Reducing Pipeline Energy Demands
with Local DVS and Dynamic Retiming

Seokwoo Lee, Shidhartha Das, Toan Pham, Todd Austin, David Blaauw, and Trevor Mudge

Advanced Computer Architecture Lab

The University of Michigan,1301 Beal Ave, Ann Arbor, MI 48109

razor@eecs.umich.edu

ABSTRACT

The quadratic relationship between voltage and energy has made dynamic voltage scaling (DVS) one of the most powerful techniques to reduce system power demands. Recently, techniques such as Razor DVS, voltage overscaling, and Intelligent Energy Management have emerged as approaches to further reduce voltage by eliminating costly voltage margins inserted into traditional designs to ensure always-correct operation. The degree to which a global voltage controller can shave voltage margins is limited by imbalances in pipeline stage latency. Since all pipeline stages share the same voltage, the stage exercising the longest critical path will define the overall voltage of the system, even if other stages could potentially run at lower voltages. In this paper, we evaluate two local tuning mechanisms in the context of Razor DVS, a local voltage controller scheme that allows each pipeline stages it's own voltage level, and a lower cost dynamic retiming scheme that incorporates per-stage clock delay elements to allow longer-latency pipeline stages to "borrow" time from shorter-latency stages.

Using simulation, we draw two key insights from our study. First, mitigating pipeline stage imbalances renders additional DVS energy savings. A Razor pipeline design with dynamic retiming finds an additional 12% energy savings over global voltage control (resulting in an overall energy savings of more than 28% compared to fully-margined DVS). Second, we demonstrate that imbalances arise not only from design factors, but also from run-time characteristics. As the program (or program phase) changes, we see different logic paths in multiple stages exercised frequently, necessitating a dynamic fine-tuning of local control. This result suggests that even well-balanced pipelines could benefit from dynamic retiming.

Categories and Subject Descriptors:

C.0 [**Computer System Organization**]: System Architecture

General Terms:

Design

Keywords:

Razor, Local DVS, Dynamic retiming with global DVS

1. INTRODUCTION

In recent years, portable electronic devices have endeavored to deliver higher levels of performance within increasingly constrained power budgets. While many techniques exist to lower

energy demands, many do so at the cost of reduced processing throughput. The gap between high performance and low power can be bridged through the use of dynamic voltage scaling (DVS), where periods of low processor utilization can be exploited by lowering the clock frequency to the minimum required level [7]. Frequency reductions enable similar reductions in supply voltage, which in turn renders a quadratic decrease in circuit energy demands [8,9].

With traditional DVS, the voltage allowed at any frequency is determined at design time using static timing analysis under a combination of worst-case fabrication and environmental factors, including process variation, temperature fluctuations, supply voltage noise, among others. To accommodate these uncertainties, designers add voltage margins to the critical voltage to guarantee the correct operation in even with the worst case scenario. However, previous studies have reported this conservative approach overly constrains voltage, because the worst case scenarios are rare [1].

Of particular focus in this work is the recently proposed Razor DVS technique, a voltage scaling technology that utilizes in-situ error detection and correct mechanisms to gauge voltage margins at run-time [1]. The key idea behind Razor is to tune supply voltage by monitoring circuit timing error rates at run-time. A global voltage controller seeks out the optimal operating voltage where the energy benefits of reduced voltage operation are balanced with the energy cost of recovery due to circuit timing violations. The approach eliminates all forms of voltage margin, including those that accommodate design, fabrication and run-time factors. Previously, a Razor prototype processor demonstrated (through detailed simulation) that significant energy savings are possible [1]. In a prototype design, the ALU was shown to use 42% less energy with Razor DVS, while only incurring at most a 2.5% performance impact due to circuit timing error recovery.

As proposed, Razor DVS utilizes a global voltage controller that adjusts the supply voltage by monitoring the error rate of the entire pipeline. Voltage is reduced until the most frequently executed critical logic path is exposed, at which point the energy cost of recovering from timing errors begin to outweigh the energy reductions of operating the entire pipeline at a lower voltage. Because the technique uses a single global voltage, this constraint is enforced even if other pipeline stages in the design are not operating at the lowest voltage possible.

In this paper, we investigate two local voltage tuning techniques to mitigate the effects of imbalances that may arise in the latency of pipeline stages. The first technique, *local DVS*, simply provides each pipeline stage with its own voltage controller, thus allowing each stage to choose its own optimized voltage. While ideal for energy reduction, local DVS adds significant complexity in the form of level converters and added voltage regulation complexity. To achieve the benefits of local DVS at lower cost, we propose a novel *dynamic retiming* scheme. Dynamic retiming incorporates per-stage clock delay elements that allow longer-latency stages to "borrow" time from shorter-latency stages. Using simulation, we draw two key insights from our analyses. First, we find that eliminating

voltage margins due to imbalances in pipeline stage latency renders significant DVS energy savings. A Razor pipeline design with local DVS improves energy saving by 38% on average, while the reduced-cost dynamic retiming scheme finds a 28% energy savings. Second, we see that imbalances arise not only from design factors, but also from run-time characteristics. As the program (or program phase) changes, the logic paths exercised most frequently also changes, necessitating a fine-tuning of local control. This result suggests that even well-balanced pipeline designs could benefit from dynamic retiming.

The remainder of the paper is organized as follows. In Section 2, we detail Razor DVS and its global voltage control scheme. In Section 3, we present the local DVS and dynamic retiming techniques. In Section 4, we give detailed simulation results that demonstrate the relative benefits of the approaches. Finally, Section 5 presents related work, and we draw conclusions in Section 6.

2. RAZOR SYSTEM OVERVIEW

2.1 In-situ Error Detection and Recovery

Razor supports a combination of circuit and architectural techniques to implement low cost in-situ error detection and correction of circuit delay failures. By monitoring global errors rates, Razor's global voltage controller is able to eliminate voltage margins by seeking a voltage that minimizes pipeline energy demands without incurring excessive circuit timing failures. At the circuit level, the Razor pipeline flip-flops are augmented with a shadow latch that takes a second sample of stage values approximately 1/2 clock period into the following clock cycle. Razor pipeline operating voltage is constrained such that the worst-case stage delay is guaranteed to meet the shadow latch setup time, even though the main flip-flop could fail.

By comparing the values latched by main flip-flop and shadow latch, it is possible to detect timing errors in the main latch. In this event, the known-correct value in the shadow latch is forwarded to main latch. Since incorrect values may have been forward to other pipeline stages at the moment the timing error occurred, a microarchitectural recovery mechanism must be invoked to flush potentially incorrect values out of the processor pipeline. This process can be treated much in the same way as a branch misprediction, in which all instructions behind the errant instruction are invalidated, and the pipeline is restarted after the errant instruction. A particularly important aspect of the Razor design is that recovery guarantees forward progress for the erring instruction, hence, it is possible to ensure that a program will continue to make forward progress (albeit slowly) even if all Razor latches are failing in all cycles. For additional information on Razor, including details on metastability detection, Razor latch design, and microarchitectural recovery techniques, the reader is referred to a recent article on the subject [1].

2.2 Global Voltage Control

The role of the Razor global voltage controller is to continually adjust voltage to the point where all voltage margins have been eliminated. Figure 1 illustrates the Razor global voltage control system. The controller samples at regular intervals the pipeline error rate, E_{sample}, to determine the extent of margins that exist. By maintaining a small but non-zero error rate E_{ref}, the controller can ensure that system is operating with minimal voltage margins and minimal performance impacts (due to timing error recoveries). The controller computes the error differential, E_{diff}, which is the input to a simple proportional voltage controller. The controller operates continuously because environmental conditions such as temperature or supply noise may change sufficiently to warrant fine-tuning of global voltage.
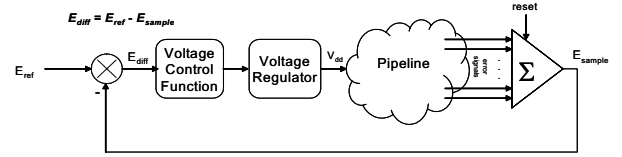

Figure 1: Razor global voltage control

2.3 Razor Prototype design

The Razor prototype implementation details, as well as a die picture, are shown in Figure 2. The entire processor was specified in Verilog and synthesized using Synopsys Design Analyzer (version 2003.03-2). The design was taped out in December 2003, and silicon is expected to be available by the end of February 2004. The prototype design was implemented in the TSMC 0.18um process, and it is validated to operate at 180 Mhz. The design implements a 64-bit Alpha pipeline with 4 pipeline stages and 8k-bytes of I-cache and D-cache. Of the 2408 flip-flops in the design, only 192 were on logic paths sufficiently critical to require Razor flip-flops. (If a logic path is guaranteed to never fail at the worst-case conditions and voltage, it does not require a Razor latch.) The delayed Razor clock is delayed by 1/2 the clock cycle from the system clock, generated locally within the Razor flip-flop by inverting the main clock.

Power analysis was performed on the prototype Razor design, using both gate level power simulations and SPICE to evaluate the overhead of the error correction and detection circuits. The total power consumption during error free operation is expected to be 425 mW at 1.8 V at a clock frequency of 180 MHz. The energy consumption of the standard and Razor flip-flops during error free operation is listed in Figure 2. Two values are shown for each flip-flop, reflecting the cases when the latched data is changing (*switching*) and is not changing (*static*). The total power overhead due to the presence of the Razor error detection and correction circuitry in error-free operation is 3.1% of total power. The final three rows of the table show the Razor flip-flop power overheads due to error detection and recovery. The energy required to detect an error and restore the correct shadow latch data into the main flip-flop was 210 fJ per error event for each Razor flip-flop. The total energy to perform a single error detection and correction event in the Alpha pipeline was 189 pJ, resulting in an additional Razor flip-flop power overhead of approximately 1% of total power when operating at a 10% error rate. Note that additional power overheads (not shown in the table) are incurred when instructions are flushed out of the pipeline to recover program state. Simulation based analysis of pipeline recovery estimated the energy cost of pipeline recovery to be about 18 times greater than the execution of a single instruction (details of this analysis can be found in [1]).

3. LOCAL CONTROL SYSTEM

The Razor global voltage control system performs best when each stage exhibits equal evaluation latency. In this case, the global control system will tune the voltage just to the point where each stage continues to operate correctly for most cycles. With good pipeline balance each stage will run at its own optimal voltage, *i.e.,* the voltage where all design- and run-time margins are eliminated. If imbalances occur in the latency of the pipeline stages, the effectiveness of global DVS decreases, as one (or a few) stages will set an operating voltage that leaves additional headroom to lower voltage in shorter-latency stages.

Experimental analysis of the prototype Razor design with global voltage control indicated that over 90% of timing failures

Technology node	0.18 μm
Voltage range	1.8 V to 1.2 V
Total number of logic gates	45,661
D-cache size	8 KBytes
I-cache size	8 KBytes
Die size	3 x 3.3 mm
Clock frequency	180 MHz
Clock delay	2.5 nS
Total number of flip-flops	2408
Number of Razor flip-flops	192
Error free operation	
Total power	425 mW
Standard FF energy (switching/static)	49 fJ / 95 fJ
Razor FF energy (switching/static)	60 fJ / 160 fJ
% total power overhead	3.1%
Error correction and recovery overhead	
Energy per Razor FF per error event	210 fJ
Total energy per error event	189 pJ
Razor FF recovery overhead at 10% error rate	1%
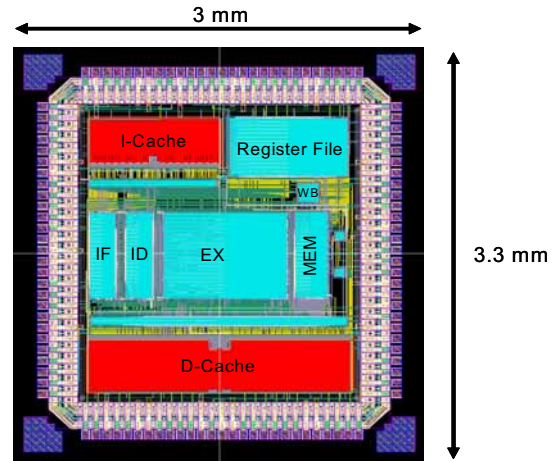
Figure 2: Razor prototype implementation details and layout

were confined to the decode (ID) and execute (EX) stages of the pipeline. This result is in part due to the fact that the ID and EX stages have high worst-case latency (as measured by the static timing analyzer). However, additional factors, such as the frequency of logic path evaluations, also factor into the effectiveness of global voltage control. For example in the prototype Razor design, the MEM stage has the longest latency, yet it rarely constrained global voltage because it generated very few errors (*i.e.,* its frequently executed circuit paths exhibited much shorter latency). In the following subsections, we develop two local control techniques, designed to allow individual stages to minimize their individual energy requirements.

3.1 Local DVS

The local DVS voltage controller optimization is quite simple in concept. Instead of constraining pipeline voltage to single global voltage, local DVS provides each pipeline stage with its own locally adjustable voltage. The local voltage controller monitors the local error rate of each pipeline stage, tuning voltage to maintain a small but non-zero local error rate. Consequently, each pipeline stage will individually minimize its energy demands. Each local voltage controller is implemented using a proportional control function identical to the Razor global voltage controller.

Local DVS control will certainly perform better than the more constrained global DVS control, but this added energy savings comes with an increased implementation cost. In a design with local DVS, each stage will require its own voltage regulator. Additionally, logic that interfaces between stages will require voltage level conversion circuitry. While some early evidence suggests that multiple level voltage regulation may be possible without excessive cost [10], we largely consider local DVS to be a benchmark design - ideal in capability but too costly at this point in time for a real implementation.

3.2 Dynamic Retiming

Dynamic retiming is an optimization that gives much of the benefit of local DVS, but without the need for costly local voltage regulation. The optimization is based on the observation that with global DVS pipeline stages that have low error rates are not fully utilizing the clock period. Consequently, an opportunity exists to use standard FSM retiming techniques [4,5] to allow stages with high error rates to "borrow" evaluation time from

Figure 3: Dynamic retiming example

stages with low error rates. This redistribution of evaluation time is accomplished by carefully skewing the clock boundaries between pipeline stages. After retiming is performed, stages with high error rates will have more than a clock cycle to evaluate, which will in turn reduce their timing error rates and afford additional reductions in global voltage levels.

Figure 3 illustrates the concept of dynamic pipeline retiming. In the example, the clock cycle time of each stage is initially set to the global clock period. The errors at the initial voltage level are concentrated in the EX stage of the pipeline (with a 0.3% error rate). The dynamic retiming control will recognize this imbalance in error rates and adjust pipeline clock boundaries to borrow time from stages with lower error rates, in this case the IF stage (with a 0.001% error rate). The retiming is implemented by skewing all clock edges between the "borrowing" stage (EX) and the "lending" stage (IF). The retimed pipeline is illustrated in the bottom pipeline of Figure 3. The period of time for EX stage processing is increased to 6.1ns by skewing clock edge at the ID/EX and IF/ID stage boundaries 1.1ns earlier. Consequently, the clock period of IF stage will decrease to 3.9ns. The throughput of the pipeline as a whole is unchanged, as the entire pipeline is still capable of completing one instruction every clock period, with an average stage cycle time equal to the global clock period.

Figure 4 illustrates the global voltage control and dynamic retiming support added to the pipeline to implement dynamic retiming. At a regular interval, the global voltage controller samples individual stage error rates. Like the Razor global
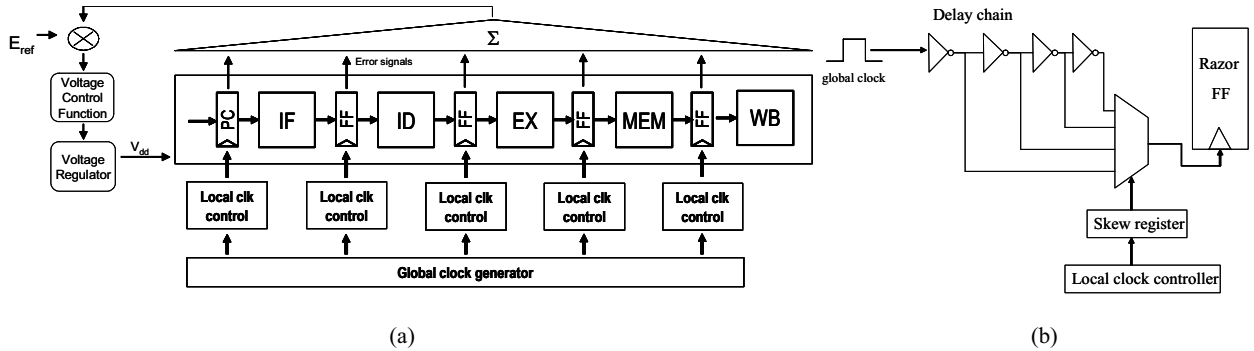
Figure 4: Dynamic retiming implementation

voltage controller, if the total error rate of the pipeline deviates substantially from the reference error rate, E_{ref}, the voltage of the system is changed in proportion to the difference. (If the error rate is too low, voltage is dropped, if the error rate is too high voltage is raised.) In addition, if the error rates of individual stages differ significantly, the pipeline is retimed to allow the stage with the highest error rate to borrow time from the stage with the lowest error rate. Ultimately, the pipeline retiming works to balance the error rate of all stages, which in turn provides opportunity to achieve the lowest possible global voltage.

Figure 4(b) shows the implementation of the local clock controllers. The local clock controllers are programmable delay elements, able to insert positive or negative skew into the arrival time of the clock at pipeline boundaries. The programmable delay elements are implemented with simple inverter chains of varied length fed into a MUX that selects the appropriate delay. In our simulated implementation, the clock controller has 10 delay chains capable of skewing the clock boundary from -1.375ns to +1.375ns in steps of 0.275ns. Clock skew control can be invoked in any clock cycle via memory mapped I/O, however, changes are limited to at most one skew step (0.275ns) each subsequent clock cycle. With this limitation, the programmable clock delay elements can reconfigure in less than one clock cycle and prevent any clock glitches. To simplify the design, both the main flip-flip and Razor shadow latch clocks are skewed in tandem with the same clock delay element. SPICE measurements of the clock control circuits indicate that they add little to overall processor energy. In our test design, the clock delay elements consume less than 0.1% of total chip energy.

3.3 Retiming Design Constraints

The introduction of clock delay elements adds uncertainty (at design time) into main flip-flop and Razor shadow latch clock arrival times. Consequently, this uncertainty must be factored into the Razor design to ensure that a worst-case clock skew scenario does not violate latch setup and hold times. Note that the clock skew of the main and shadow latches are controlled independently, allowing for a better ability to meet the setup and hold constraints. Figure 5 illustrates the timing paths that must be considered when making a guarantee that retiming does not violate the main flip-flip or Razor shadow latch setup and hold times. Note that the skew of the main and shadow latches are controlled and constrained individually, as shown in Table 1. The main flip-flop has a two sided timing constraint. Its skew M_i must be small enough to ensure that the shortpath delay $T_{short(j,i)}$ from flip-flop j to flip-flop i exceeds the hold time (T_{hold}). On the other hand, its skew must be large enough relative to the shadow latch skew SH_i such that there is sufficient time for the restore
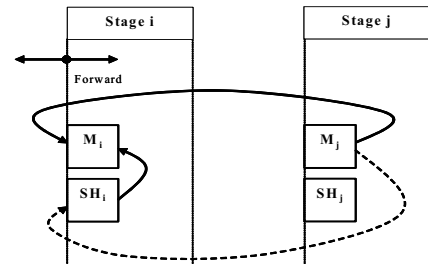

Figure 5: Retiming constraint paths

The dotted line represents the timing constraints on the shadow latch, while the solid line denotes the constraints on the main flip-flop.

Main flip-flop
$T_{restore} - (0.5 * T_{cycle}) + SH_i < M_i < M_j + T_{short(j,i)} - T_{hold}$

Shadow latch
$T_{setup} + M_j + T_{long(j,i)} - (1.5 * T_{cycle})$ $< SH_i <$ $M_j + T_{short(j,i)} - T_{hold} - (0.5 * T_{cycle})$

Table 1: Razor flip-flop retiming constraints

operation ($T_{restore}$) from shadow latch to main flip-flop to complete. Similarly, the shadow latch has a two sided timing constraint. Again, its skew must be small enough to ensure that the short path delay $T_{short(j,i)}$ exceeds the hold time constraint. Note that in this case, a half clock cycle is introduced in the constraint, due to the delayed clock of the shadow latch. Also, the skew must be large enough to ensure that the long path delay $T_{long(j,i)}$ from flip-flop j to flip-flop i meets the setup time of the shadow latch (T_{setup}).

We used PrimeTime [6] to extract the relevant logic path delays from a physical design description of the Razor prototype pipeline. PrimeTime performs static timing analysis to compute worst case delay for any input vector. The maximum skew constraints are maintained by the global voltage controller during run-time, to ensure that main flip-flop and Razor shadow latch setup and hold constraints are never violated.

4. EXPERIMENTAL EVALUATION

4.1 Simulator Framework and Benchmarks

A detailed evaluation of our DVS optimizations requires intimate knowledge of circuit evaluation characteristics, since Razor timing errors are a direct function of circuit-evaluation
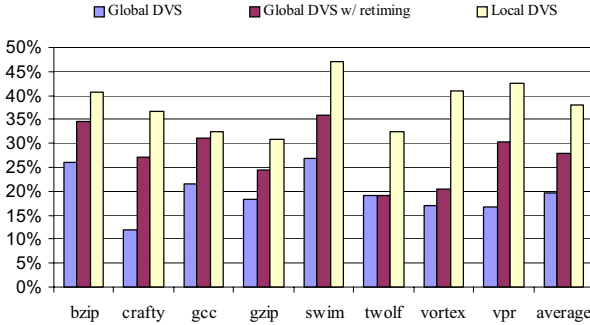
Figure 6: Energy savings over baseline

latency. Typical architectural based simulation methodologies do not have this level of detail. At most, architectural simulators will vary the number of cycles an operation executes based on some model of its circuit complexity, *e.g.,* cache latency vs. size. To support detailed evaluation of a Razor pipeline with local voltage control, we embedded a circuit simulator into our architectural simulator. Our architectural simulator is based on SimpleScalar models [2]. The embedded circuit simulator references a combinational logic description of each relevant component of the architecture under evaluation, and interfaces with the architectural simulator on a stage-by-stage basis.

At initialization, the circuit description of the various components loaded from a structural Verilog netlist. The netlist specifies standard cells and their interconnections with capacitive loadings. Global routing capacitance was estimated by performing global place and route using Cadence Silicon Ensemble (version 5.4.126) and Mentor Graphics Xcalibre (version 9.1 5.6). In addition, a technology model is loaded that details the switching characteristics of the standard cell blocks used in the physical instrumentation. During each simulation cycle, each logic block is fed a new input vector from the architectural simulator. With this information, the circuit simulator can compute the relevant measures for the analysis. A detailed description of our circuit-aware architectural simulation methodology is available in a previous report [3].

To perform our evaluation, we analyze 8 spec2000 benchmarks. For each program, we simulated 10 million instruction samples, selected using the SimPoint tool's "early multiple SimPoint" option [16].

4.2 Energy Savings

We simulated three design variants: original Razor *Global DVS*, *Local DVS*, and *Global DVS with Retiming*. For each design we measured the energy savings over the baseline and the performance impact due to Razor timing error recovery. The baseline pipeline design is the Razor prototype design without Razor support (*i.e.*, fully-margined DVS) running with a fixed supply voltage of 1.8V. All energy measurements are based on circuit-level analyses which include the cost of Razor error recover and clock delay elements.

Figure 6 shows the relative energy savings for the simulated benchmarks. Clearly, *Local DVS* out-performs all other techniques. This is to be expected as this ideal approach to local voltage tuning permits all stages to minimize their energy requirements. Overall, it achieves nearly twice the energy savings of the Razor global DVS, and it finds 38% total reduction in energy compared to fully-margined DVS. *Global DVS with Retiming* sees good gains as well. The approach found
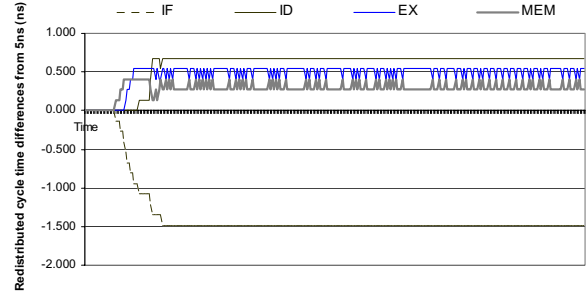

Figure 7: Retiming trace

Each line shows the amount of time borrowed from the other stages. For instance, the EX stage borrows up to 0.545ns from the other stages due to a high local error rate, while the IF stage lends up to 1.5ns to other stages, due to a low local error rate.

a 28% energy savings over the baseline, and it rendered a 12% improvement in energy savings compared to original Razor global DVS. The reduced design cost of dynamicretiming compared to local DVS does come at a reduction in energy savings. Local DVS provides an additional 15% reduction in energy compared to global DVS with retiming.

4.3 Dynamic Retiming Analysis

The graph in Figure 7 shows how dynamic retiming redistributes stage latencies over time for the *GCC* benchmark. Each line shows the amount of clock period increase (above zero) or decrease (below zero) for each of the pipeline stages. Initially, the cycle time of each stages is 5.5ns with 180Mhz clock. Retiming logic monitors the error rate globally and redistributes the available cycle time throughout the stages to minimize local timing error rates. For our prototype design, the clock skew increment is limited to 0.275ns. As shown in the graph, the decode (ID), execute (EX), and memory (MEM) stages all borrow cycle time from the fetch (IF) stage. For this experiment, typically 1.5ns of clock period is taken from IF and distributed to other stages, with the majority going to ID and EX and a lesser amount to MEM.

An interesting characteristic of this trace is the swapping of time between the ID and EX stages over the execution of the program. This redistribution of stage latency is the result of dynamic changes in program execution, which over time causes different circuit paths to be exercised frequently. An implication of this is result is that even perfect balancing of pipelines at design time can lead to run-time imbalances. Thus, well-balanced pipeline designs would likely benefit from some level of support for dynamic retiming as well. Moreover, given the effort and design time required to balance a complex design, it may be prudent to forego this large effort in favor of inclusion of a dynamic retiming capability that balances stage latency at run-time. Finally, we should point out that design topology is not the only source of stage latency imbalance. As silicon geometries shrink, process variation has a greater effect on circuit evaluation latency [17]. At the same time, architects are moving toward longer pipelines, which reduces the amount of logic per stage [15]. The end result of these trends is greater variance in stage latency. Since this variance is introduced at fabrication time, it cannot be mitigated at design time.

4.4 IPC analysis

Fundamental to the Razor technology is a trade-off between energy reduction and error rate. Given this trade-off, it is important to be cognizant of potential performance impacts due

to Razor timing errors, which incur a pipeline flush that reduces pipeline throughput. The original Razor global voltage control algorithm was designed with this concern in mind. The algorithm works to limit the error rate of the pipeline, which is directly proportional to both the energy savings and the pipeline throughput (IPC) impacts. As shown following Table 2, the performance impact of all of the explored control optimization are very close to the performance impact of original Razor global DVS. Although local DVS provides a greater energy reduction, it does come at a slightly greater performance impact. Hence, the choice between local DVS and global DVS with retiming bears a slight dependance on performance demands.

Benchmark	Global DVS	Global DVS w/retiming	Local DVS
Bzip2	2.80%	3.73%	4.58%
Crafty	3.75%	3.13%	5.23%
Gcc	2.83%	2.14%	5.44%
Gzip	1.56%	1.71%	2.64%
Swim	0.48%	0.47%	1.38%
Twolf	0.78%	0.54%	2.40%
Vortex	0.73%	0.73%	1.91%
Vpr	2.19%	2.04%	2.78%
Average	1.89%	1.82%	3.29%

Table 2: IPC degradation of benchmarks (%)

5. RELATED WORK

Njølstad proposed a local DVS technique for the globally-asynchronous and locally-synchronous system (GALS) [12]. They presented a socket interface which permits local dynamic voltage scaling adapted to the processing rate requirement for each module. The module speed is propositional to device speed with the same dependence on local power supply level, process parameters and the temperature variations. Magklis used similar techniques to reduce power in a complex microarchitecture adapted to GALS design [13].

Our timing constraint analysis borrows from Sakallah's work [11]. Sakallah presented a detail timing model to calculate optimal pipeline cycle time. They extensively studied pipeline timing constraints to optimize short and long path propagation. They implement their algorithm by solving a linear program with respect to given timing model.

6. CONCLUSIONS

The quadratic relationship between voltage and energy has made dynamic voltage scaling (DVS) one of the most powerful techniques to reduce system power demands. Razor utilizes in-situ timing error detection and correction mechanisms that eliminate both design- and run-time voltage margins. Razor DVS incorporates a global voltage controller that reduces voltage until error recovery costs outweigh the energy savings of circuit operation at reduced voltage.

The ability of a global voltage controller to shave voltage margins is limited by imbalances in circuit latency within the pipeline design. Since all pipeline stages share the same voltage, the stage exercising the longest critical path will define the overall voltage of the system, even if other stages could potentially run at lower voltages. In this paper, we evaluate two local tuning mechanisms in the context of Razor DVS. A local voltage controller scheme is evaluated that allows each pipeline stages to run at its own voltage level. While an ideal technique to minimize energy, local voltage control suffers from high design costs, including level converters and complex voltage regulation. To achieve the benefits of local DVS with lower cost, we propose a novel dynamic retiming scheme. Dynamic retiming

incorporates per-stage clock delay elements that allow longer-latency stages to "borrow" time from shorter-latency stages.

Using simulation, we draw two key insights from our analysis. First, eliminating voltage margins due to imbalances in pipeline stage latency renders additional energy savings. A Razor pipeline design with dynamic retiming finds an additional 12% energy savings over global voltage control (resulting in an overall energy savings of more than 28% compared to traditional fully-margined DVS). Second, we see that imbalances arise not only from design factors, but also from run-time characteristics. As the program (or program phase) changes, we see logic paths in different stages exercised frequently, necessitating a dynamic fine-tuning of local control. This result suggests that even well-balanced pipelines could benefit from dynamic retiming. Alternatively, the costly phase of design to balance pipelines could be mitigated with a dynamic retiming capability.

Finally, it is important to note that design topology is not the only source of pipeline imbalance. As process geometries decrease in size, there is significantly greater uncertainty in circuit evaluation latency due to process variation [14]. At the same time, architects are moving toward longer pipelines with less logic per stage [15]. The end result of these trends is greater variance in per-stage latency. Since much of this variance is introduced at fabrication time, it cannot be "designed away". Hence, this trend will further reinforce the need for tuning techniques like local DVS and dynamic retiming.

7. REFERENCES

[1] D. Ernst, N. 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", *MICRO-36,* December 2003.

[2] T. Austin, E. Larson, D. Ernst. "SimpleScalar: an Infrastructure for Computer System Modeling", *IEEE Computer*, 35 (2), February 2002.

[3] Seokwoo Lee, Shidhartha Das,Valeria Bertacco, Todd Austin, David Blaauw, and Trevor Mudge. "Circuit-Aware Architectural Simulation", *41st Design Automation Conference (DAC),* June, 2004.

[4] D. Harris. "Skew-Tolerant Circuit Design",*Morgan Kaufman Publishers* 2001.

[5] K. Bernstein, *et al. "*High Speed Cmos Design Styles"*, Kluwer Academic Publishers,* 1999.

[6] Synopsis Corporation, "PrimeTime", http://www.synopsys.com/products/analysis/analysis.html.

[7] T. Pering. et,al "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms" *Proceedings of Int'l Symposium on Low Power Electronics and Design 1998*, pp. 76-81, June 1998.

[8] T. Mudge. "Power: A first class design constraint", *Computer*, vol. 34, no. 4, April 2001, pp. 52-57.

[9] T. Burd. et, al "A Dynamic Voltage Scaled Microprocessor System", *IEEE Journal of Solid-State Circuits,* Vol 35, No. 11*,* November 2000.

[10] A. Dancy, R. Amirtharajah, and A. P. Chandrakasan, "High Efficiency Multiple Output DC-DC Conversion for Low-Voltage Systems", *IEEE Trans.on Very Large Scale Integration (VLSI) Systems*, pp. 252-263, June 2000.

[11] K. Sakallah, T. Mudge, and O. Olukotun. "checkTc and MinTc: Timing Verification and Optimal Clocking of Synchronous Digital Circuits", *1990 IEEE.*

[12] T. Njølstad. et.al "A Socket Interface For GALS Using Locally Dynamic Voltage Scaling For Rate-Adaptive Energy Saving", *IEEE 2001.*

[13] M. Semeraro. et, al "Dynamic Frequency and Voltage Scaling for a Multiple-Clock-Domain Microprocessor", *IEEE Micro,* Special Issue on Power-Aware Issue on the Top Picks from Microarchitecture Conference, Vol 36, No.12.

[14] A. Agarwal, D. Blaauw, V. Zolotov, "Statistical Timing Analysis for Intra-Die Process Variations with Spatial Correlations", *ACM/IEEE International Conference on Computer-Aided Design (ICCAD)*, November 2003.

[15] V. Agarwal, M.S. Hrishikesh, S. Keckler, D. Burger, "Clock Rate versus IPC: The End of the Road for Conventional Microarchitectures", *ISCA-2000.*

[16] T. Sherwood, E. Perelman, G. Hamerly and B. Calder, "Automatically Characterizing Large Scale Program Behavior", *ASPLOS-X*, October 2002.

[17] R. Gonzalez, B. Gordon, and M. Horowitz, "Supply and Threshold Voltage Scaling for Low Power CMOS", *IEEE JSSC*, 32 (8), August 1997.