mV FFT processor using subthreshold circuit techniques. In *Solid-State Circuits Conference, 2004. Digest of Technical Papers. ISSCC. 2004 IEEE International*, pages 292–529 Vol.1, Feb 2004.

[32] Mingoo Seok, Dongsuk Jeon, C. Chakrabarti, D. Blaauw, and D. Sylvester. A 0.27v 30mhz 17.7nj/transform 1024-pt complex fft core with super-pipelining. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2011 IEEE International*, pages 342–344, Feb 2011.

[33] Dongsuk Jeon, M.B. Henry, Yejoong Kim, Inhee Lee, Zhengya Zhang, D. Blaauw, and D. Sylvester. An energy efficient full-frame feature extraction accelerator with shift-latch fifo in 28 nm cmos. *Solid-State Circuits, IEEE Journal of*, 49(5):1271–1284, May 2014.

[34] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008.

[35] K. DeHaven and J. Dietz. Controlled collapse chip connection (c4)-an enabling technology. In *Electronic Components and Technology Conference, 1994. Proceedings., 44th*, pages 1–6, May 1994.

[36] A.P. Chandrakasan, S. Sheng, and R.W. Brodersen. Low-power CMOS digital design. *Solid-State Circuits, IEEE Journal of*, 27(4):473–484, Apr 1992.

[37] Bo Zhai, D. Blaauw, D. Sylvester, and K. Flautner. Theoretical and practical limits of dynamic voltage scaling. In *Design Automation Conference, 2004. Proceedings. 41st*, pages 868–873, July 2004.

[38] Leyla Nazhandali, Bo Zhai, Javin Olson, Anna Reeves, Michael Minuth, Ryan Helfand, Sanjay Pant, Todd Austin, and David Blaauw. Energy optimization of subthreshold-voltage sensor network processors. In *Proceedings of the 32Nd Annual International Symposium on Computer Architecture*, ISCA '05, pages 197–207, Washington, DC, USA, 2005. IEEE Computer Society.

[39] Jan M. Rabaey, Anantha Chandrakasan, and Borivoje Nikolic. *Digital Integrated Circuits*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2008.

[40] S. Hanson, Bo Zhai, Mingoo Seok, B. Cline, K. Zhou, M. Singhal, M. Minuth, J. Olson, L. Nazhandali, T. Austin, D. Sylvester, and D. Blaauw. Performance and variability optimization strategies in a sub-200mV, 3.5pJ/inst, 11nW subthreshold processor. In *VLSI Circuits, 2007 IEEE Symposium on*, pages 152–153, June 2007.

[41] Gene M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*, AFIPS '67 (Spring), pages 483–485, New York, NY, USA, 1967. ACM.

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

[43] R. Gonzalez and M. Horowitz. Energy dissipation in general purpose microprocessors. *Solid-State Circuits, IEEE Journal of*, 31(9):1277–1284, Sep 1996.

[44] D.M. Brooks, P. Bose, S.E. Schuster, H. Jacobson, P.N. Kudva, A. Buyuktosunoglu, J.-D. Wellman, V. Zyuban, M. Gupta, and P.W. Cook. Power-aware microarchitecture: design and modeling challenges for next-generation microprocessors. *Micro, IEEE*, 20(6):26–44, Nov 2000.

[45] Intel Corporation. White Paper: Intel Turbo Boost Technology in Intel core microarchitecture (Nehalem) based processors, November 2008.

[46] Arun Raghavan, Yixin Luo, Anuj Chandawalla, Marios Papaefthymiou, Kevin P. Pipe, Thomas F. Wenisch, and Milo M. K. Martin. Computational sprinting. In *Proceedings of the 2012 IEEE 18th International Symposium on High-Performance Computer Architecture*, HPCA '12, pages 1–12, Washington, DC, USA, 2012. IEEE Computer Society.

[47] Ronald G Dreslinski, Bharan Giridhar, Nathaniel Pinckney, David Blaauw, Dennis Sylvester, and Trevor Mudge. reevaluating fast dual-voltage power rail switching circuitry. In *10th Ann. Workshop Duplicating, Deconstructing, and Debunking*, 2012.

[48] Bo Zhai, R.G. Dreslinski, D. Blaauw, T. Mudge, and D. Sylvester. Energy efficient near-threshold chip multi-processing. In *Low Power Electronics and Design (ISLPED), 2007 ACM/IEEE International Symposium on*, pages 32–37, Aug 2007.

[49] Ronald G. Dreslinski, David Fick, David Blaauw, Dennis Sylvester, and Trevor Mudge. Reconfigurable multicore server processors for low power operation. In *Proceedings of the 9th International Workshop on Embedded Computer Systems: Architectures, Modeling, and Simulation*, SAMOS '09, pages 247–254, Berlin, Heidelberg, 2009. Springer-Verlag.

[50] R Dreslinski, Michael Wieckowski, D Sylvester Blaauw, and T Mudge. Near threshold computing: Overcoming performance degradation from aggressive voltage scaling. In *Proc. Workshop Energy-Efficient Design*, pages 44–49, 2009.

[51] R.G. Dreslinski, G.K. Chen, T. Mudge, D. Blaauw, D. Sylvester, and K. Flautner. Reconfigurable energy efficient near threshold cache architectures. In *Microarchitecture, 2008. MICRO-41. 2008 41st IEEE/ACM International Symposium on*, pages 459–470, Nov 2008.

[52] Wonyoung Kim, M.S. Gupta, Gu-Yeon Wei, and D. Brooks. System level analysis of fast, per-core dvfs using on-chip switching regulators. In *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, pages 123–134, Feb 2008.

[53] N. Pinckney, M. Fojtik, B. Giridhar, D. Sylvester, and D. Blaauw. Shortstop: An on-chip fast supply boosting technique. In *VLSI Circuits (VLSIC), 2013 Symposium on*, pages C290–C291, June 2013.

[54] M. Aater Suleman, Onur Mutlu, Moinuddin K. Qureshi, and Yale N. Patt. Accelerating critical section execution with asymmetric multi-core architectures. In *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS XIV, pages 253–264, New York, NY, USA, 2009. ACM.

[55] Peter Greenhalgh. Big. little processing with arm cortex-a15 & cortex-a7. *ARM White Paper*, 2011.

[56] Jayanth Gummaraju, Laurent Morichetti, Michael Houston, Ben Sander, Benedict R. Gaster, and Bixia Zheng. Twin peaks: A software platform for heterogeneous computing on general-purpose and graphics processors. In *Proceedings of the 19th International Conference on Parallel Architectures and Compilation Techniques*, PACT '10, pages 205–216, New York, NY, USA, 2010. ACM.

[57] E. Rotem, A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann. Power-management architecture of the Intel microarchitecture code-named Sandy Bridge. *Micro, IEEE*, 32(2):20–27, March 2012.

[58] P. Hazucha, G. Schrom, J. Hahn, B.A. Bloechel, P. Hack, G.E. Dermer, S. Narendra, D. Gardner, T. Karnik, V. De, and S. Borkar. A 233-MHz 80%-87% efficient four-phase DC-DC converter utilizing air-core inductors on package. *Solid-State Circuits, IEEE Journal of*, 40(4):838–845, April 2005.

[59] Wonyoung Kim, D.M. Brooks, and Gu-Yeon Wei. A fully-integrated 3-level DC/DC converter for nanosecond-scale DVS with fast shunt regulation. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2011 IEEE International*, pages 268–270, Feb 2011.

[60] S. Pant and D. Blaauw. A charge-injection-based active-decoupling technique for inductive-supply-noise suppression. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 416–624, Feb 2008.

[61] Ron Ho, B. Amrutur, Ken Mai, B. Wilburn, T. Mori, and M. Horowitz. Applications of on-chip samplers for test and measurement of integrated circuits. In *VLSI Circuits, 1998. Digest of Technical Papers. 1998 Symposium on*, pages 138–139, June 1998.

[62] D. Hisamoto, Wen-Chin Lee, J. Kedzierski, H. Takeuchi, K. Asano, C. Kuo, Erik Anderson, Tsu-Jae King, J. Bokor, and Chenming Hu. Finfet-a self-aligned double-gate mosfet scalable to 20 nm. *Electron Devices, IEEE Transactions on*, 47(12):2320–2325, Dec 2000.

[63] Saurabh Sinha, Greg Yeric, Vikas Chandra, Brian Cline, and Yu Cao. Exploring sub-20nm finfet design with predictive technology models. In *Proceedings of the 49th Annual Design Automation Conference*, DAC '12, pages 283–288, New York, NY, USA, 2012. ACM.

[64] Shien-Yang Wu, C.Y. Lin, M.C. Chiang, J.J. Liaw, J.Y. Cheng, S.H. Yang, M. Liang, T. Miyashita, C.H. Tsai, B.C. Hsu, H.Y. Chen, T. Yamamoto, S.Y. Chang, V.S. Chang, C.H. Chang, J.H. Chen, H.F. Chen, K.C. Ting, Y.K. Wu, K.H. Pan, R.F. Tsui, C.H. Yao, P.R. Chang, H.M. Lien, T.L. Lee, H.M. Lee, W. Chang, T. Chang, R. Chen, M. Yeh, C.C. Chen, Y.H. Chiu, Y.H. Chen, H.C. Huang, Y.C. Lu, C.W. Chang, M.H. Tsai, C.C. Liu, K.S. Chen, C.C. Kuo, H.T. Lin, S.M. Jang, and Y. Ku. A 16nm finfet cmos technology for mobile soc and computing applications. In *Electron Devices Meeting (IEDM), 2013 IEEE International*, pages 9.1.1–9.1.4, Dec 2013.

[65] K.-I. Seo, B. Haran, D. Gupta, D. Guo, T. Standaert, R. Xie, H. Shang, E. Alptekin, D.-I. Bae, G. Bae, C. Boye, H. Cai, D. Chanemougame, R. Chao, K. Cheng, J. Cho, K. Choi, B. Hamieh, J.G. Hong, T. Hook, L. Jang, J. Jung, R. Jung, D. Lee, B. Lherron, R. Kambhampati, B. Kim, H. Kim, K. Kim, T.S. Kim, S.-B. Ko, F.L. Lie, D. Liu, H. Mallela, E. Mclellan, S. Mehta, P. Montanini, M. Mottura, J. Nam, S. Nam, F. Nelson, I. Ok, C. Park, Y. Park, A. Paul, C. Prindle, R. Ramachandran, M. Sankarapandian, V. Sardesai, A. Scholze, S.-C. Seo, J. Shearer, R. Southwick, R. Sreenivasan, S. Stieg, J. Strane, X. Sun, M.G. Sung, C. Surisetty, G. Tsutsui, N. Tripathi, R. Vega, C. Waskiewicz, M. Weybright, C.-C. Yeh, H. Bu, S. Burns, D. Canaperi, M. Celik, M. Colburn, H. Jagannathan, S. Kanakasabaphthy, W. Kleemeier, L. Liebmann, D. McHerron, P. Oldiges, V. Paruchuri, T. Spooner, J. Stathis, R. Divakaruni, T. Gow, J. Iacoponi, J. Jenq, R. Sampson, and M. Khare. A 10nm platform technology for low power and high performance application featuring finfet devices with multi workfunction gate stack on bulk and soi. In *VLSI Technology (VLSI-Technology): Digest of Technical Papers, 2014 Symposium on*, pages 1–2, June 2014.

[66] M.G. Bardon, P. Raghavan, G. Eneman, P. Schuddinck, M. Dehan, A. Mercha, A. Thean, D. Verkest, and A. Steegen. Group iv channels for 7nm finfets: Performance for socs power and speed metrics. In *VLSI Technology (VLSI-Technology): Digest of Technical Papers, 2014 Symposium on*, pages 1–2, June 2014.

[67] C. Bienia, S. Kumar, and K. Li. Parsec vs. splash-2: A quantitative comparison of two multithreaded benchmark suites on chip-multiprocessors. In *Workload Characterization, 2008. IISWC 2008. IEEE International Symposium on*, pages 47–56, Sept 2008.

[68] Shien-Yang Wu, J.J. Liaw, C.Y. Lin, M.C. Chiang, C.K. Yang, J.Y. Cheng, M.H. Tsai, M.Y. Liu, P.H. Wu, C.H. Chang, L.C. Hu, C.I. Lin, H.F. Chen, S.Y. Chang, S.H. Wang, P.Y. Tong, Y.L. Hsieh, K.H. Pan, C.H. Hsieh, C.H. Chen, C.H. Yao, C.C. Chen, T.L. Lee, C.W. Chang, H.J. Lin, S.C. Chen, J.H. Shieh, M.H. Tsai, S.M. Jang, K.S. Chen, Y. Ku, Y.C. See, and W.J. Lo. A highly manufacturable 28nm cmos low power platform technology with fully functional 64mb sram using dual/tripe gate oxide process. In *VLSI Technology, 2009 Symposium on*, pages 210–211, June 2009.

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

[70] T. Sakurai. Approximation of wiring delay in mosfet lsi. *Solid-State Circuits, IEEE Journal of*, 18(4):418–426, Aug 1983.

[71] S. Das, C. Tokunaga, S. Pant, Wei-Hsiang Ma, S. Kalaiselvan, K. Lai, D.M. Bull, and D.T. Blaauw. Razorii: In situ error detection and correction for pvt and ser tolerance. *Solid-State Circuits, IEEE Journal of*, 44(1):32–48, Jan 2009.

[72] Yejoong Kim, Wanyeong Jung, Inhee Lee, Qing Dong, M. Henry, D. Sylvester, and D. Blaauw. 27.8 a static contention-free single-phase-clocked 24t flip-flop in 45nm for low-power applications. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, pages 466–467, Feb 2014.

[73] Mingoo Seok, D. Blaauw, and D. Sylvester. Robust clock network design methodology for ultra-low voltage operations. *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, 1(2):120–130, June 2011.

[74] Ronald G Dreslinski, Qi Zheng, Robert P Higgins, Johann Hauswald, David Blaauw, Trevor Mudge, Chaitali Chakrabarti, Jon Ballast, and Warren Snapp. An architecture for low-power high-performance embedded computing. In *GOMACTech Conference (GOMAC), Thirty ninth Annual*, pages 423–426, Apr 2014.

[75] Steven Cameron Woo, Moriyoshi Ohara, Evan Torrie, Jaswinder Pal Singh, and Anoop Gupta. The splash-2 programs: Characterization and methodological considerations. In *Proceedings of the 22Nd Annual International Symposium on Computer Architecture*, ISCA '95, pages 24–36, New York, NY, USA, 1995. ACM.

[76] Dongsuk Jeon, Mingoo Seok, C. Chakrabarti, D. Blaauw, and D. Sylvester. A super-pipelined energy efficient subthreshold 240 ms/s fft core in 65 nm cmos. *Solid-State Circuits, IEEE Journal of*, 47(1):23–34, Jan 2012.

[77] Omid Azizi, Aqeel Mahesri, Benjamin C. Lee, Sanjay J. Patel, and Mark Horowitz. Energy-performance tradeoffs in processor architecture and circuit design: A marginal cost analysis. In *Proceedings of the 37th Annual International Symposium on Computer Architecture*, ISCA '10, pages 26–36, New York, NY, USA, 2010. ACM.

[78] Bo Zhai, S. Hanson, D. Blaauw, and D. Sylvester. A variation-tolerant sub-200 mv 6-t subthreshold sram. *Solid-State Circuits, IEEE Journal of*, 43(10):2338–2348, Oct 2008.

[79] Ik Joon Chang, Jae-Joon Kim, Sang Phill Park, and K. Roy. A 32 kb 10t sub-threshold sram array with bit-interleaving and differential read scheme in 90 nm cmos. *Solid-State Circuits, IEEE Journal of*, 44(2):650–658, Feb 2009.

[80] G. Chen, D. Sylvester, D. Blaauw, and T. Mudge. Yield-driven near-threshold sram design. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 18(11):1590–1598, Nov 2010.