OPTIMAL POWER/PERFORMANCE PIPELINING FOR ERROR RESILIENT PROCESSORS

BY

NICOLAS ZEA

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Adviser:

Assistant Professor Rakesh Kumar

ABSTRACT

Timing speculation has been proposed as a technique for maximizing energy efficiency of processors with minimal loss in performance. A typical implementation of timing speculation involves relaxing the timing constraints of a processor to a point where errors are possible but rare, and employing an error recovery mechanism to ensure correct functionality. This allows significant energy efficiency gains with a small recovery overhead.

Previous work on timing speculation has either explored the benefits of customizing the design methodology for a particular error resilience mechanism or attempted to understand the benefits from error resilience for a particular resiliency mechanism. There is no work, to the best of our knowledge, that attempts to understand the benefits of co-optimizing microarchitecture and error resilience.

In this thesis, we present the first study on co-optimizing a processor pipeline and an error resilience mechanism. We develop an analytical model that relates the benefits from error resiliency to the depth of the pipeline as well as its circuit structure. The model is then used to determine the optimal pipeline depth for different energy efficiency metrics for different error resilience overheads.

Our results demonstrate that several interesting relationships exist between error resilience and pipeline structure. For example, we show that there are significant energy efficiency benefits to pipelining an architecture for an error resiliency mechanism versus error resiliency-agnostic pipelining. As another example, we show that benefits from error resiliency are greater for short pipelines than long pipelines. We also confirm that the benefits from error resiliency are higher when the circuit structure is such that the error rate increases slowly on reducing input voltage versus a circuit optimized for power where a slack wall exists at the nominal operating point. We quantify the difference in benefits from error resiliency for irregular versus

regular workloads and show that benefits from error resiliency are higher for irregular workloads. Finally, we discuss the relationship between frequency and voltage-based timing speculation schemes, and draw conclusions about when is best to employ each. Our analytical results were validated using a cycle-accurate simulation-based model.

To my parents, Angela Prior and Horacio Zea, for giving me the opportunities and abilities to achieve my dreams.

ACKNOWLEDGMENTS

This thesis could not have been accomplished without the patience and guidance of my adviser, Professor Rakesh Kumar. I would like to profoundly thank him for his patience, advice, and friendship, all of which have been invaluable throughout my graduate career.

Many thanks go to John Sartori for his aid and contributions to both this thesis and my sanity. Without his help and patience, this would not have been possible, and would have been considerably more dull.

Lastly, I'd like to extend my heartfelt gratitude and appreciation to Megan Kniepkamp and James Ivaska, for their understanding, patience, and sense of humor, and to Dr. Varsha Ramoutar, for providing the support and drive I needed to complete this thesis.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Increasing power density due to Moore's Law and a push towards mobility have made energy the primary design constraint for computing devices. Several power reduction techniques have been proposed, but their effectiveness is often hindered due to manufacturing and environmental variations, which force system designers to design for the worst case.

There has been a flurry of recent work on designing for better-than-worst-case (BTWC). BTWC techniques design for correct operation under nominal conditions and provide support for a software or hardware-based error resilience technique to detect and correct errors due to variations. One promising BTWC technique is timing speculation [1–3]. Timing speculation advocates operating the processor at an aggressive voltage (*voltage overscaling*) or frequency (*frequency overscaling*) which may cause timing errors. A mechanism such as Razor [1] may be provided to detect and correct these timing errors.

Previous work on timing speculation has either explored the benefits of customizing the design methodology for a particular error resilience mechanism [2–5] or attempted to understand the benefits from error resilience for a particular processor design [1, 6–9]. There is no work, to the best of our knowledge, that attempts to understand the benefits of co-optimizing microarchitecture and error resilience.

In this thesis, we present the first study on co-optimizing a processor pipeline and an error resilience mechanism. We develop an analytical model that relates the benefits from error resiliency to the depth of the pipeline as well as its circuit structure. Our model builds upon Hartstein and Puzak's model for optimizing pipeline depth considering both power and performance [10]. We have added a model for either voltage or frequency overscaling to enhance energy efficiency in conjunction with various relationships to timing error rates and error recovery mechanisms. The overhead of er-

ror recovery may either be fixed or depend on the length of the processor's pipeline. The new model allows us to optimize both the pipeline depth and operating voltage/frequency for a given error recovery mechanism.

We further create a model for comparing voltage and frequency overscaling's effects on energy efficiency. By analyzing how the two timing speculation schemes affect path slack differently, we are able to understand their relative benefits for a particular energy efficiency metric and determine which is most sensitive to architectural and workload changes.

Our results demonstrate that several interesting relationships exist between error resilience and pipeline structure. We show that not only can the optimal pipeline depth be significantly different when error resilience is taken into account, but that different error resilience mechanisms (as reflected by their recovery overhead) impact the architecture differently. We additionally explore the importance of other architectural and workload parameters on the effects of error resilient designs. We show that frequency-based timing speculation schemes can yield greater energy efficiency gains from timing speculation, but are limited by workload memory sensitivity. Finally, we demonstrate that optimizing an architecture without considering error resiliency results in sub-optimal energy efficiency benefits. We explain why this is the case and show that optimal architectures should take error resilience mechanisms into consideration.

The remainder of this thesis is organized as follows. We survey the related work in Chapter 2. Chapters 3 and 4 discuss the benefits of optimizing the pipeline depth when employing voltage and frequency overscaling techniques, respectively. Chapter 5 compares the benefits of optimizing a pipeline for voltage overscaling versus corresponding benefits when the pipeline is optimized for frequency overscaling. Chapter 6 discusses the limitations of this study. We also discuss future work and conclude.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Timing Speculation

Due to the disparity between lithography technology and physical device scaling, variations in circuits are often outside of the control of designers, resulting in inevitable errors. To counter this variability, along with variability from environmental conditions, voltage guardbands are typically employed. The act of operating either at voltages below the guardbands or frequencies higher than those permitted by the guardband is considered timing speculation. This technique is gaining momentum as a viable method for increasing energy efficiency. Works including [1–3, 11] all employ timing speculation for the purpose of either improving performance or reducing power. However, there is no work, to the best of our knowledge, that attempts to understand the benefits of co-optimizing microarchitecture and error resilience.

The work closest to ours is by de Kruijf et al. [11], who develop a performance/power model for understanding the effectiveness of timing speculation for different process technologies, power designs, and error recovery techniques. Their work is focused on understanding the efficiency of timing speculation for a *given* architecture. We attempt to understand the benefits of co-optimizing a processor's pipeline with circuit structure and error resilience strategy.

2.2 Error Resiliency Techniques

Timing speculation requires a software or hardware-based error resilience mechanism to detect and correct errors. A number of techniques have been proposed for dealing with overscaling-induced timing errors. One class of

techniques employs checker logic in hardware to detect and correct errors [1, 9, 12]. For example, [1] employs *shadow latches* along vulnerable paths that must be guaranteed to receive a correct, albeit delayed, value, which can be compared against the value latched at the clock period. Another technique involves coupling three latches together in a triple-latch technique [9], with the third latch guaranteed to have the correct value while the first two determine how close to the critical point the circuit is operating.

Alternatively, separate processors can be employed to ensure correctness or enhance performance [7, 8]. Other techniques, such as those described in [6] and [13], allow the propagation of errors up to the software, where they can either be ignored if the application is robust enough, or corrected through software error correction.

2.3 Designing for Error Resiliency

Previous research has also considered design methodologies for error resilient processors. [3–5] attempt to use cell sizing to redistribute path slack to those circuits most affected by voltage scaling, thereby creating a more gradual increase in errors as voltage is reduced. They employ voltage overscaling with the goal of reducing power. [2] targets the most vulnerable timing paths in a similar fashion, but employs either forward biasing or tighter timing constraints, and focuses on enhancing performance.

Our work focuses not on the design methodology, but the architecture itself: namely, the pipeline depth. Furthermore, we consider both voltage and frequency overscaling, in the context of an arbitrary energy-efficiency metric.

2.4 Optimal Pipelining

The subject of determining the optimal pipeline length for an architecture has been studied significantly. Hrishikesh et al. [14] determined that the optimal logic depth per pipeline stage is 6 to 8 FO4 delays when considering only performance. Hartstein and Puzak built on power models from Srinivasan et al. [15] to develop an analytical model that determines the optimal

pipeline depth for metrics that consider both power and performance [10]. We build on Harstein and Puzak's model to develop a model that determines the optimal pipeline depth for processors that tolerate voltage and frequency overscaling-induced timing errors.

2.5 Low Power Designs

The subject of low power designs has been discussed for decades. [16] discusses some fundamental techniques for low power architecture and design. [17] discusses scheduling issues for low power. [18] discusses the impact of computer-aided design on low power processors. Our work maximizes energy efficiency by co-optimizing architecture with an error resilience mechanism.

CHAPTER 3

OPTIMAL PIPELINING FOR VOLTAGE OVERSCALING

Voltage overscaling allows energy efficiency benefits without much loss in throughput [19]. This chapter discusses co-optimizing processor pipeline with an error resilience mechanism for voltage overscaling induced timing errors.

3.1 Theory

We first present the baseline analytical model from which our work is derived. From this, we move on to our enhancements which account for error resilient designs.

3.1.1 Baseline

First, we consider the analytical model developed by Hartstein and Puzak [10] for optimizing a processor pipeline for a metric that considers power and performance ($\text{Metric}_{P/P}$):

$$Metric_{P/P} = 1/((T/N_I)^m P_T) \tag{3.1}$$

This is composed of the following two parts:

$$T/N_I = 1/(fa) + (\gamma_h N_h p)/f \tag{3.2}$$

and

$$P_T = (F_{cg} f P_d + P_l) N_L p^\eta \tag{3.3}$$

where m in Equation (3.1) is the exponential weighting for delay in the energy efficiency metric; T/N_I, defined in Equation (3.2), is the runtime of a benchmark weighed by the number of instructions (i.e. cycles per instruction

or CPI); and P_T, defined in Equation (3.3), is the average power consumption during the benchmark. Following the example of [10], we use $m = 3$, representing an energy-delay^2 metric, for our studies unless mentioned otherwise.

Common to both Equation (3.2) and Equation (3.3) are the p and f variables. The term p represents the pipeline depth of the processor and is varied in the optimization process; f is defined as the operating frequency, and is derived from

$$f = 1/(t_o + t_p/p) \tag{3.4}$$

where t_o is the latch delay employed in the system and t_p is the logic delay of the full pipeline.

The CPI formula, Equation (3.2), is composed of two parts, the busy time and the non-busy time. The busy time is simply the clock period weighted by the superscalar width factor, a, representing the average amount of instruction level parallelism (ILP) per cycle for a workload. The non-busy time uses a single variable, N_h, defined as the fraction of all instructions which might cause hazards. These hazards include mispredictions, structural hazards, data dependence stalls, etc. The term γ_h is then defined as the average performance penalty factor for hazards. It represents an average of the fraction of pipeline stages which must stall/bubble when a hazard occurs. Because it is a fraction of the pipeline stages, the non-busy time is weighted by p in addition to the clock period $1/f$.

The power equation, derived from Srinivasan et al.'s work [15] includes three components: dynamic power, leakage power, and a latch growth factor. Dynamic power is represented by P_d, the average dynamic energy/cycle per latch (note that these units are not in watts), weighted by the clock gating factor, f_{cg}, and the frequency. The clock gating factor is 1 when no clock gating is performed, and greater than 1 for different degrees of clock gating. A f_{cg} value of 1.3 is considered to be an aggressively clock gated design. P_l represents the average leakage power per latch in energy/second or watts. Because both these power values are per latch, they are weighted by the average number of latches per stage, N_L. The latch growth component of the system accounts for the superlinear growth in latches as pipeline depth increases, argued by Srinivasan et al. in [15]. This is represented by η, the latch growth factor.

By accounting for workload variation in hazards and ILP and architectural

variation in delays and power consumption, Equation (3.1) is able to optimize the number of pipeline stages for particular architectures based on an energy-delay metric.

3.1.2 Modeling Voltage Overscaling and Error Resilience

The key to modeling voltage overscaling and error resilience is accounting for the power and reliability impact of overscaling and the performance impact of error recovery. The magnitude of voltage overscaling directly determines the power savings and the timing error rate. The error rate, given an error recovery mechanism and the associated recovery cost, determines the performance penalty.

As argued in [1], in order to make an error recovery mechanism feasible to design, the error recovery cost will normally be proportional to the pipeline depth. In this scenario, the performance cost of error recovery can be modeled as

$$T_{err}/N_I = \gamma_e e p (T_o/N_I) \tag{3.5}$$

where γ_e is the average number of pipeline stages delayed by error recovery, p is the number of pipestages for that design, e is the average number of errors per cycle (the error rate), and T_o/N_I is the CPI of the system described in Equation (3.2). When the cost of error recovery is independent of the total number of pipestages, the performance cost of error recovery can be modeled as

$$T_{err}/N_I = \gamma_e e c (T_o/N_I) \tag{3.6}$$

where c is a constant. The overhead of error recovery can then be added to the CPI in Equation (3.2). The new performance (CPI) equation that accounts for the overhead of error recovery is

$$T/N_I = 1/(fa) + (\gamma_h N_h p)/f + T_{err}/N_I \tag{3.7}$$

To model the impact of voltage overscaling on processor power and reliability, we introduce a voltage overscaling factor, F_{scale}. We scale the dynamic power quadratically with the normalized voltage. Leakage power is scaled

linearly with the normalized voltage. Our new power model is as follows:

$$P_T = (f_{cg} f P_d {F_{scale}}^2 + P_l F_{scale}) N_L p^{\eta} \tag{3.8}$$

For modeling the relationship between error rate and voltage overscaling, we assume that a slack wall exists at which the error rate approaches 100% [4, 20]. The relationship between voltage overscaling and error rate can then be modeled by

$$e = min(1, ((1 - F_{scale})/(1 - F_{wall}))^w) \tag{3.9}$$

where e is the error rate, F_{scale} is the voltage overscaling factor ($0 \leq F_{scale} \leq 1$ with $F_{scale} = 1$ corresponding to the nominal voltage), F_{wall} is the normalized voltage at which the slack wall is reached ($0 \leq v_{wall} \leq 1$), and w is the exponential relating how steeply the errors increase on overscaling. A small w value corresponds to a relatively smooth increase in error rate as voltage is reduced.

Note that v_{wall} is only constant with regard to a particular pipeline depth. The amount of available voltage slack actually decreases as the length of the pipeline is increased. Figure 3.1 illustrates this effect. We model this dependence of F_{wall} on the length of the pipeline using the following equation:

$$F_{wall} = 1 - (1 - F_{base_w}) * (p_b/p)^k \tag{3.10}$$

where F_{base_w} is the normalized voltage at which the slack wall is reached for the base pipeline depth; p_b is the base pipeline depth (we assume the traditional 5 stage pipeline as the baseline in our experiments), k controls how quickly the error rate grows with the number of pipestages, and p is the current pipeline depth. Effectively, as the pipeline depth exceeds the base pipeline depth, the available voltage slack decreases. Note that the equation assumes that all timing paths can be equally divided when pipelining (all previous works on optimal pipelining depth make the same assumption).

3.2 Methodology

Our analytical model requires data on dynamic and static power per latch (note that we make the assumption that all power is consumed in latches,

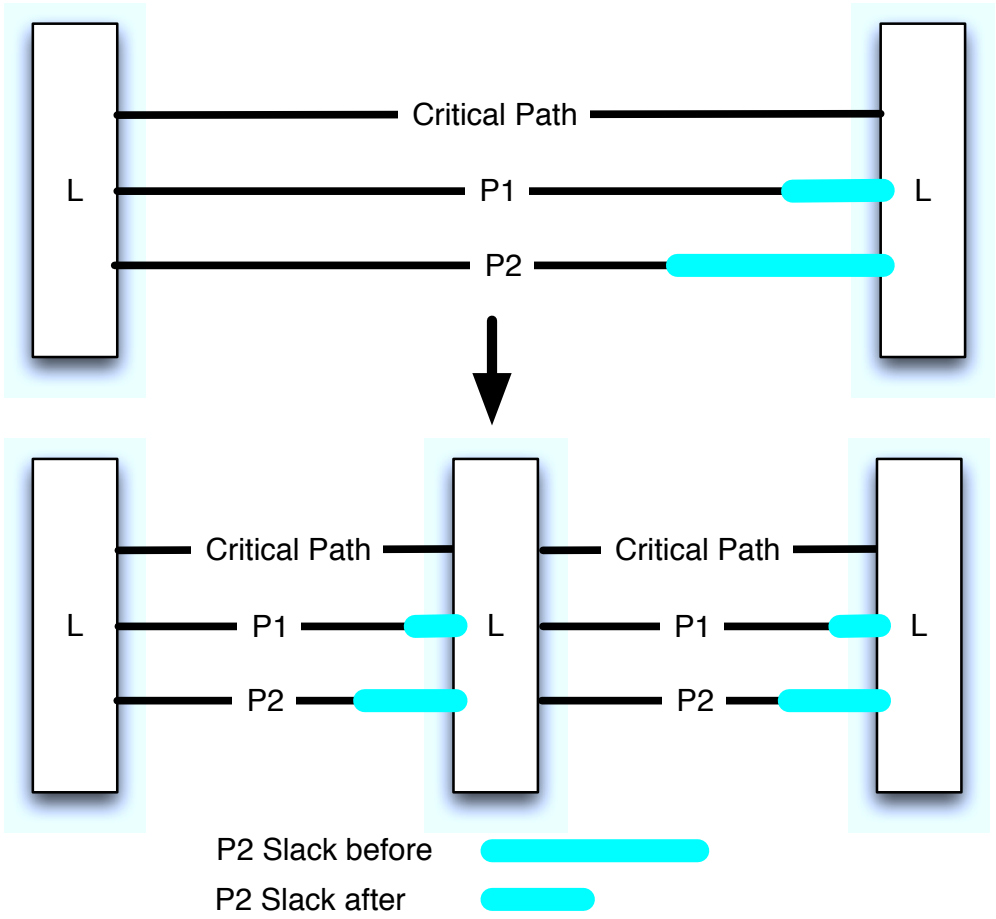


Figure 3.1: The effect of pipelining on the slack of a design. When a logic stage is pipelined, the absolute length of the timing paths, and therefore the amount of slack per stage, is reduced. This causes more errors for a given absolute reduction in voltage.

the same assumption made in all previous work on optimal pipelining). Because we do not have actual gate-level data available to use as parameters in our model, we rely on data from an architecture-level power simulator (Wattch [21]) that is coupled with a cycle-accurate processor simulator (SMTSIM [22]) simulating an Alpha core. The dynamic power estimates are derived as an average over SPEC2000 benchmarks [23] (listed in Table 3.3 on page 13) when run for 100 million instructions after fast-forwarding them to the Early Simpoints [24]. We assume that leakage power is 30% of the total power at the nominal voltage. We do not consider clock gating, and we assume $\eta = 1.3$, based on [15]. Our power formula, therefore, is

$$P_T = (f(P_{sim}/f_{sim}){F_{scale}}^2 + (.3P_{sim}/.7)F_{scale})p^{1.3} \tag{3.11}$$

where P_{sim} is the dynamic power reported by the simulator at the nominal voltage and f_{sim} is the frequency at which that power was reported.

For validating our analytical model and confirming the conclusions we drew from the analytical model, we performed further experiments using a modified version of SMTSIM [22] coupled with power estimates from Wattch [21]. Our modifications allowed us to vary the frequency and operating voltage (V_{dd}), insert errors at a particular rate per cycle, and control the error recovery penalty. To model error recovery, we simply penalize the system for $\gamma_e \times p$ cycles (or $\gamma_e \times c$ cycles when the recovery penalty is fixed).

To change the length of the simulated pipeline, we added extra stages to the front end of the simulated processor. This ensures that the increased length of the pipeline affects the overhead of hazards. In addition, Wattch does not account for power growth due to pipeline depth. We assumed the same latch growth exponent of $\eta = 1.3$ as in our analytical model, and scaled our power accordingly. As the pipeline depth increased, we scaled the operating frequency based on Equation (3.4), while keeping the memory latency constant. Our validation experiments were run using the SPEC2000 binaries. We fast-forwarded to the Early SimPoint [24] of each benchmark before beginning error injection simulations.

Table 3.1 presents our SMTSIM settings, while Table 3.2 presents our power settings for Wattch. Lastly, Table 3.3 describes the benchmarks we used in our simulations. The benchmarks were chosen randomly, with five floating point and three integer benchmarks. The Base IPC is the IPC of the

Table 3.1: *SMTSIM Parameters*

Core	
Number of instructions simulated	100 Million
Instruction order	in-order
Number of threads	Single Threaded
Number of stages	8+
L1 Split I/D Cache	
Size	32 KB
Assoc	4-Way
Miss Penalty	8 cycles
L2 Cache	
Size	2 MB
Assoc	4-way
Miss penalty	40 cycles
L3 cache	
Size	4 MB
Assoc	4-way
Miss penalty (to memory)	255 ps

benchmark when simulated on the minimal 8 stage pipeline without support for errors (no timing speculation).

3.3 Results and Analysis

In this section, we analyze the relationship between the benefits from error resilience and pipeline, circuit, and workload characteristics. We also present results from our validation experiments.

Table 3.2: *Wattch Parameters*

Wattch Parameter	**Value**
Process Technology	65 nm
Vdd (nominal)	1.5 V
Vth	0.7 V
Dynamic Power vs. Voltage relationship	$v^2 f$

Table 3.3: *SPEC2000 Benchmarks Employed*

Benchmark	**Description**	**Base IPC**
SPECFP		
applu	Parabolic / Elliptic Partial Differential Equations	0.307
art	Image Recognition / Neural Networks	0.44
equake	Seismic Wave Propagation Simulation	0.331
swim	Shallow Water Modeling	0.302
wupwise	Physics/Quantum Chromodynamics	0.649
SPECINT		
bzip	Compression	0.837
crafty	Game Playing: Chess	0.719
vpr	FPGA Circuit Placement and Routing	0.293

3.3.1 Exploring the Interaction between Error Resilience, Pipelining, Circuit Structure, and the Metric for Energy Efficiency

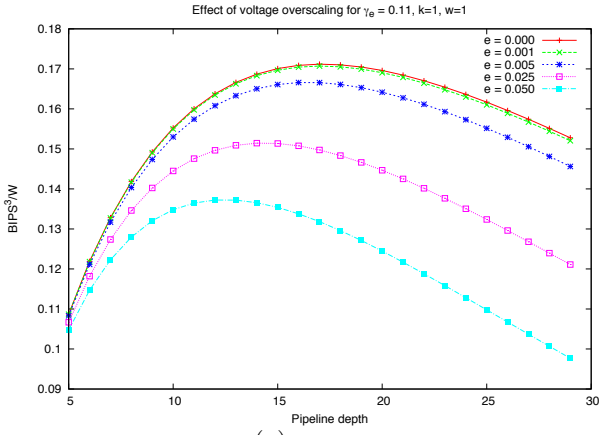
We begin by exploring the benefits from error resilience when voltage is overscaled to allow errors which are then assumed to be tolerated using a suitable error tolerance mechanism (the recovery penalty is considered while evaluating energy efficiency). Figure 3.2 illustrates the benefits of error resilience for pipelines of different lengths and for different error rates. The panels also illustrate the sensitivity to the voltage versus error rate relationship. From top to bottom, the panels correspond to a steeper voltage versus error rate relationship.

Figure 3.2 confirms the conclusion from the previous studies that there can indeed be significant error efficiency benefits from introducing error resilience into a design. In this case, we observe up to 30% energy efficiency benefit relative to a processor that is not allowed to produce errors ($e = 0$).

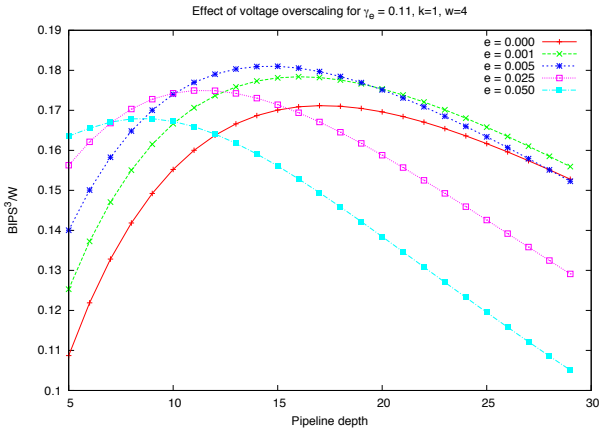
We also observe that the benefits of error resilience are strongly dependent on the relationship between voltage and error rate. When the voltage versus error rate relationship is steep, the benefits diminish as the error recovery time starts outweighing the power benefits of voltage overscaling. Note that the voltage versus error rate relationship is largely dictated by the timing slack distribution of the design, which in turn is affected by microarchitectural choices as well as the design methodology.

Figure 3.2 also demonstrates that the of error resilience are strongly tied to the number of pipestages. The figure shows that the optimal length of the pipeline (i.e., the one that maximizes energy efficiency) when errors are allowed is shorter than the optimal length of the pipeline when no errors are allowed. This relates to two aspects of error resilience: the time spent recovering from errors, and the relationship between path slack and the number of pipestages. For error recovery mechanisms in which recovery time is proportional to the length of the pipeline, shorter pipelines see shorter recovery time than longer pipelines for the same error rate. Similarly, for architectures whose available path slack is strongly dependent on the length of the pipeline, as modeled by Equation (3.10), shorter pipelines allow greater voltage overscaling before hitting the slack wall.

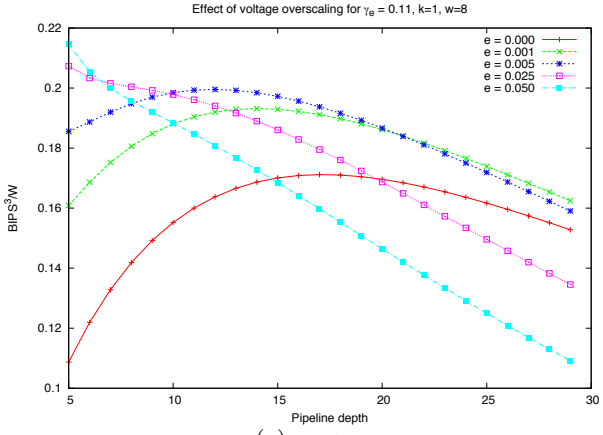
Figure 3.3 illustrates two examples where the optimal architecture has



(a) w=1



(b) w=4



(c) w=8

Figure 3.2: Error resiliency benefits can be substantial, and are closely tied to both the length of the pipeline and the relationship between error rate and voltage scaling. The figures show error resiliency benefit for different error rate versus voltage scaling relationships.

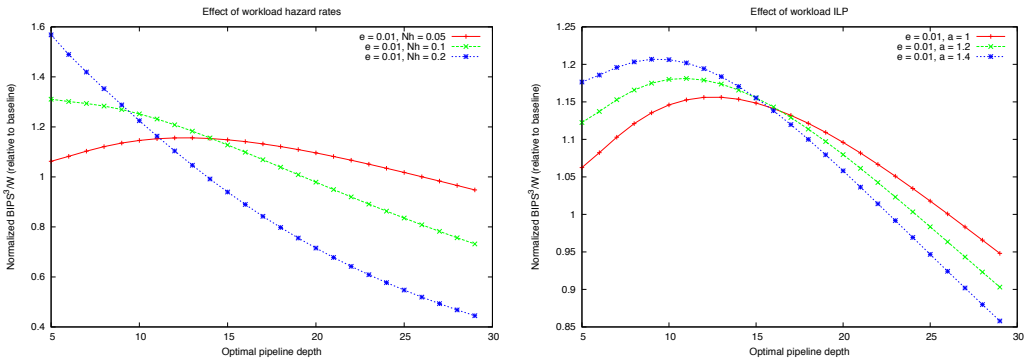
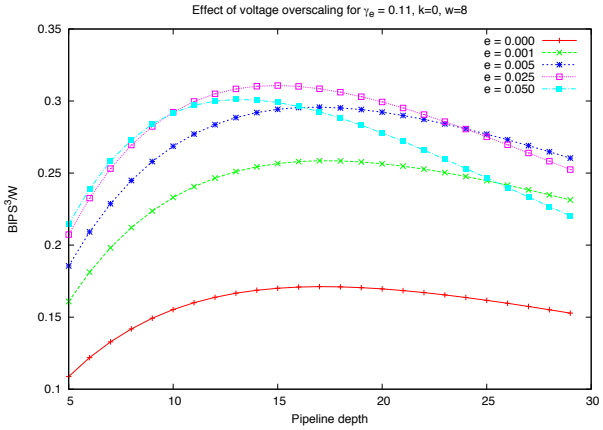


Figure 3.3: Error resiliency sees greater benefit for architectures with shorter optimal pipelines. Two instances where this can occur are when optimizing for workloads with greater control dependency (left panel), or larger amounts of ILP (right panel). Results are normalized to optimal non-error resilient design.

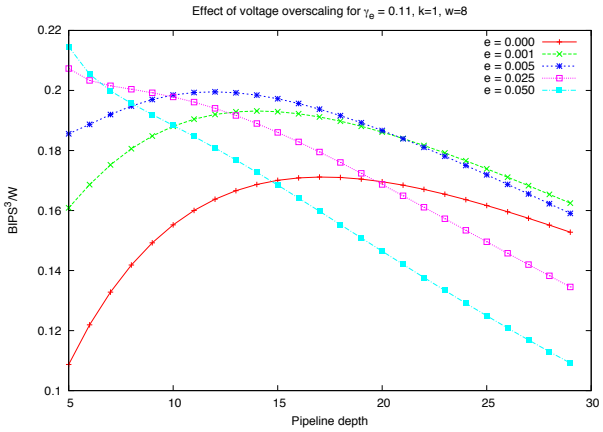
shorter pipeline depths and the effect this has on the benefit of error resiliency. The left panel shows the error resiliency benefit when designing for typical workloads consisting of various hazard rates. The highest hazard rate, having a shorter optimal pipeline depth due to the hazard recovery time, sees the greatest error resiliency benefit. Similarly, the right panel shows the error resiliency benefit increasing as the architecture is designed for a workload with higher average ILP. These illustrate the previous observation that error resiliency sees the greatest benefit in architectures with shorter pipeline depths, such as those targeting irregular workloads.

To further confirm the dependence of error resiliency benefits on the slack distribution and the number of pipestages, we studied the impact on energy efficiency benefits of pushing the slack wall closer to the nominal voltage at different rates when the number of pipestages is increased. Figure 3.4 shows the results. The top panel, $k = 0$, represents an architecture in which path slack is independent of the number of pipestages. As the length of the pipeline is increased, the performance improves proportionally with the frequency change, increasing the energy efficiency until the point where the hazard and error recovery time, in addition to latch growth, outweighs the performance improvement. For architectures in which path slacks are tightly coupled with the length of the pipeline ($k = 1$ or $k = 2$), the slack wall is hit sooner as the length of the pipeline increases, decreasing the energy efficiency benefits.

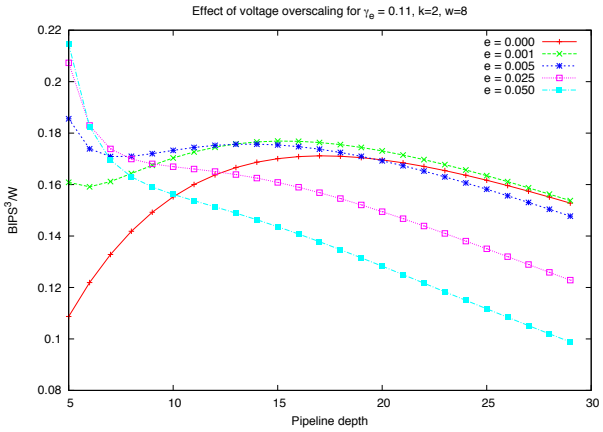
Finally, we observed the benefits from error resiliency for other energy



(a) k=0



(b) k=1



(c) k=2

Figure 3.4: Error resilient designs see greater benefits from shorter pipelines as the available path slack decreases faster due to pipelining.

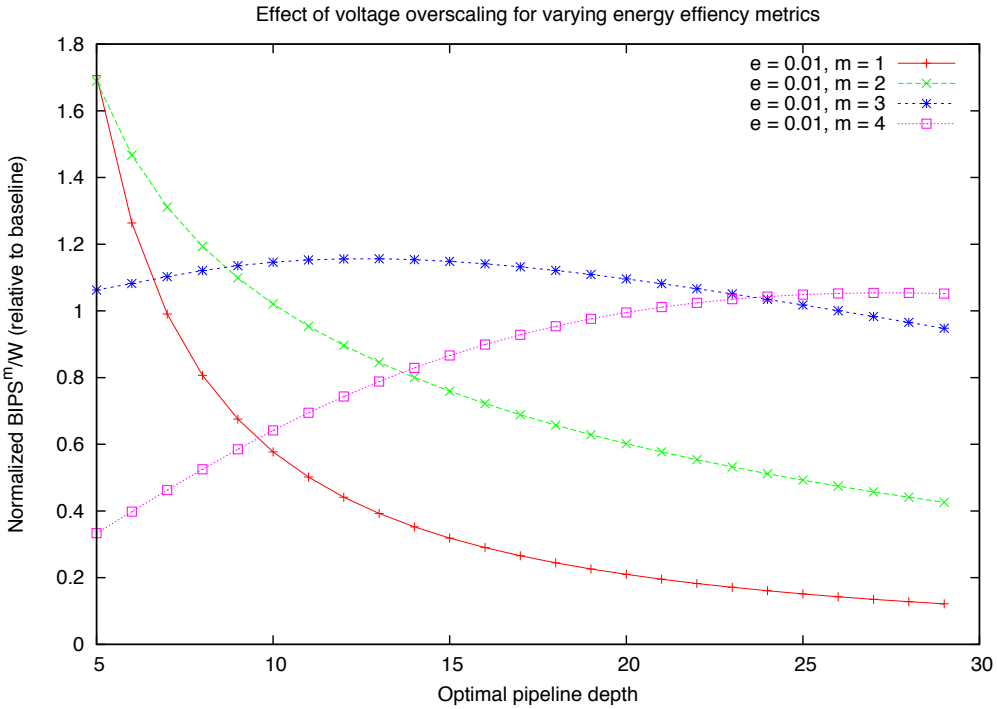


Figure 3.5: Benefits of error resiliency improve for energy efficiency metrics dominated by power. The figure shows the benefit of error resiliency for m=1,2,3,4 (BIPS^m/W) over the optimal non-error resilient baseline design.

efficiency metrics. As expected, the greatest error resiliency benefits are seen for the energy efficiency metrics dominated by power (lower values of m). The $m = 1$ curve sees the greatest error resiliency benefit and has the shortest optimal pipeline (pipelining only improves the performance portion of the metric, not the power). For performance-dominated energy efficiency metrics, the optimal pipelines are long; therefore, the power benefits from voltage overscaling are outweighed by the error recovery overheads. Long pipelines also have reduced path slack, further reducing the benefits of error resilience. Figure 3.5 demonstrates the benefits of error resiliency for the BIPS^m/W metric as m is varied.

3.3.2 Exploring the Benefits of Co-optimization

The previous results show the sensitivity of energy efficiency of error resilient designs to various architectural, circuit, and modeling parameters. We now consider the following question: how important is it to reconsider the archi-

tecture when introducing an error resilience mechanism into a design?

Figures 3.6 and 3.7 illustrate the benefits of error resiliency for an architecture that was optimized without error resiliency in mind against an architecture designed with error resiliency in mind. These figures show the energy efficiency gains that can be had from co-optimizing the architecture with error resiliency. Note that co-optimization, in this case, simply corresponds to identifying the optimal pipeline depth and the corresponding operating voltage for a *given* error resilience mechanism.

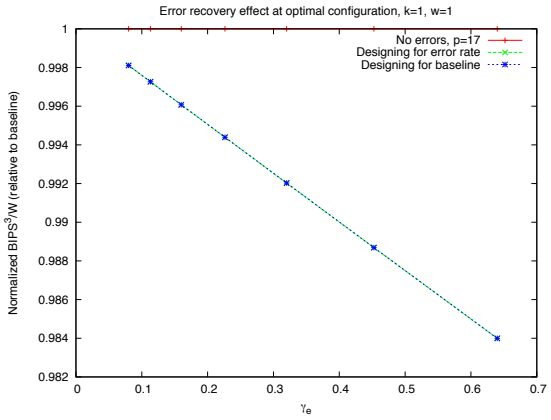
For small error recovery penalties, where the largest gains from error resiliency are achieved, we observe significant benefits from re-architecting the processor with error resiliency in mind. In fact, we observe gains greater than 15% in Figure 3.6. The gains from co-optimization diminish as the optimal error recovery penalty increases and the optimal error rate decreases, which has the effect of moving the optimal pipeline lengths closer to that of the baseline (i.e., the optimal pipeline when no errors are allowed).

We also observe that the benefits of co-optimization are strongly dependent on the relationship between error rate and voltage. From top to bottom, Figure 3.6 shows decreasing steepness of the voltage vs error rate curve. The greater the amount of possible voltage overscaling before hitting a certain error rate, the higher the optimal error rate, and therefore the greater the benefits from co-optimization.

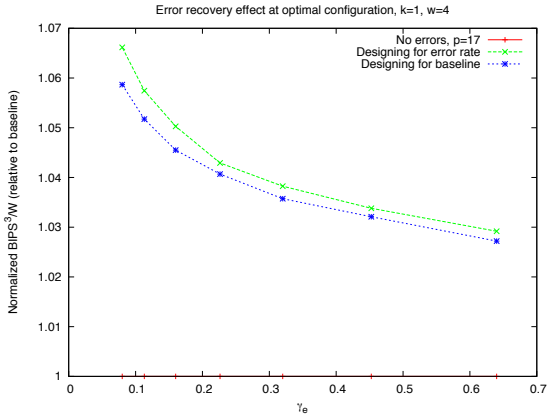
Lastly, we show that the benefits of co-optimization are also closely linked to the sensitivity of path slack to pipeline length. Figure 3.7 illustrates the advantages of co-optimization as the path slack moves from being independent of pipeline length to decreasing rapidly as the length of the pipeline increases ($k = 0$ to $k = 2$). The increased benefit can be attributed to the path slack's sensitivity to the pipeline length causing the optimal architectures to have shallower pipelines. In general, the greater the reduction in the optimal pipeline length when error resiliency is considered, the greater the benefit from co-optimization.

3.3.3 Validation

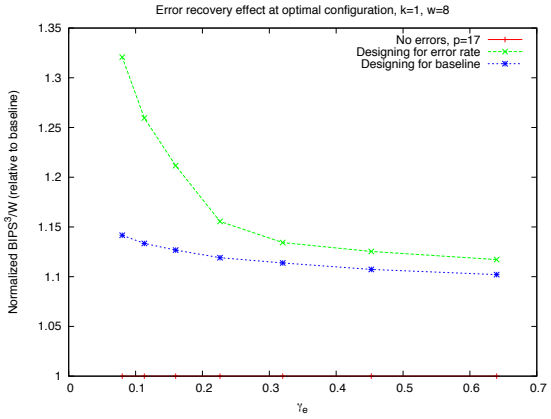
We used the cycle accurate simulation-based methodology described in Section 3.2 to validate our analytical model from Section 3.1. Our validation



(a) w=1

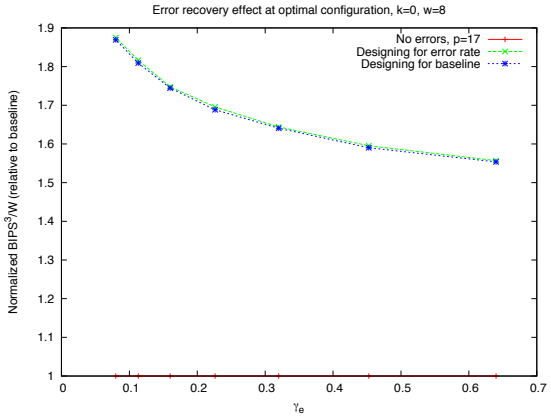


(b) w=4

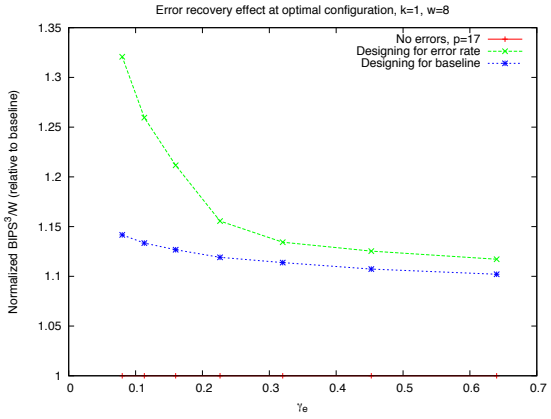


(c) w=8

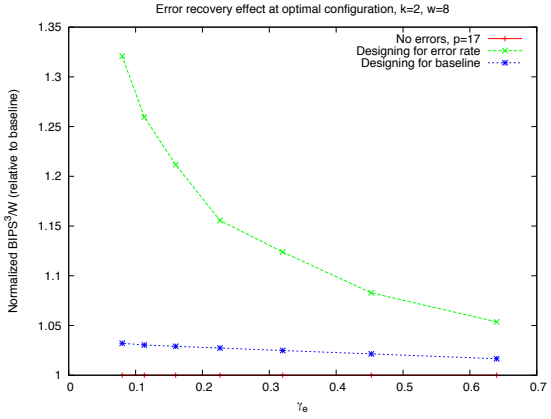
Figure 3.6: The benefit of co-optimizing an architecture depends strongly on the error rate versus voltage relationship. Architectures consisting of circuits seeing fewer errors at a particular voltage (bottom panel) will see the greatest benefit from co-optimization.



(a) k=0



(b) k=1



(c) k=2

Figure 3.7: Architectures with path slacks strongly sensitive to pipeline depths (bottom panel, k=2) see the greatest benefit from pipeline co-optimization with error resiliency. As path slack sensitivity increases, the optimal pipeline depth's deviation from the baseline's optimal pipeline depth increases, resulting in a a greater need for co-optimization.

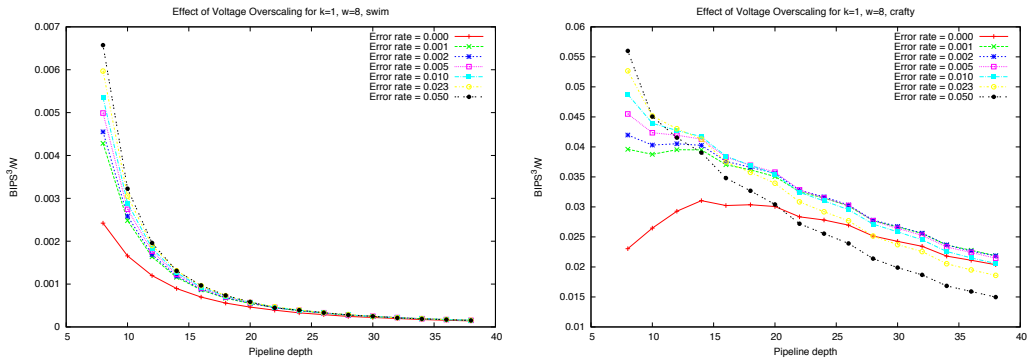


Figure 3.8: Simulated results demonstrating the benefits of error resiliency for two benchmarks, SWIM (left) and CRAFTY (right).

experiments were performed using the 8 SPEC2000 benchmarks in Table 3.3, mixing benchmarks from both the integer and floating point suites. Here, we focus on two benchmarks that illustrate the accuracy of our results and show how optimizing for two different workloads affects error resiliency benefits. These results assume the following parameters: $\gamma_e = 0.11$, $k = 1$, and $w = 8$.

Figure 3.8 shows the error resiliency benefits for the SWIM and CRAFTY benchmarks.

The results confirm that significant energy efficiency benefits are indeed possible from error resiliency. SWIM sees up to 171% improvement in energy efficiency, while CRAFTY sees up to 80% gain. Also, we observe that error resiliency benefits have a strong dependence on the pipeline length for CRAFTY. The error resiliency benefits are maximized when the pipeline has 8 stages, the minimum number of stages supported by the simulator. This is significantly different from the optimal pipeline length of 15 when no errors are allowed.

The SWIM benchmark is significantly more memory sensitive, and therefore has a shorter optimal pipeline than CRAFTY. In fact, the optimal pipeline depth is the minimum of 8 even when no errors are allowed. We observe that the benefits from error resilience are indeed higher for SWIM than CRAFTY (171% versus 80%), despite the fact that all other architectural parameters are the same. This confirms our previous conclusion that error resiliency benefits increase when the optimal architecture is a shorter pipeline.

Lastly, the CRAFTY results illustrate the need for co-optimization. As can be seen, the energy efficiency gains from error resilience are only 34%

over the baseline when operating at the optimal non-error resilient pipeline depth. If the architect were to co-optimize the architecture with the error resilience mechanism, therefore reconsidering the pipeline depth, the energy efficiency could be as high as 80% over the baseline.

Figure 3.9 summarizes the results for all 8 SPEC benchmarks investigated and compares benefits to the pipeline depth. On average, we see a 136% energy efficiency gain from error resiliency, 25% of which is due to co-optimizing the pipeline depth and error resiliency mechanism. In addition, we confirm that those systems designed for the shortest pipeline depths (those points highest on the negative pipeline depth scale), see the largest benefits from voltage overscaling-based error resiliency.

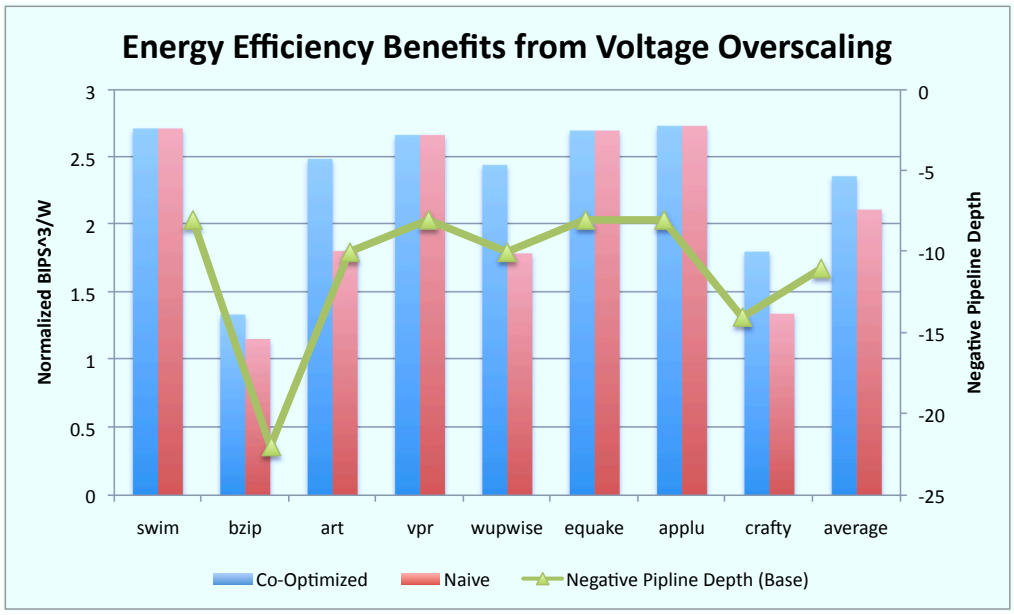


Figure 3.9: Simulated results illustrating energy-delay^2 when operating at multiple error rates and pipeline depths. Results are normalized to the energy-efficiency of the optimal non-error resilient design.

CHAPTER 4

OPTIMAL PIPELINING FOR FREQUENCY OVERSCALING

Frequency overscaling allows greater throughput at the expense of increased dynamic power and reduced reliability. Reliability degrades as the critical or near-critical paths start having timing violations on overscaling. The goal of co-optimizing architecture with an error resilience mechanism for frequency overscaling-induced timing errors is to carefully balance the throughput benefits of frequency overscaling with the increased cost of error recovery.

4.1 Theory

Our analytical model for an error resilient processor needs only slight modification to describe frequency overscaling-based timing speculation.

Frequency overscaling is modeled as

$$f_s = f_o / F_{scale} \tag{4.1}$$

where f_s is the scaled frequency, f_o is the base frequency defined in the original Equation (3.4), and F_{scale} is the clock scaling factor (defined on the range $0 < F_{scale} \leq 1$). For example, when F_{scale} is 0.75, we operate the clock at 75% of the nominal clock period, which is equivalent to a 33% increase in operating frequency.

Because we are scaling down the clock period, there is a linear decrease in path slack within a circuit, and therefore an increase in timing errors. To determine the error rate for a given scaling factor, we use the same error model as for voltage overscaling:

$$e = min(1, ((F_{scale} - 1)/(F_{wall} - 1))^w)$$

where F_{scale} is the frequency overscaling factor, F_{wall} is fraction of the base

clock period at which the slack wall is reached, and w is the exponential growth factor. F_{wall} is defined in the range $0 < F_{wall} \leq 1$. In other words, as F_{scale} approaches F_{wall}, the error rate approaches 100%.

As previously illustrated in Figure 3.1, our error model assumes that the amount of slack in a path is also dependent on pipeline depth. As the average slack of a circuit decreases, so does the amount of scaling possible before the slack wall is reached. As a result, F_{wall} when performing frequency overscaling remains dependent on the pipeline depth, defined by Equation (3.10):

$$F_{wall} = 1 - (1 - F_{wallb}) * (p_b/p)^k$$

where F_{wallb} is the fraction of the clock period for a 5 stage pipeline at which the slack wall is reached, and k is a parameter relating how quickly slack decreases as pipeline depth increases. Therefore, as the pipeline becomes deeper, less frequency overscaling is possible before the slack wall is reached.

Note that the above model does not account for the increased impact of memory access latency on performance as frequency is increased.

The new power and performance equations are

$$T/N_I = F_{scale}/f_o a + \gamma_h N_h p F_{scale}/f_o + T_{err}/N_I \tag{4.2}$$

$$P_T = ((F_{cg} f_o P_d)/F_{scale} + P_l) N_L p^\eta \tag{4.3}$$

where f_o is the base pipelined frequency from Equation (3.4) and T_{err}/N_I is the time per instruction spent recovering from timing speculation errors, as defined in Equation (3.5). We use the same overall power/performance metric equation as defined in Equation (3.1).

4.2 Methodology

Our methodology for frequency overscaling follows the same methodology as for voltage overscaling. Like before, we evaluated the various design tradeoffs using our analytical model and used cycle accurate simulations to verify our results. We again used dynamic latch power data derived from WATTCH in our analytical model based on the following power formula:

$$P_T = (f(P_{sim}/f_{sim}) + (.3P_{sim}/.7))p^{1.3}$$

where P_{sim} is the dynamic power reported by the simulator at the base frequency f_{sim}. As before, we assume a leakage power that is 30% of the total power, and $\eta = 1.3$.

For validating our analytical results we again used cycle accurate simulations of SPEC2000 benchmarks [23] in a version of SMTSIM modified to support frequency overscaling coupled with an error recovery mechanism. The parameters were the same as described in Tables 3.1 and 3.2.

4.3 Results and Analysis

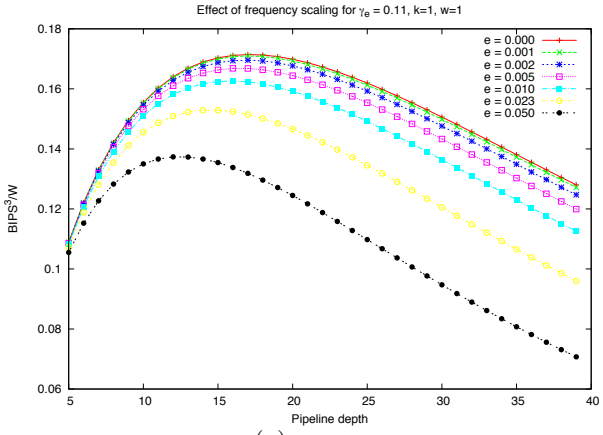
In this section, we analyze the relationship between the benefits from error resilience and pipeline, circuit, and workload characteristics when performing frequency overscaling. We also present results from our validation experiments.

4.3.1 Exploring the Interaction between Error Resilience, Pipelining, Circuit Structure, and the Metric for Energy Efficiency

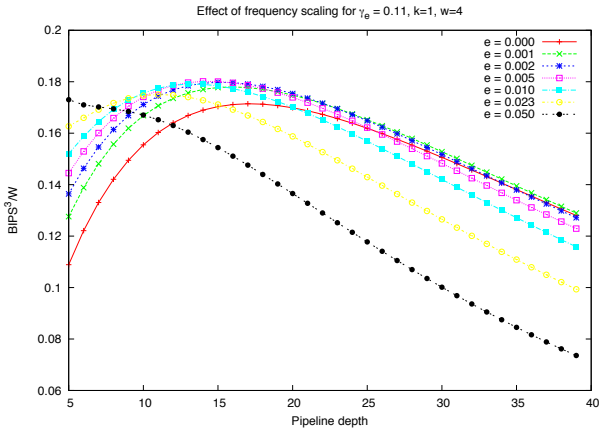
We begin by discussing the energy efficiency benefits of frequency-based timing speculation. Figure 4.1 shows the benefits of error resilient systems employing frequency overscaling at various error rates and pipeline depths. The multiple plots show the sensitivity of energy efficiency to the error growth rate, parameter w from Equation (3.9).

The panels confirm that the energy-efficiency benefits from frequency overscaling increase as fewer errors occur for a particular frequency overscaling (bottom panel). This is a trend similar to that observed in the voltage overscaling results from the previous chapter.

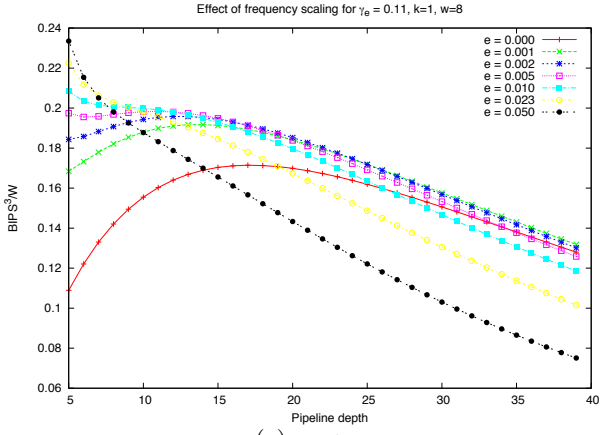
Due to the path slack dependence on pipeline depth, we continue to see that shorter pipelines see greater benefits from error resiliency. Although frequency overscaling is a mechanism for improving performance, we see that it is most effective when employed for systems with shorter pipelines. Figure 4.2



(a) w=1

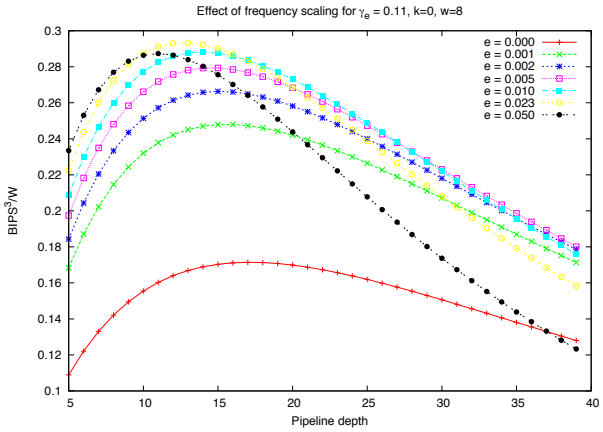


(b) w=4

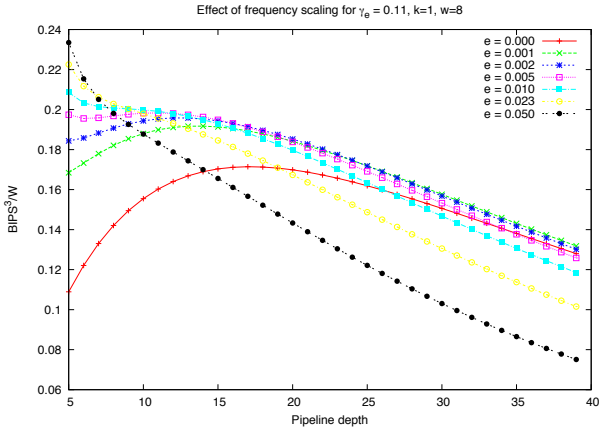


(c) w=8

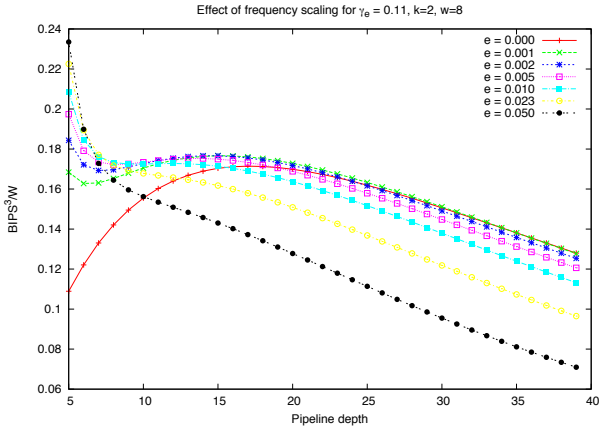
Figure 4.1: Error resiliency benefits can be substantial, and are closely tied to both the length of the pipeline and the relationship between error rate and frequency overscaling. The panels show error resiliency benefit for different error rate growth rates.



(a) k=0



(b) k=1



(c) k=2

Figure 4.2: Frequency overscaling-based timing speculation systems continue to see greater gains when employed on systems with shorter pipeline depths. The panels show increasing sensitivity of slack to pipeline depth (larger values of k).

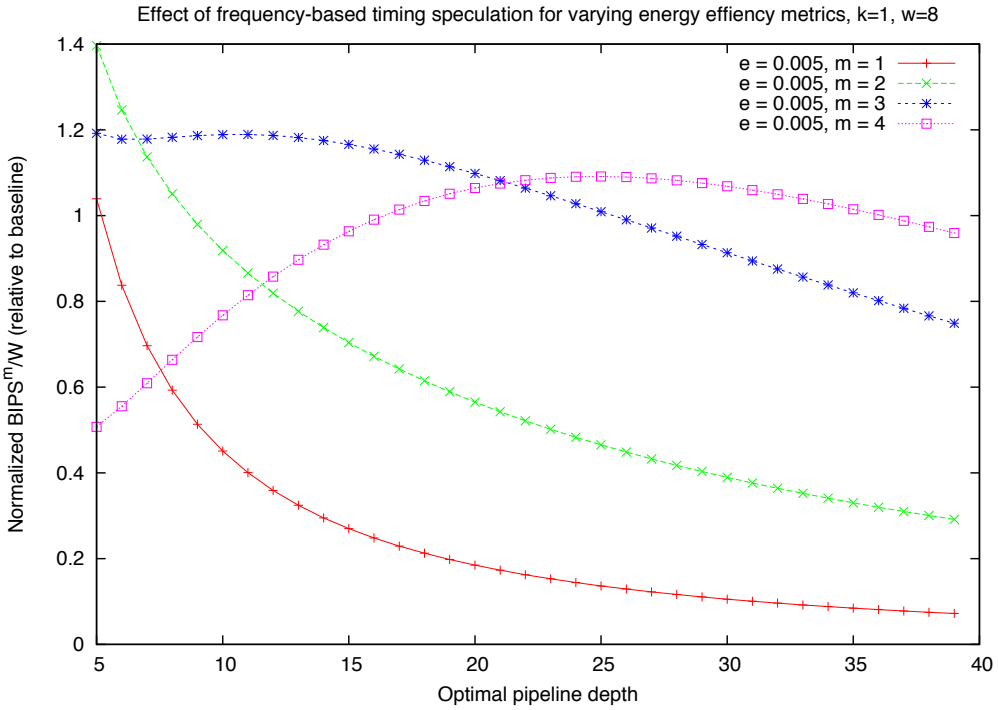


Figure 4.3: Benefits of frequency-based timing speculation improve for metrics that weight energy efficiency by both power and performance, but begin to diminish as they are dominated by performance. The figure shows the benefit of error resiliency for m=1,2,3,4 (BIPS^m/W) over the non-error resilient baseline.

illustrates the effect on energy efficiency of different path slack sensitivities to pipeline depth (higher k = more sensitive). The relationship between frequency overscaling efficiency and pipeline sensitivity approximately matches that for voltage overscaling.

Lastly, we considered the effect of designing an architecture for different energy efficiency metrics. Figure 4.3 illustrates the difference in energy efficiency as the metric is increasingly dominated by performance (m=4), as opposed to power ($m = 1$). Unlike voltage overscaling, where the cost of error recovery is offset by savings in power, frequency overscaling results in enhanced performance. As a result, metrics dominated by power see little benefit from frequency overscaling. On the other hand, if the metric is dominated by performance, the benefits of deeper pipelines begin to outweigh the benefits of error resiliency. As a result, systems not employing error resiliency and operating with deep pipelines see the greatest performance for large values of m,.

To summarize, error resiliency mechanisms do not significantly enhance

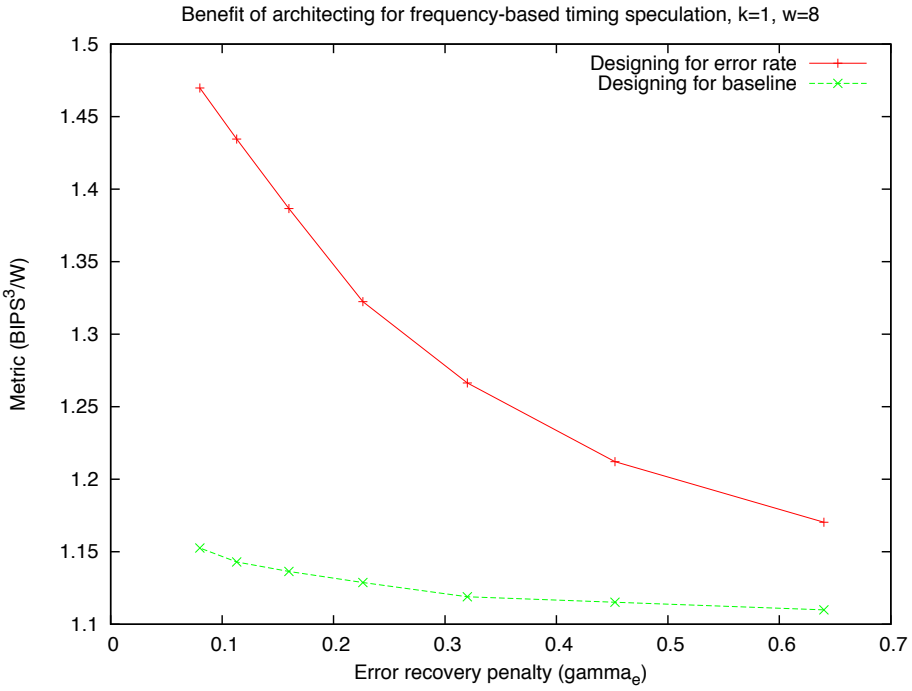


Figure 4.4: Gains from energy resiliency are sensitive to architectural decisions. In this case, up to 30% of the gain over baseline can be lost when the system is designed without considering error resiliency.

efficiency in scenarios where the optimization metric is dominated by performance. In such scenarios, designs benefit more from deeper pipelines than from timing speculation. Timing speculation has the greatest energy efficiency benefits when the optimization metric is dominated by power (voltage overscaling may be employed in such scenarios), or when performance and power are both important, in which case both voltage overscaling and frequency overscaling may be viable techniques. In the next chapter, we attempt to further compare the two techniques.

4.3.2 Exploring the Benefits of Co-optimization

Due to the slack dependence on pipeline depth, we continue to see a significant effect of architecture on energy efficiency gains from error resiliency. Figure 4.4 illustrates these effects. As before, the amount by which the error resiliency gains are sensitive to pipeline depth is tied to the slack sensitivity to pipeline depth.

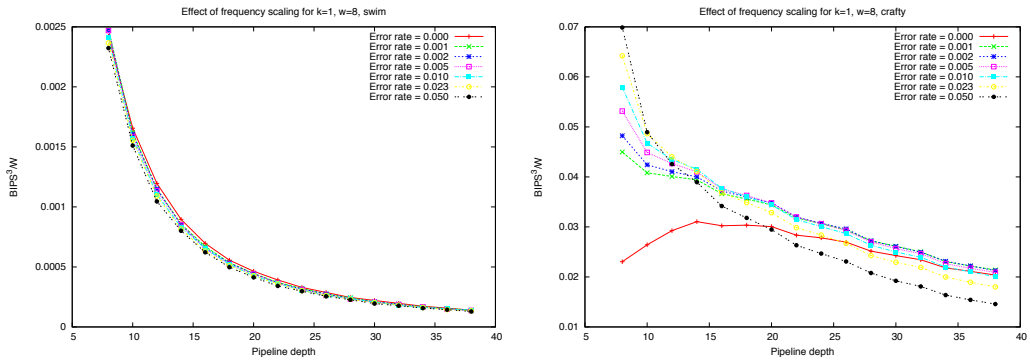


Figure 4.5: Simulated results illustrating energy-delay^2 when operating at multiple error rates and pipeline depths for two benchmarks, SWIM (left) and CRAFTY (right). CRAFTY sees considerably larger gains due to it's computational nature, versus SWIM's memory sensitive nature.

4.3.3 Validation

We validated our analytical results using the cycle-accurate simulations described in Section 4.2. To illustrate the sensitivity to pipeline depth and error rate, Figure 4.5 focuses on the SWIM and CRAFTY benchmarks. For these results we assume that $\gamma_e = 0.11$, $k = 1$, and $w = 8$.

The results demonstrate that CRAFTY sees significant benefits from frequency overscaling, up to 125% over the baseline. Most of these benefits are achieved when employing a shallow pipeline. When operating at the same pipeline depth as the baseline, only 33% improvement is seen, again illustrating the importance of co-optimization.

SWIM, on the other hand, sees no benefits from frequency overscaling, even for the shortest pipeline. This can be explained by the different memory sensitivity of the two benchmarks. CRAFTY is a largely computational benchmark, with a small memory footprint [25] and a high cache hit rate for both L1 and L2. SWIM on the other hand, has a large footprint, and had misses frequently in the L1 and L2 caches. Because significantly more time was spent accessing memory, which operates on a separate clock than our core and hence does not scale, SWIM benefits significantly less from frequency increases. This is seen both in that the optimal pipeline depth of the baseline (e=0 case) is as short as possible, and in that introducing frequency overscaling only introduces error overheads without any performance benefit. In general, benefits from frequency overscaling greatly depend on the benchmark, often considerably more than voltage overscaling.

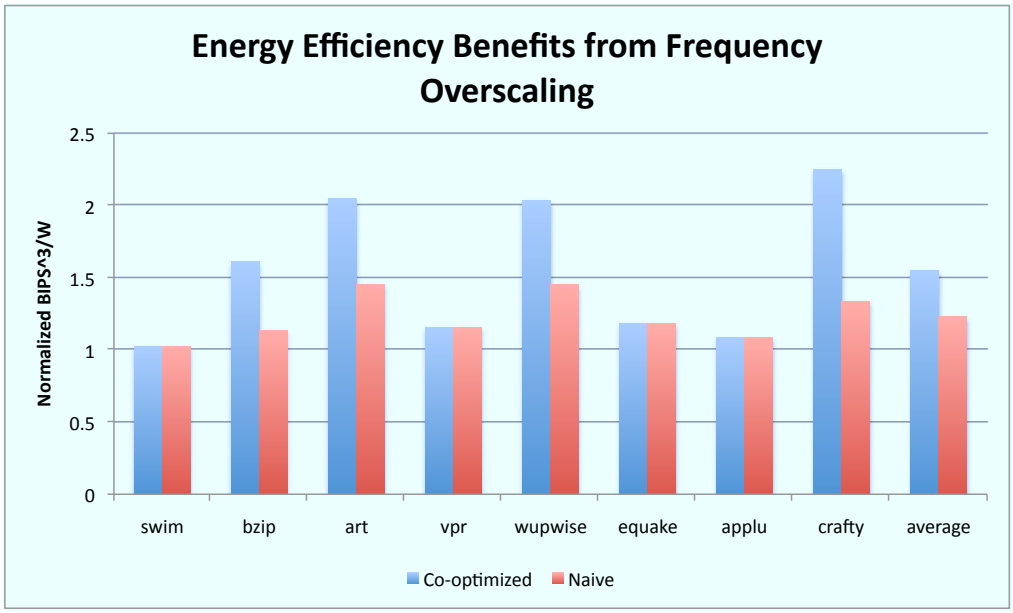


Figure 4.6: Simulated results illustrating energy-delay^2 benefits from frequency-based timing speculation normalized to the optimal non-error resilient design.

Figure 4.6 compares the energy efficiency gains from introducing error resiliency without reconsidering pipeline depth against the benefits from co-optimizing error resiliency and pipeline depth. We see an average energy efficiency benefit of 55%, compared with a 23% benefit when co-optimization is not considered.

CHAPTER 5

COMPARING PIPELINING BENEFITS: FREQUENCY VS. VOLTAGE OVERSCALING

In the previous two chapters we have demonstrated that both frequency overscaling and voltage overscaling can lead to considerable energy efficiency benefits when coupled with an error recovery mechanism. When architecting a processor for error resiliency, architects must decide which type of overscaling technique to employ, and include appropriate circuitry. Voltage overscaling requires DVFS support, for example. In this chapter we further investigate how the two speculation techniques relate to one another in order to better understand when a particular overscaling technique is more viable than the other.

The benefits of frequency and voltage overscaling are closely tied to their scaling versus error rate relationships. In order to compare the two, we must consider how those relationships differ. To compare the scaling versus error rate relationships for the two overscaling mechanisms, we need to understand how path slack, and therefore error rate, depends on each type of overscaling.

In this chapter, we investigate how the two speculation methods affect path slack differently, and discuss how this affects timing speculation and error resiliency. We then present a methodology for comparing voltage and frequency overscaling. Finally, we discuss the scenarios under which one mechanism of overscaling is more appropriate than the other.

5.1 Impact of Overscaling on Path Slack

We first discuss the basic equations for path delay which determine the path slack. Then we employ circuit timing data to create a model for the change in slack wall with overscaling.

5.1.1 Theory

Path delay of a circuit is given by the following equation:

$$T_d = \frac{C_L V_{dd}}{k'_n (W/L)(V_{dd} - V_t)^2} \tag{5.1}$$

where C_L is the load capacitance of the wire, W and L are the physical properties of the wire, and k'_n is a constant. The path slack can, therefore, be expressed as

$$T_s = T_c - T_d \tag{5.2}$$

where T_c is the clock period (inverse of the operating frequency) and T_d is the delay time expressed in Equation (5.1). Errors are caused when the path slack becomes negative.

Both the timing speculation techniques greatly affect the path slack. Frequency overscaling decreases the clock period, T_c, while voltage overscaling increases the path delay, T_d. However, while the path slack decreases linearly with frequency overscaling, it decreases super-linearly with voltage overscaling due to an exponential dependence of path delay on voltage. The insight that path slack decreases slower with frequency overscaling than voltage overscaling is the key to comparing voltage and frequency overscaling-based timing speculation techniques.

5.1.2 Analyzing Circuit Timing Data

In order to better understand the dependence of path slack on the magnitude of frequency and voltage overscaling, we analyzed timing data from eight modules of the *OpenSparc T1* processor [26] (see Table 5.1). Module designs were implemented with a TSMC 65GP library (65 nm) and the initial netlists were synthesized with Synopsys Design Compiler vY-2006.06-SP5.

By varying the voltage from nominal (1.5 V), to 50% of nominal (0.75 V), we were able to determine how each module's worst negative slack varied with voltage, and therefore the path delay versus voltage. This was used to determine the voltage at which the slack becomes negative, causing errors.

In addition, the path delay at nominal voltage was used to determine the slack when frequency overscaling was employed, based on Equation (5.2).

Table 5.1: *T1 OpenSparc Modules*

Module	Stage	Desc	Cell #
lsu_dctl	MEM	L1 DCache Control	4537
lsu_qctl1	MEM	LDST Queue Control	2485
lsu_stb_ctl	MEM	ST Buffer Control	854
sparc_exu_div	EX	Integer Division	4809
sparc_exu_ecl	EX	Execution Unit Control	2302
sparc_ifu_errdp	FD	Error Datapath	4184
sparc_ifu_fcl	FD	L1 ICache and PC Control	2431
spu_ctl	SPU	Stream Processing Control	3341

This was then used to determine the operating frequency before the slack becomes negative and errors occur.

Figure 5.1 illustrates the change in slack as the voltage or clock period is scaled down. It is apparent that slack decreases exponentially with voltage overscaling, while the decrease is linear with frequency overscaling.

Because of the different rate at which slack changes under frequency and voltage overscaling, the amount of overscaling that can be done before errors occur is different for the two overscaling mechanisms. From our circuit timing data, presented in Table 5.2, frequency can be overscaled by twice as much as voltage before errors occur. In general, frequency can be overscaled by a greater magnitude than voltage before hitting a certain error rate. However, the exact difference depends on the initial path slack as well as the intrinsic and the load capacitance of the circuit.

5.2 Methodology

Based on our understanding of the relationship between path slack and the magnitude of frequency/voltage overscaling, we use the following relationships between frequency overscaling and voltage overscaling (in terms of the magnitude of overscaling before a certain error rate – in this case 100% – is reached) to compare their energy efficiency benefits.

1. 1:1 Scaling - The same amount of overscaling results in reaching the slack wall for both frequency and voltage.

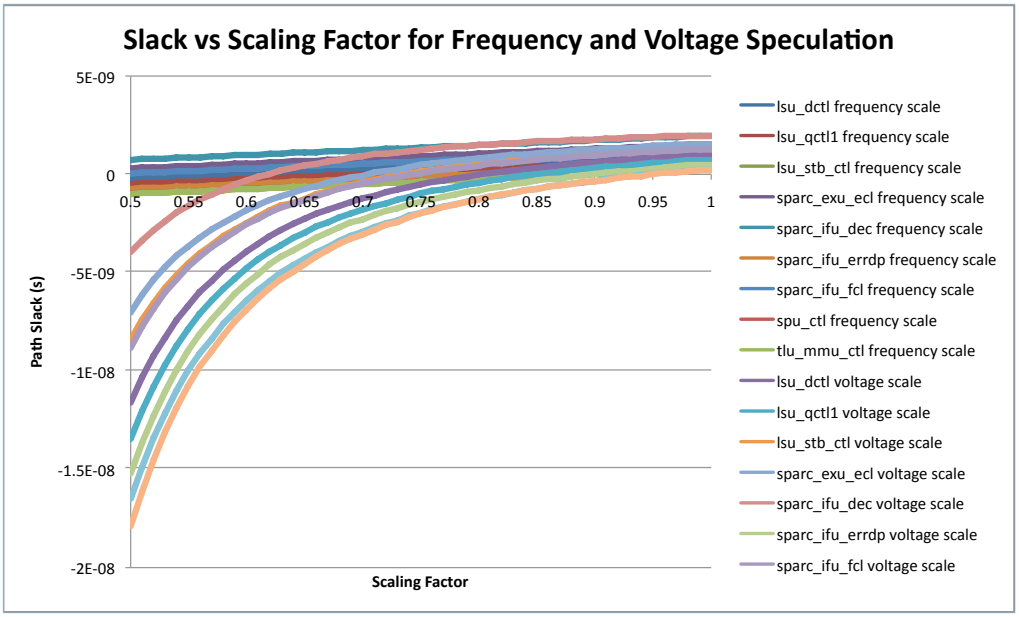


Figure 5.1: Change in slack with either voltage or frequency overscaling. Frequency overscaling causes a significantly slower decrease in slack (linear vs. exponential).

Table 5.2: *Amount of Voltage vs. Frequency Overscaling before Errors are Observed*

Module	Initial Slack (ns)	V %	F %	F/V
lsu_dctl_c	0.97	20	39	1.95
lsu_qctl1_c	0.72	15	29	1.93
lsu_stb_ctl_c	1.32	27	53	1.96
sparc_exu_ecl_c	1.52	30	61	2.03
sparc_ifu_dec_c	1.95	38	79	2.08
sparc_ifu_errdp_c	0.47	10	19	1.90
sparc_ifu_fcl_c	1.26	26	51	1.96
spu_ctl_c	0.23	5	10	2.00
tlu_mmu_ctl_c	0.22	5	9	1.80
Average				1.96
StdDev				0.08

2. Quadratic Scaling - Frequency can be overscaled quadratically higher than voltage before the slack wall is reached.

3. Double Scaling - Frequency can be overscaled twice as much as voltage before the slack wall is reached.

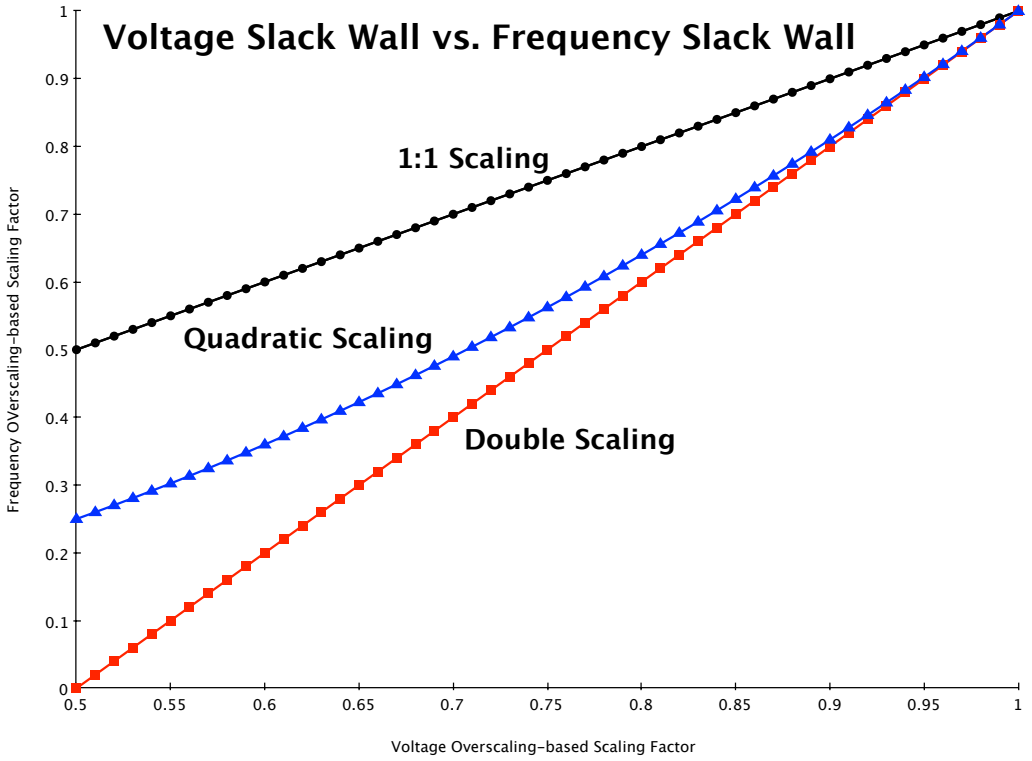


Figure 5.2: The three models for the amount of frequency overscaling at which the slack wall is reached (y axis), compared with voltage scaling (x axis). An x value of 75 and a y value 50 means that when the slack wall for voltage is reached after scaling down to 75% of the nominal, frequency overscaling in the same system would be able to scale down to 50% of the nominal before the slack wall is reached.

The relationships are illustrated in Figure 5.2. The effect on the error rate is then shown in Figure 5.3.

We can now use these relationships to adjust the slack wall, F_{wall} in Equation (3.9), appropriately.

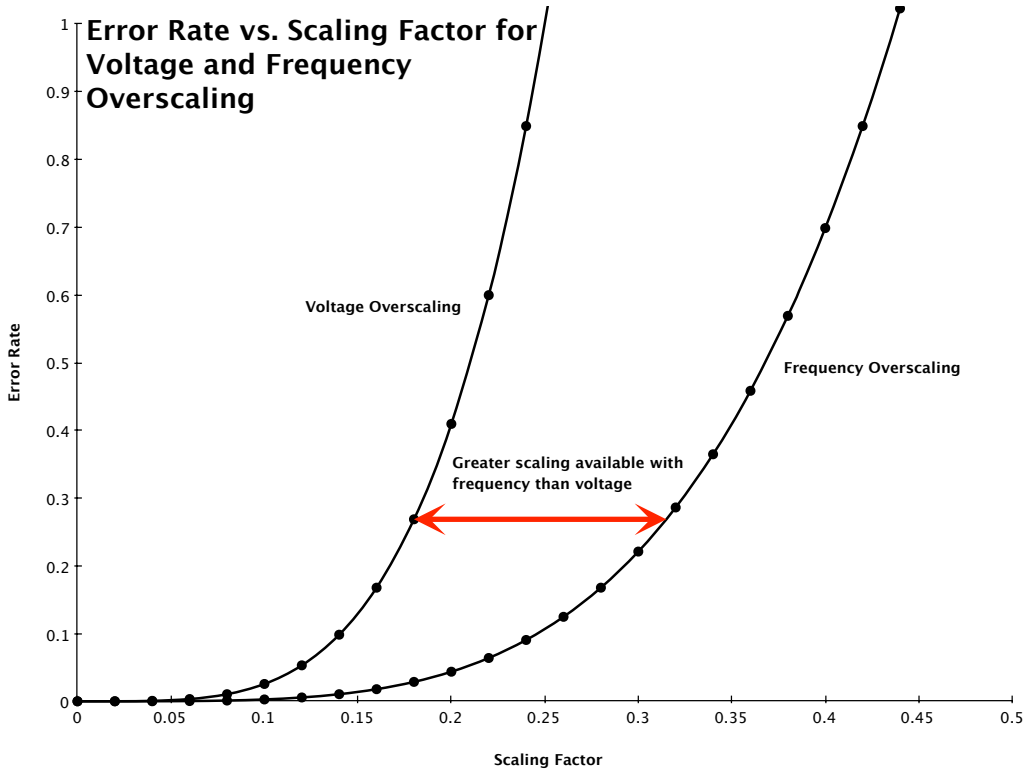


Figure 5.3: Error rate vs. scaling for voltage and frequency overscaling. Frequency-based speculation can achieve significantly more scaling at the same error rate due to a further slack wall.

5.3 Results and Analysis

We now discuss the relative merits of the two timing speculation methods when co-optimizing architecture and error resiliency.

5.3.1 Energy Efficiency

Figure 5.4 illustrates how the different relationships for frequency and voltage overscaling before the slack wall is reached affect the energy efficiency savings. The results show that as more frequency overscaling can be performed before reaching the slack wall, the energy efficiency gains compared with voltage overscaling increase.

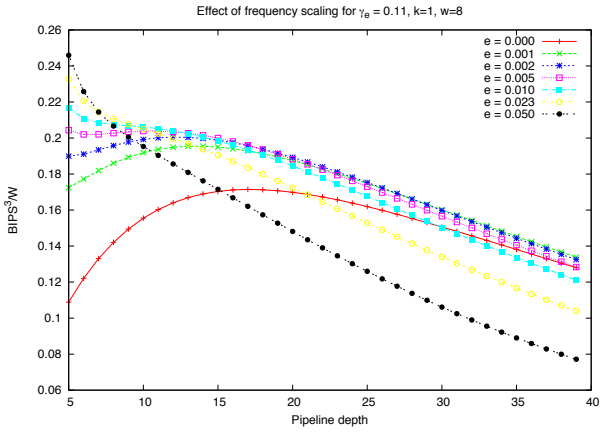
We also investigated the maximum energy efficiency gains due to timing speculation when operating at the optimal configuration for various energy-delay metrics. Figure 5.5 shows that although voltage overscaling will see the largest gains for an energy metric (m=1), it is slightly less effective for an energy-delay metric (m=2), given a quadratic scaling. Furthermore, it provides substantially smaller energy efficiency gains for an energy-delay^2 (m=3) metric. Even in the case where we assume that there is no difference in amount of scaling before reaching the slack wall (Figure 5.5(c)), frequency overscaling still sees energy efficiency gains for an energy-delay^2 metric.

Note that these results assume a constant hazard penalty. When memory sensitivity is considered, the hazard penalty will actually become a function of the frequency. As the extent to which this is the case is benchmark dependent, we leave the analysis of memory sensitivity to the validation section, where we consider cycle-accurate simulations of SPEC2000 benchmarks.

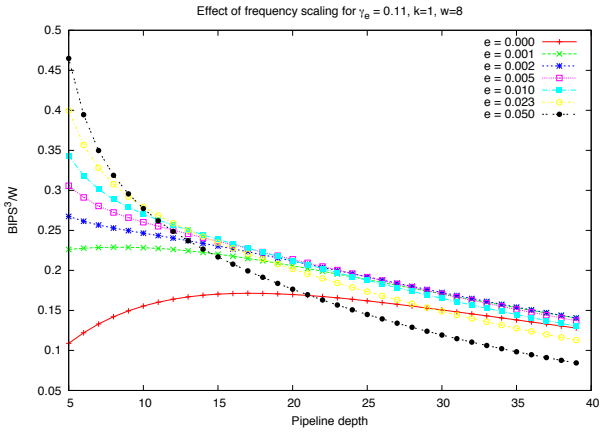
As discussed in Chapter 4, once the metric is dominated by performance the metric gains due to deeper pipelines outweigh those from timing speculation, resulting in error recovery time being too expensive (due to the proportional error recovery penalty).

5.3.2 Benefits of Co-optimization

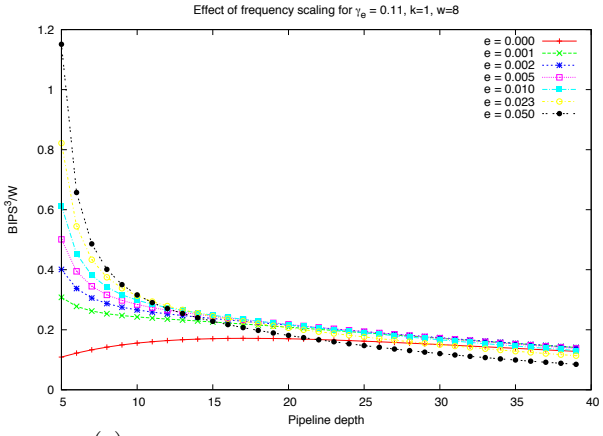
We showed in Figure 5.4 that frequency overscaling results in designs that are more sensitive to pipeline depth, but are able to achieve substantially larger



(a) 1:1 Frequency vs. Voltage scaling

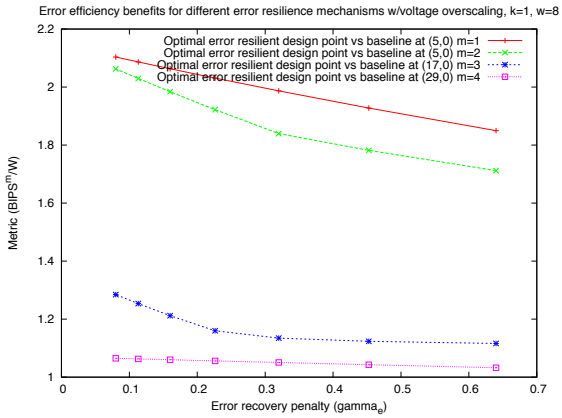


(b) Quadratic Frequency vs. Voltage scaling

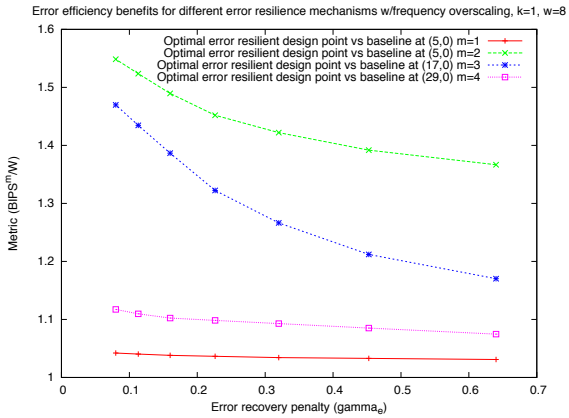


(c) Double Frequency vs. Voltage scaling

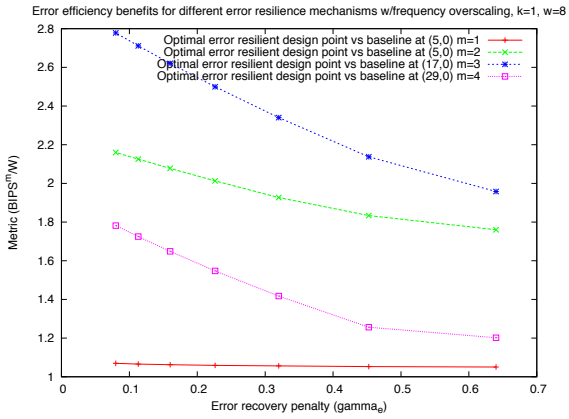
Figure 5.4: Error resilient systems employing frequency overscaling are more sensitive to pipeline depth and can see greater energy efficiency gains due to smoother error rate curves. As the amount of scaling before reaching slack wall increases, so do energy efficiency benefits and sensitivity to pipeline depth.



(a) Voltage Overscaling



(b) Frequency Overscaling (1:1 slack wall)



(c) Frequency Overscaling (Quadratic slack wall relationship)

Figure 5.5: Comparison of energy efficiency gains from timing speculation for various energy efficiency metrics. Voltage overscaling only sees larger energy efficiency gains for an energy metric (m=1), while frequency overscaling sees a slight improvement for energy-delay (m=2), and substantial improvement for energy-delay^2 (m=3).

energy efficiency gains, as seen in Figure 5.5. As a result, the benefits of co-optimization are substantially larger when considering frequency overscaling.

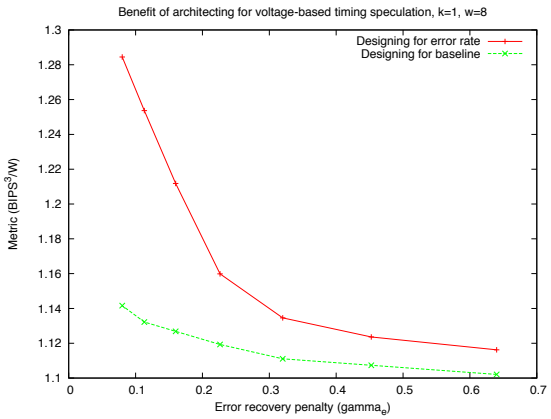
Figure 5.6 shows the difference in energy efficiency gains when operating at the optimal pipeline depth when error resiliency is considered, compared with a design that optimizes the pipeline depth for the no-error case. We see that frequency-based error resilient designs still see greater benefits from co-optimization than voltage scaling, even for the 1:1 scaling case. The benefits increase as more frequency overscaling can be done for a given amount of voltage overscaling.

5.3.3 Validation

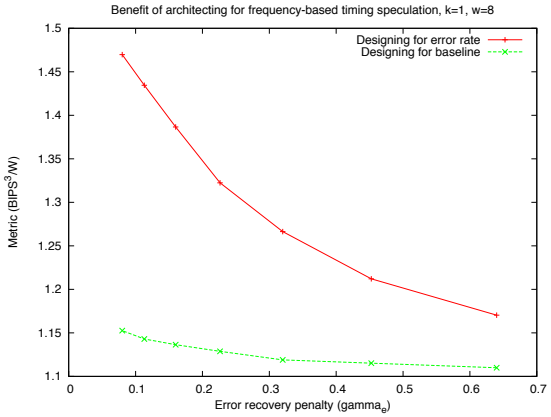
Figure 5.7 summarizes the energy efficiency gains from frequency-based timing speculation when optimizing systems for the 8 SPEC2000 benchmarks in Table 3.3. We assume that quadratically higher overscaling can be done for frequency than voltage before reaching the slack wall. These results assume the following parameters: $\gamma_e = 0.11$, $k = 1$, and $w = 8$. We can see that, on average, frequency-based timing speculation sees 170% energy efficiency improvement, 34% higher than the average from voltage-based speculation. The majority of these improvements are from a selection of four benchmarks (BZIP, ART, WUPWISE, and CRAFTY), which are particularly compute-intensive, and hence can fully exploit the increase in frequency. On the other hand, the remaining four benchmarks (SWIM, VPR, EQUAKE, APPLU) see greater energy efficiency gains from voltage overscaling. This again reinforces the fact that the benefits from frequency-based timing speculation are limited by the the workload, despite the fact that it has a more tolerant error rate curve. This is unlike voltage speculation where the benefits are maximized for any design benefitting from short pipelines.

It is interesting to note that BZIP and CRAFTY, although seeing the smallest gains under voltage overscaling, see the largest gains from frequency overscaling. This can be attributed to their memory insensitive nature, and reinforces the conclusion that frequency-based speculation and voltage-based speculation can be complementary in nature.

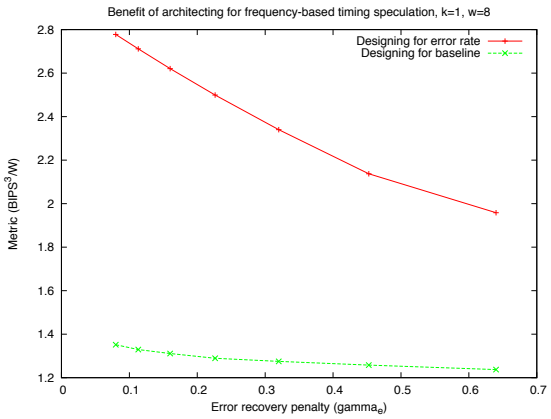
Lastly, we confirm that frequency-based timing speculation is more sensitive to pipeline depth. This means that when frequency-based timing spec-



(a) Voltage Overscaling



(b) Frequency Overscaling (1:1 slack wall)



(c) Frequency Overscaling (Quadratic slack wall relationship)

Figure 5.6: Energy efficiency gains from timing speculation for designs that co-optimize pipeline depth with error resiliency vs. designs that optimize pipeline depth independently. Pipeline depth co-optimization is significantly more important for frequency overscaling than voltage scaling, even for 1:1 Scaling.

ulation does see energy efficiency gains, the benefits from co-optimization are larger. On average, frequency-based timing speculation sees 125% more improvement from co-optimization over the baseline, compared with voltage-based timing speculation that sees 29%.

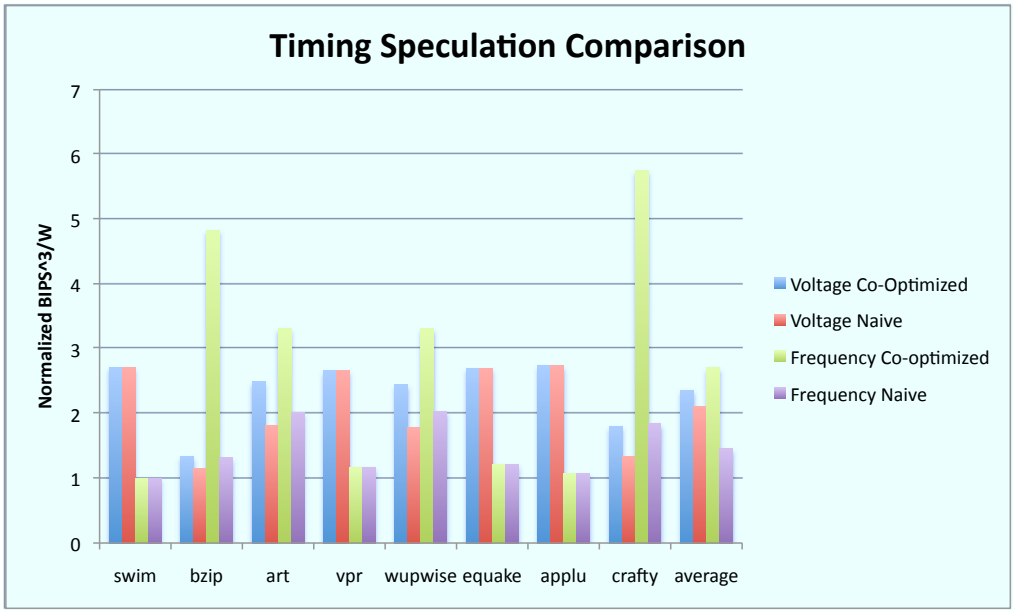


Figure 5.7: Simulated results illustrating energy-delay^2 benefits from both voltage-based and frequency-based timing speculation. Results are normalized to the energy efficiency of the optimal design that does not employ timing speculation. Co-optimized refers to optimizing the pipeline depth while considering error resiliency. Naive refers to employing the pipeline depth optimized without considering error resiliency.

CHAPTER 6

CONCLUSIONS

6.1 Limitations and Future Work

In this section we discuss the limitations of the model and ways in which to further the research in this area.

First of all, Hartstein and Puzak made several simplifying assumptions in the generation of their model. Because our work builds on their model, we inherently incorporate many of the same assumptions. One of these is the assumption that as you pipeline a processor, the frequency scales linearly. Depending on the ability to divide logic, this may not be the case. This may affect the accuracy of the model.

Another of the aspects not explored is that, depending on the error recovery mechanism, errors can be compounding in nature. This can arise if the error recovery itself may be prone to error. We make the assumption that our error recovery mechanisms are not themselves prone to errors.

In addition, we assume a continuous relationship between voltage/frequency overscaling and error rate. In reality, the distribution of path lengths may be clustered, resulting in plateaus in the voltage versus error rate curve, as well as other irregularities. This would require further analysis and timing data. Our models provide conclusions for the estimated average case.

An issue which might arise in practice, but we do not explore, is the possibility of performing useful work during error recovery time. For example, if a cache miss is initiated before an error occurs, then during the error recovery time the memory system can actually perform useful work, masking some of the cost of error recovery. Particularly in single-issue in-order systems, this could largely mitigate the performance effects of errors. A similar situation could arise if error recovery simply introduces bubbles in the pipeline. If an error occurs in an early pipeline stage, the subsequent stages can continue to

perform useful work and commit instructions. These secondary effects could have further impact on the benefits of employing timing speculation.

Lastly, our research shows that frequency-based timing speculation has the potential for large energy efficiency gains, but is dependent on workload. This leads to the possibility of intelligent hardware mechanisms that can monitor the memory sensitivity of the current workload in order to intelligently decide between voltage or frequency-based timing speculation. This is a possible line of future research.

6.2 Conclusions

Previous work on timing speculation has either explored the benefits of customizing the design methodology for a particular error resilience mechanism [2–5] or has attempted to understand the benefits from error resilience for a particular resiliency mechanism [1, 6–8]. There is no work, to the best of our knowledge, that attempts to understand the benefits of co-optimizing microarchitecture and error resilience.

In this thesis, we presented the first study on co-optimizing a processor pipeline with an error resilience mechanism. We developed an analytical model that relates the benefits from error resiliency to the depth of the pipeline as well as its circuit structure. The model was used to determine the optimal pipeline depth for different energy efficiency metrics for different error resilience overheads. Our model was capable of considering both frequency-based and voltage-based timing speculation.

Our results demonstrated that several interesting relationships exist between error resilience and pipeline structure. For example, we showed that there are significant energy efficiency benefits to pipelining an architecture for an error resiliency mechanism versus error resiliency-agnostic pipelining. As another example, we showed that benefits from error resiliency are greater for short pipelines than long pipelines, although in the case of frequency-based timing speculation the workload can limit the benefit. We also confirmed that the benefits from error resiliency are higher when the circuit structure is such that error rate increases slowly on reducing input voltage or increasing frequency versus a circuit optimized for power where a slack wall exists at the nominal operating point [4, 20].

Lastly, we presented a model for performing comparison between frequency-based and voltage-based timing speculation. We showed that frequency-based speculation can feasibly achieve larger benefits (given an appropriate energy efficiency metric) than voltage speculation if the workloads are not overly memory latency bound. This presents potential future lines of research in employing two complementary methods of timing speculation. We further showed the frequency-based speculation is more sensitive to pipeline depth, increasing the importance of co-optimization.

Overall, this thesis concludes that the architecture of the processor must be reconsidered to fully exploit the potential of error resiliency.

REFERENCES

[1] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: A low-power pipeline based on circuit-level timing speculation," in *MICRO 36: Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*, 2003, p. 7.

[2] B. Greskamp, L. Wan, W. R. Karpuzcu, J. J. Cook, J. Torrellas, D. Chen, and C. Zilles, "Blueshift: Designing processors for timing speculation from the ground up," in *International Symposium on High Performance Computer Architecture*, 2009, pp. 213–224.

[3] A. B. Kahng, S. Kang, R. Kumar, and J. Sartori, "Recovery-driven design: A methodology for power minimization for error tolerant processor modules," in *DAC '10: Proceedings of the 47th Annual Design Automation Conference*, 2010, pp. 825–830.

[4] A. B. Kahng, S. Kang, R. Kumar, and J. Sartori, "Slack redistribution for graceful degradation under voltage overscaling," in *Proceedings of the Asia and South Pacific Design Automation Conference*, 2010, pp. 825–831.

[5] A. B. Kahng, S. Kang, R. Kumar, and J. Sartori, "Designing a processor from the ground up to allow voltage/reliability tradeoffs," in *International Symposium on High Performance Computer Architecture*, 2010, pp. 1–11.

[6] R. Hegde and N. R. Shanbhag, "Energy-efficient signal processing via algorithmic noise-tolerance," in *ISLPED '99: Proceedings of the 1999 International Symposium on Low Power Electronics and Design*, 1999, pp. 30–35.

[7] J. C. Smolens, B. T. Gold, B. Falsafi, and J. C. Hoe, "Reunion: Complexity-effective multicore redundancy," in *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, 2006, pp. 223–234.

[8] B. Greskamp and J. Torrellas, "Paceline: Improving single-thread performance in nanoscale cmps through core overclocking," in *PACT '07: Proceedings of the 16th International Conference on Parallel Architecture and Compilation Techniques*, 2007, pp. 213–224.

[9] T. Kehl, "Hardware self-tuning and circuit performance monitoring," in *IEEE International Conference on Computer Design*, 1993, pp. 188–192.

[10] A. Hartstein and T. R. Puzak, "Optimum power/performance pipeline depth," in *MICRO 36: Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*, 2003, p. 117.

[11] M. de Kruijf, S. Nomura, and K. Sankaralingam, "A unified model for timing speculation: Evaluating the impact of technology scaling, cmos design style, and fault recovery mechanism," to appear in *DSN '10: Proceedings of the 2010 International Conference on Dependable Systems and Networks*, 2010.

[12] C. Weaver and T. M. Austin, "A fault tolerant approach to microprocessor design," in *DSN '01: Proceedings of the 2001 International Conference on Dependable Systems and Networks (formerly: FTCS)*, 2001, pp. 411–420.

[13] J. Sloan, D. Kesler, R. Kumar, and A. Rahimi, "A numerical optimization-based methodology for application robustification: Transforming applications for error tolerance," to appear in *DSN '10: Proceedings of the 2010 International Conference on Dependable Systems and Networks*, 2010.

[14] M. S. Hrishikesh, D. Burger, N. P. Jouppi, S. W. Keckler, K. I. Farkas, and P. Shivakumar, "The optimal logic depth per pipeline stage is 6 to 8 FO4 inverter delays," in *ISCA '02: Proceedings of the 29th Annual International Symposium on Computer Architecture*, 2002, pp. 14–24.

[15] V. Srinivasan, D. Brooks, M. Gschwind, P. Bose, V. Zyuban, P. N. Strenski, and P. G. Emma, "Optimizing pipelines for power and performance," in *Proceedings of the 35th International Symposium on Microarchitecture*, 2002, pp. 333–344.

[16] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low power cmos digital design," *IEEE Journal of Solid State Circuits*, vol. 27, pp. 473–484, 1995.

[17] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for reduced cpu energy," in *OSDI '94: Proceedings of the 1st USENIX conference on Operating Systems Design and Implementation*, 1994, p. 2.

[18] K. Keutzer and P. Vanbekbergen, "The impact of cad on the design of low power digital circuits," in *IEEE Symposium on Low Power Electronics*. IEEE, 1994, pp. 42–45.

[19] N. Shanbhag, R. Abdallah, R. Kumar, and D. Jones, "Stochastic computation," in *DAC '10: Proceedings of the 47th Annual Design Automation Conference*, 2010, pp. 859–864.

[20] J. Patel, "Cmos process variations: A critical operation point hypothesis," Online Presentation, 2008. [Online]. Available: http://www.stanford.edu/class/ee380/Abstracts/080402-jhpatel.pdf

[21] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," in *ISCA '00: Proceedings of the 27th annual international symposium on Computer architecture*, 2000, pp. 83–94.

[22] D. Tullsen, "The SMTSIM multithreading simulator," 2010. [Online]. Available: http://cseweb.ucsd.edu/users/tullsen/smtsim.html

[23] "SPEC CPU2000," 2000. [Online]. Available: http://www.spec.org/cpu2000/

[24] E. Perelman, G. Hamerly, M. Van Biesbrouck, T. Sherwood, and B. Calder, "Using simpoint for accurate and efficient simulation," in *SIGMETRICS '03: Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. New York, NY, USA: ACM, 2003, pp. 318–319.

[25] S. Sair and M. Charney, "Memory behavior of the spec2000 benchmark suite," IBM, Tech. Rep. 21852, 2000.

[26] "Sun OpenSPARC Project," 2010. [Online]. Available: http://www.sun.com/processors/opensparc/