A Self-Tuning DVS Processor Using Delay-Error Detection and Correction

Shidhartha Das, *Student Member, IEEE*, David Roberts, *Student Member, IEEE*, Seokwoo Lee, Sanjay Pant, David Blaauw, *Member, IEEE*, Todd Austin, Krisztián Flautner, *Member, IEEE*, and Trevor Mudge, *Fellow, IEEE*

***Abstract*—In this paper, we present a dynamic voltage scaling (DVS) technique called Razor which incorporates an *in situ* error detection and correction mechanism to recover from timing errors. We also present the implementation details and silicon measurements results of a 64-bit processor fabricated in 0.18-μm technology that uses Razor for supply voltage control. Traditional DVS techniques require significant voltage safety margins to guarantee computational correctness at the worst case combination of process, voltage and temperature conditions, leading to a loss in energy efficiency. In Razor-based DVS, however, the supply voltage is automatically reduced to the point of first failure using the error detection and correction mechanism, thereby eliminating safety margins while still ensuring correct operation. In addition, the supply voltage can be intentionally scaled below the point of first failure of the processor to achieve an optimal tradeoff between energy savings from further voltage reduction and energy overhead from increased error detection and correction activity. We tested and measured savings due to Razor DVS for 33 different dies and obtained an average energy savings of 50% over worst case operating conditions by scaling supply voltage to achieve a 0.1% targeted error rate, at a fixed frequency of 120 MHz.**

***Index Terms*—Dynamic voltage scaling (DVS), error detection and correction, self-tuning processor, voltage safety margins.**

I. Introduction

THE tremendous boost in microprocessor performance enabled by technology scaling has come at the price of ever increasing power consumption. Power budgets are even more stringent for battery-operated embedded processors which handle a broad spectrum of applications with diverse energy and performance requirements [7], [14]. Dynamic voltage scaling (DVS) is a widely used technique to reduce the overall energy consumption of a processor, especially under wide workload variations. In a DVS system, the supply voltage and operating frequency are dynamically adjusted according to application demands. Due to the quadratic dependence of energy with supply voltage [12], significant energy savings are achievable with DVS.

A critical issue for a DVS-enabled processor is determining the safe operating voltage under which energy savings are maximized while guaranteeing correct operation under all conditions. Traditional techniques [2]–[6] described in literature use a delay chain to determine the minimum voltage necessary for error-free operation at a particular frequency. The delay chain replicates the worst case critical path of the chip with additional latency margins. Design time characterization of the critical path determines the margins that need to be added in order to ensure that the replica delay path is guaranteed to fail before the core does even in the presence of a worst case combination of inter- and intra-die process variations, temperature hot spots, and supply voltage uncertainties. The supply voltage is then lowered to the point where the delay chain just fails to meet timing. As silicon predictability reduces with technology scaling, the safety margins are likely to increase [13]. This leads to overly conservative operation given the extremely rare occurrence of worst case conditions [1]. Significantly greater energy savings can be achieved with DVS by scaling the supply voltage below the "always correct" voltage level dictated by safety margins and using an efficient mechanism to recover from rare worst case errors.

We proposed a novel voltage management technique for DVS processors, called Razor [1], which uses a delay-error tolerant flip-flop on critical paths to scale the supply voltage to the point of first failure for a given frequency. This allows voltage margins to be eliminated, resulting in significant energy savings. In addition, Razor allows the supply voltage to be scaled even lower than the first failure point into the subcritical region, deliberately tolerating a targeted error rate, thereby providing additional energy savings.

The operational principle of Razor is illustrated in Fig. 1 which shows the qualitative relationship between the supply voltage, energy consumption and pipeline throughput of a Razor-enabled processor. The point of first failure of the processor (V_{ff}) and the minimum allowable voltage of traditional DVS techniques (V_{margin}) are also labeled in the figure. V_{margin} is much higher than V_{ff} under typical conditions, since safety margins need to be included to accommodate for worst case operating conditions. Razor relies on *in situ* error detection and correction capability to operate at V_{ff}, rather than at V_{margin}. The total energy of the processor (E_{tot}) is the sum of the energy required to perform standard processor operations (E_{proc}) and the energy consumed in recovery from timing errors (E_{recovery}). Of course, implementing Razor incurs power overhead due to which the nominal processor energy (E_{nom}) *without* Razor technology is slightly less than E_{proc}. This overhead is attributed to the use of delay-error tolerant flip-flops on the critical paths and the additional recovery logic required for Razor. However, since the extra circuitry is deployed only for those flip-flops which have critical paths terminating in them, the power overhead due to Razor is fairly minimal. In the

Manuscript received September 5, 2005; revised December 19, 2005.

S. Das, D. Roberts, S. Lee, S. Pant, D. Blaauw, T. Austin, and T. Mudge are with the University of Michigan, Ann Arbor, MI, USA, 48109 USA (e-mail: siddas@umich.edu).

K. Flautner is with ARM Ltd., Cambridge CB1 9NJ, U.K.

Digital Object Identifier 10.1109/JSSC.2006.870912

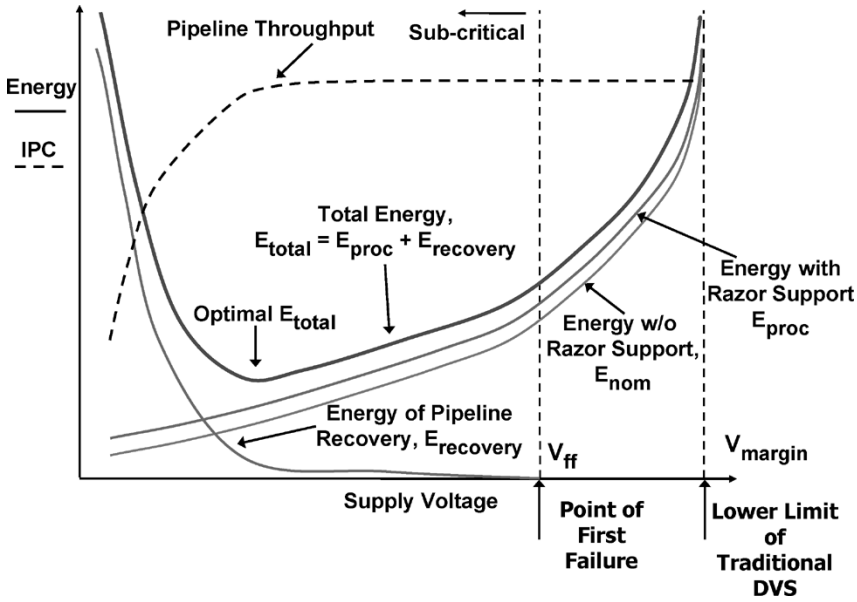


Fig. 1. Qualitative relationship between supply voltage, energy, and IPC.

processor that we present in this paper, only 7.4% of the total flip-flops were critical and needed Razor recovery protection. The net power overhead due to Razor was less than 3% of the nominal chip power.

As the supply voltage is scaled, the processor energy $(E_{\rm proc})$ reduces quadratically with voltage. However, as voltage is scaled below the first failure point $(V_{\rm ff})$, a significant number of paths fail to meet timing. Hence, the error rate and the recovery energy $(E_{\rm recovery})$ increase exponentially. The processor throughput also reduces due to the increasing error rate because the processor now requires more cycles to complete the instructions. The total processor energy $(E_{\rm tot})$ shows an optimal point where the rate of change of $E_{\rm recovery}$ and $E_{\rm proc}$ offset each other. Thus, in the context of Razor, a timing error is not a catastrophic failure but a tradeoff between the quadratic energy savings due to voltage scaling versus the overhead of recovery due to errors.

In this paper, we present the first silicon implementation of a Razor design [11]. We discuss the circuit structures used in this new implementation and present silicon measurements for 33 tested dies. The 64-bit processor implements a subset of the Alpha instruction set and was fabricated with MOSIS [10] in an industrial 0.18-μm technology. Voltage control is based on the observed error rate and power savings are achieved by: 1) eliminating the safety margins under nominal operating and silicon conditions and 2) scaling voltage 120 mV below the first failure point to achieve a 0.1% targeted error rate. We tested and measured savings due to Razor DVS for 33 different dies and obtained an average energy savings of 50% over the worst case operating conditions by operating at the 0.1% error rate voltage, at a fixed frequency of 120 MHz.

The remainder of this paper is organized as follows. In Section II, we give an overview of Razor. Section III describes the transistor level design and the operational details of the delay-error tolerant Razor flip-flop. Section IV discusses the processor implementation details. We present our measurement results in Section V and discuss the Razor voltage control scheme in Section VI. Finally, we offer concluding remarks in Section VII.

II. Razor Overview

Fig. 2(a) shows the conceptual representation of the delay-error tolerant Razor flip-flop (henceforth referred to as the RFF) and timing diagrams that explain its working principle. The standard positive edge triggered D-flip-flop (DFF) is augmented with a shadow latch which is transparent in the positive phase of the clock and samples at the negative edge. Thus, the input data is given additional time, equal to the duration of the positive clock phase, to settle down to its correct state before being sampled by the shadow latch. In order to ensure that the *shadow* latch always captures the correct data, the minimum allowable supply voltage needs to be constrained during design time such that the setup time at the shadow latch is never violated even under worst case conditions. A comparator flags a timing error when it detects a discrepancy between the speculative data sampled at the main flip-flop and the correct data sampled at the shadow latch. This is illustrated in Fig. 2(b) where the RFF input (D_in) transitions after the positive clock edge in cycle 2 causing the state captured at the shadow latch $(Q_{\rm shadow})$ to be different from that captured at the main flip-flop (Q). This leads to the $error$ signal being flagged.

Error signals of individual RFFs are OR-ed together to generate the pipeline $restore$ signal which overwrites the shadow latch data into the main flip-flop, thereby restoring correct state in the cycle following the errant cycle. Thus, an errant instruction is guaranteed to recover with a single cycle penalty, without having to be re-executed. This ensures that forward progress in the pipeline is always maintained. Even if every instruction fails to meet timing, the pipeline still completes, albeit at a slower speed. Upon detection of a timing error, a micro-architectural recovery technique is engaged to restore the whole pipeline to its correct state.

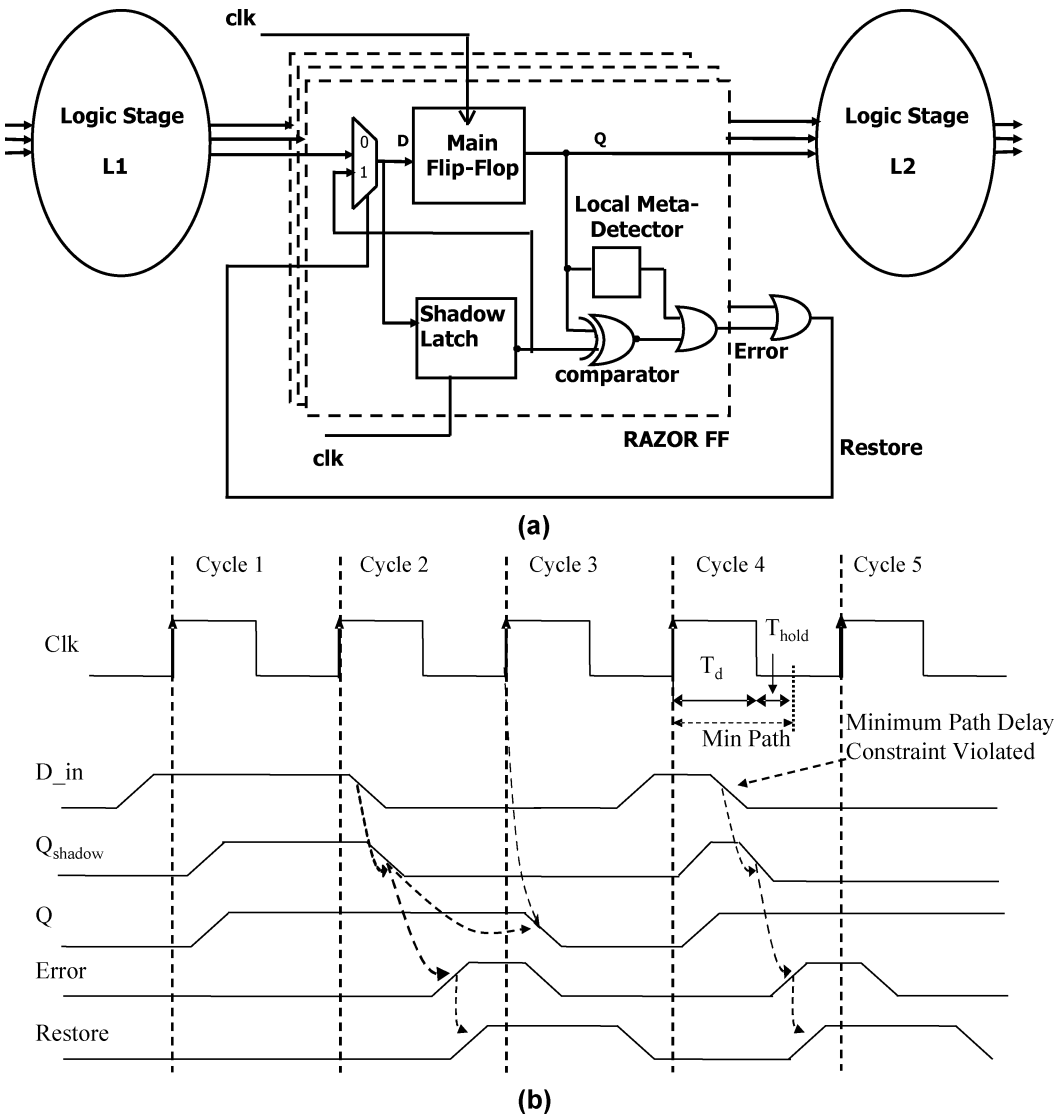


Fig. 2. Abstract view of the Razor flip-flop and conceptual timing diagrams.

Since setup and hold constraints at the main flip-flop input (D_in) are not respected, it is possible that the state of the flip-flop becomes metastable. A metastable signal increases critical path delay which can cause a shadow latch in the succeeding pipeline stage to capture erroneous data, thereby leading to incorrect execution. In addition, a metastable flip-flop output can be inconsistently interpreted by the error comparator and the downstream logic. Hence, an additional detector is required to correctly flag the occurrence of metastability at the output of the main flip-flop. The outputs of the metastability detector and the error comparator are ORed to generate the *error* signal of the RFF. Thus, the system reacts to the occurrence of metastability in exactly the same way as it reacts to a conventional timing failure.

A key point to note is the fact that metastability *need not be resolved correctly* in the RFF and that just the *detection* of such an occurrence is sufficient to engage the Razor recovery mechanism. However, in order to prevent potentially metastable signals from being committed to memory, at least two successive noncritical pipeline stages are required immediately before storage. This ensures that every signal is validated by Razor and is effectively double-latched in order to have a negligible probability of being metastable, before being written to memory. In our design, data accesses in the Memory stage were noncritical and hence we required only one additional pipeline stage to act as a dummy stabilization stage.

Using the negative edge of the clock as the sampling trigger for the shadow latch precludes the need for an additional clock tree. This simplifies implementation because only a single clock is required and prevents the excessive overhead of routing a second clock tree just for the purposes of clocking the shadow latch in the RFFs. The duration of the positive clock phase, when the shadow latch is transparent, determines the sampling delay of the shadow latch. This constrains the minimum propagation delay for a combinational logic path terminating in an RFF to be at least greater than the duration of the positive clock phase and the hold time of the shadow latch.

Fig. 2(b) conceptually illustrates this minimum delay constraint. In cycle 4, the RFF input, D_in, violates this constraint and changes state before the negative edge of the clock, thereby

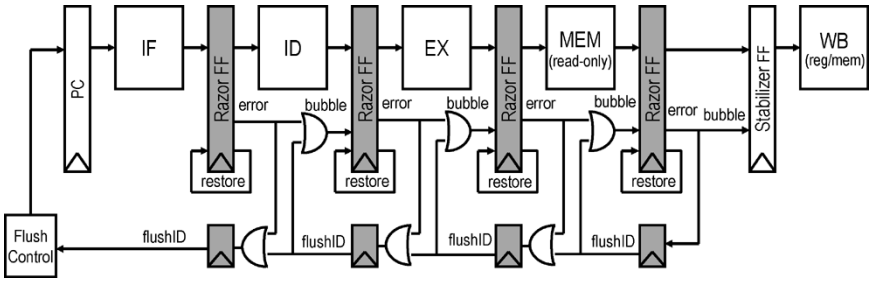


Fig. 3. Distributed pipeline recovery mechanism.

corrupting the state of the shadow latch. Delay buffers are required to be inserted in those paths which fail to meet this minimum path delay constraint imposed by the shadow latch. The insertion of delay buffers incurs power overhead because of the extra capacitance added. A large shadow latch sampling delay requires a greater number of delay buffers to be inserted, thereby increasing the power overhead. However, a small sampling delay implies that the voltage difference between the point of first failure and the point where shadow latch fails is less and, thus, reduces the voltage margin available through Razor timing speculation. Hence, the shadow latch sampling delay represents the tradeoff between power overhead due to delay buffers and the voltage margin available for Razor subcritical mode of operation. Using suitable clock chopping techniques, the duration of the positive phase of the propagated clock can be configured as required so as to exploit the above tradeoff.

A key point to note is the fact that the hold constraint imposed by the shadow latch only limits the *maximum duration of the positive clock phase* and has no bearing upon the clock frequency. Thus, a "Razor"-ed pipeline can still be operated at any frequency as required as long as the positive clock phase is sufficient to meet the minimum path delay constraint. In our design, for a sampling delay of 3.0 ns which is approximately half the cycle time at 140 MHz, it was required to add 2388 delay buffers to satisfy the short path constraint on 207 RFFs (7.4% of the total number of flip-flops). The power overhead due to these buffers was less than 3% of the nominal chip power.

Correct pipeline state is recovered in the event of a timing error by engaging a distributed pipeline recovery mechanism, as described in [1], which is based on a counter-flow pipeline architecture [9]. The primary requirement of the recovery mechanism is to prevent corrupt state being committed to storage in memory or the register file before being validated by Razor. In [1], we have discussed two possible ways in which this can be achieved. A centralized pipeline recovery mechanism uses the $restore$ signal as a global clock-gating signal to stall the pipeline for a single cycle while the errant flip-flop recovers correct state. This incurs only a one-cycle recovery penalty but imposes significant timing restrictions on the $restore$ signal which needs to be distributed through the entire chip in less than one cycle. In contrast, the distributed pipeline recovery mechanism places negligible restrictions on the cycle time at the expense of extending recovery over several cycles.

Fig. 3 conceptually illustrates the working principle of the distributed pipeline recovery mechanism. When a Razor error occurs, two actions are taken. First, the computation in the stage following the errant stage is nullified by a "bubble" signal which indicates to the next and subsequent stages that the pipeline slot is invalid. Second, a backward propagating flush train is triggered by asserting the stage identifier (ID) of the failing stage. In the following cycle, the correct value from the Razor shadow latch data is injected back into the pipeline, allowing the errant instruction to continue with its correct inputs. In addition, the flush train begins propagating the ID of the failing stage in the opposite direction of instructions. At each stage, the flush train inserts a bubble in the corresponding pipeline stage as well as in the immediately preceding stage. (Two stages must be nullified because the main pipeline appears to move twice as fast relative to the flush train.) When the flush ID reaches the start of the pipeline, the flush control logic restarts the pipeline at the instruction following the errant instruction. In the event that multiple stages experience errors in the same cycle, all will initiate recovery but only the Razor error closest to write-back (WB) will complete. Earlier recoveries will be flushed by later ones.

III. Transistor-Level Design of the RFF

Fig. 4 shows the transistor level circuit schematic of the RFF. In the absence of a timing error, the RFF behaves as a standard positive edge triggered flip-flop. The error comparator is a semi-dynamic XOR gate which evaluates when the data latched by the slave differs from that of the shadow in the negative clock phase. The error comparator shares its dynamic node Err_dyn with the metastability detector which evaluates in the positive phase of the clock when the slave output could become metastable. Thus, the RFF $error$ signal is flagged when either the metastability detector or the $error$ comparator evaluate.

This, in turn, evaluates the dynamic gate to generate the $restore$ signal by ORing together the error signals of individual RFFs (Fig. 5), in the negative clock phase. The $restore$ signal incurs significant routing and gate capacitance as it is routed to every flip-flop in the pipeline stage and needs to be driven by strong drivers. For an RFF, the $restore$ serves to overwrite the master with the shadow latch data. Hence, the slave gets the correct data at the next positive edge.

The $restore$ needs to be latched at the output of the dynamic OR gate so that it retains state during the next positive phase (recovery cycle) during which it disables the shadow latch to protect state. In addition, the $restore$ also disables all regular, non-"Razor"-ed flip-flops in the pipeline stage to preserve the state that was latched in the errant cycle. This is required to maintain the temporal consistency of all flip-flops in the pipeline stage. The stack of three pMOS transistors in the shadow latch

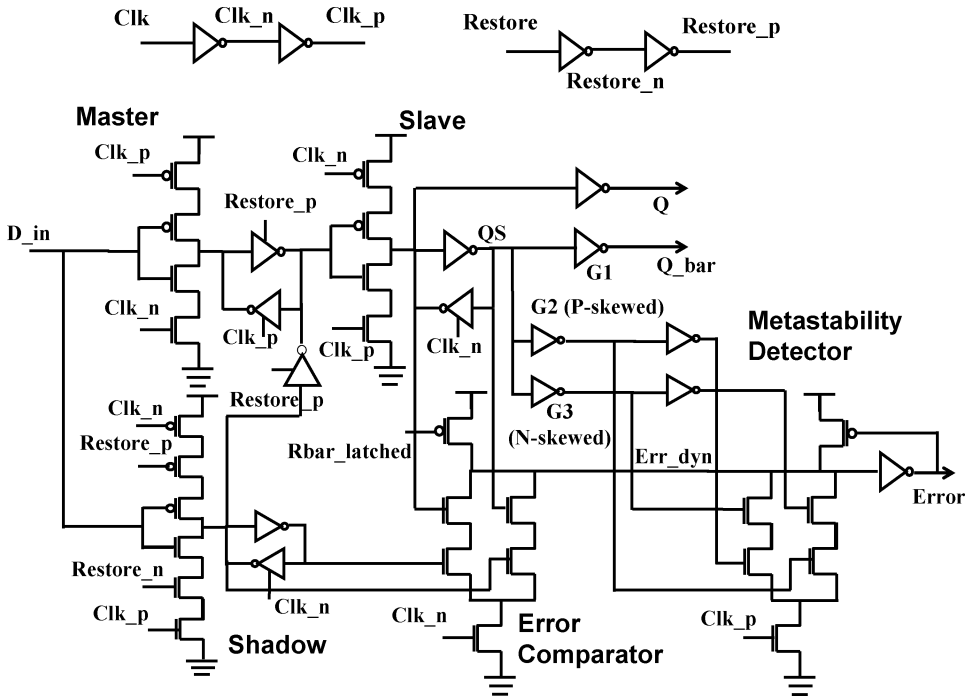


Fig. 4. Circuit schematic of the Razor flip-flop.

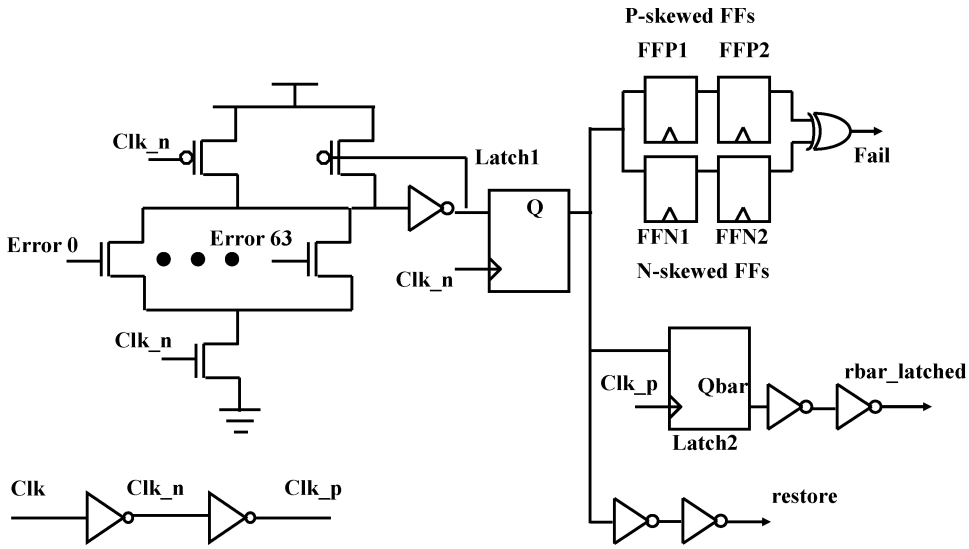


Fig. 5. Restore generation circuitry.

increases its setup time. However, the shadow latch is required only for runtime validation of the main flip-flop data and does not form a part of the critical path of the RFF.

The $rbar\_latched$ signal, shown in the restore generation circuitry in Fig. 5, which is the half-cycle delayed and complemented version of the $restore$ signal, precharges the Err_dyn node for the next errant cycle. Thus, unlike standard dynamic gates where precharge takes place every cycle, the Err_dyn node is conditionally precharged in the recovery cycle following a Razor error. Precharge can take place without contention because in this cycle the slave latch has exactly the same data as the shadow latch and is guaranteed not to be metastable. Hence, neither the error comparator nor the metastability detector evaluates. A weak pMOS half-latch protects Err_dyn from discharge due to leakage.

The RFF was compared with a standard DFF for power consumption. Both are designed for the same delay (clk-q delay + setup time) and drive strength. The characterization setup consists of the flip-flop under test driving a fanout-of-four (FO4) capacitive load. The clock and the input data are each driven by signals with a 100-ps transition time and with sufficient delay between transitions on the data and the clock so as not to violate setup time. The RFF was found to consume 22% extra (60 fJ/49 fJ) energy when the sampled data does not change state and 65% extra (205 fJ/124 fJ) energy when sampled data switches. However, in the processor only 207 flip-flops out of

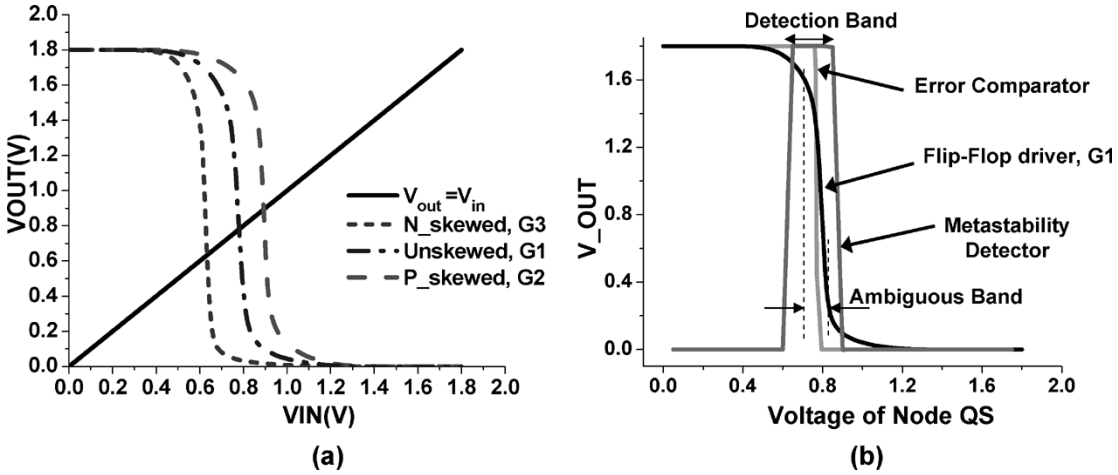


Fig. 6. Metastability detector: principle of operation.

2801 flip-flops, or 7.4%, had critical paths terminating in them and needed use of RFFs.

The measured power of the processor at 120 MHz at 25 °C for a supply voltage of 1.8 V was 130 mW. A simulation-based power analysis was performed to compute the power overhead of the RFFs and the delay buffers required to meet the short path constraint. For a conservative activity factor of 20%, the net power overhead due to RFFs was 0.31% and that due to delay buffers was 2.6%. Thus, the total power overhead due to Razor was computed to be less than 3% of the nominal chip power. Thus, most of the additional power due to Razor is attributed to the delay buffers added for meeting the short path constraint.

TABLE I
METASTABILITY DETECTOR CHARACTERISTICS

Corner			Ambiguous Band	Detection Band
Proc	VDD	TEMP		
Slow	1.2V	85C	0.57-0.60	0.53-0.64
Typ.	1.2V	40C	0.52-0.58	0.48-0.61
Fast	1.2V	27C	0.48-0.56	0.40-0.61
Slow	1.8V	85C	0.77-0.87	0.67-0.93
Typ.	1.8V	40C	0.71-0.83	0.65-0.90
Fast	1.8V	27C	0.64-0.81	0.58-0.89

A. Metastability detection

As was mentioned in Section II, metastability can potentially cause incorrect execution because of inconsistent interpretation and increase in propagation delay. Therefore, we perform metastability detection at the RFF node QS (as labeled in Fig. 4) because QS fans out to the flip-flop driver $G1$ and the error comparator and thus, directly affects the RFF outputs, namely Q and $error$.

Fig. 6 illustrates the operating principle and characteristics of the metastability detector. The metastability detector consists of a p-skewed inverter $G2$ and an n-skewed inverter $G3$ (as labeled in Fig. 4) which switch to opposite power rails under a meta-stable input voltage such that a dynamic comparator can evaluate and latch the comparison result. Fig. 6(a) shows the DC transfer characteristics of the skewed inverters compared to that of the driver inverter, $G1$. The switching points are denoted as the points where the 45 degree line intersects the DC transfer curves. We note that the switching points for the p-skewed inverter and the n-skewed inverter lie on either side of that for $G1$. During normal operation, when the output of the main flip-flop is logically well defined, the output of $G2$ and $G3$ match. Thus, the comparator does not evaluate and the dynamic node is not discharged. However, when QS is metastable at approximately VDD/2, the output of the p-skewed inverter $G2$ is at a voltage level near VDD and the output of the n-skewed inverter $G3$ is near ground. This causes the comparator to evaluate and discharge the dynamic node, Err_dyn, thereby flagging the $error$ signal.

It is imperative that the metastability detector is guaranteed to evaluate for a voltage range of the input node QS for which the fan-out of QS, namely the error comparator and the flip-flop driver $G1$, have either logically undefined or logically inconsistent outputs. This "ambiguous" band of voltage is defined as the voltage range for which the outputs of either $G1$ or the error comparator are in between 10% to 90% of VDD. The range of voltage for which the metastability detector *actually* evaluates is defined to be the "detection" band of voltage. Fig. 6(b) shows the DC transfer curve of inverter $G1$, the error comparator and the metastability detector. As is clearly shown in the figure, the "ambiguously" interpreted voltage band is contained well within the "detection" band. As shown in Table I, the "detection" band subsumes the "ambiguous" band across different process, voltage and temperature (PVT) corners to ensure correct operation under all conditions.

There is a certain delay between QS becoming metastable and the detector correctly flagging such an occurrence. If QS remains metastable for a very small duration of time, shorter than the evaluation delay through the detector, then the dynamic node Err_dyn is not discharged completely and hence the $error$ signal can become metastable. A key point to note in this case is that when the $error$ signal itself becomes metastable, the actual RFF output is already resolved and hence is not metastable. Such a situation, therefore, does not constitute an actual failure.

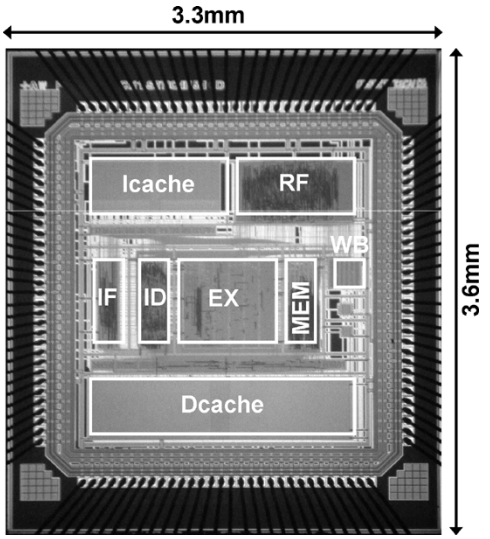


Fig. 7. Die photograph of the processor.

However, a metastable $error$ signal can potentially propagate through the $restore$ generation logic and cause unpredictable behavior of the pipeline recovery infrastructure. This can corrupt the processor state. Since the $error$ signal goes through intermediate logic gates and thus through several stages of gain until $restore$ generation takes place, it is very unlikely that metastability at the $error$ signal can propagate to cause metastability at the $restore$ node.

The probability of the $restore$ node becoming metastable was computed to be less than 2e-30 [8]. Despite this being a sufficiently low probability, the unlikely event of this happening is detected by means of skewed flip-flops, as shown in Fig. 5. A p-skewed flip-flop and an n-skewed flip-flop resolve a metastable input to opposite power rails such that an XOR comparator can detect the discrepancy by flagging the $fail$ signal. The outputs of the skewed flip-flops are latched before being compared so that the $fail$ signal itself has negligible probability of being metastable. In the event of $fail$ being flagged, the entire pipeline is flushed and the failed instruction is re-executed. Since forward progress is violated in this case, the supply voltage is immediately increased to ensure that the failed instruction completes. During the four months of chip testing, such an event was never detected.

IV. Razor Processor Design

We designed a 64-bit microprocessor implementing the Alpha instruction set with Razor-based dynamic voltage management. The processor was fabricated in a 0.18-μm industrial technology. The die photograph and the relevant implementation details are shown in Fig. 7 and Table II, respectively. The architectural state of the processor is observable and controllable by three separate scan chains for each of the Icache, Dcache, and the register file. The chip was tested by scanning in instructions into the Icache and comparing the execution output scanned out of the Dcache and the register file with a personal computer emulating the same code. A 64-bit special purpose register keeps a record of the total number of errant cycles and is sampled to compute the error rate for a particular run.

TABLE II
Processor Implementation Details

Technology Node	0.18µm
Max. Clock Frequency	140MHz
DVS Supply Voltage Range	1.2-1.8V
Total Number of Transistors	1.58million
Die Size	3.3mm*3.6mm
Measured Chip Power at 1.8V	130mW
Icache Size	8KB
Dcache Size	8KB
Total Number of Flip-Flops	2801
Total Number of Razor Flip-Flops	207
Number of Delay Buffers Added	2388
% Total Chip Power Overhead due to Razor Flip-Flops and Delay Buffers	2.9%
Error Free Operation	
Standard Flip-Flop Energy (static/switching)	49fJ/125fJ
RFF Energy (static/switching)	60fJ/205fJ
Error Detection and Recovery Overhead	
Energy of RFF per error event	260fJ

The core frequency is controlled by an internal clock generation unit (CGU). The CGU generates an asymmetric clock in a range between 60 and 400 MHz in steps of 20 MHz. The shadow latch sampling delay, defined by the duration of the positive clock phase, is configurable from 0 to 3.5 ns in steps of 500 ps. The CGU has a separate voltage domain that is not voltage scaled. Hence, the core frequency and the shadow latch sampling delay remains constant even when the core voltage is dynamically scaled.

For the current implementation, we designed an off-chip hardware loop for supply voltage control. The controller samples the error register and accordingly adjusts the supply voltage through an external voltage regulator. We report the energy consumed by the processor only, not including the external regulator. However, supply voltage control can be achieved in software by means of a subroutine that reads the error accumulator register, implements the control algorithm, and interfaces with a regulator to adjust the voltage. An on-chip voltage regulator can be designed such that the entire voltage control loop is internally located.

V. Measurement Results

We measured energy savings obtainable from Razor DVS at 140 and 120 MHz for 33 chips from two different fabrication runs. As mentioned, Razor energy savings are due to both elimination of voltage safety margins and operation below the point of first failure in the subcritical voltage regime. For every chip, we quantified the safety margin due to inter-die process variations by measuring the difference between the first failure point

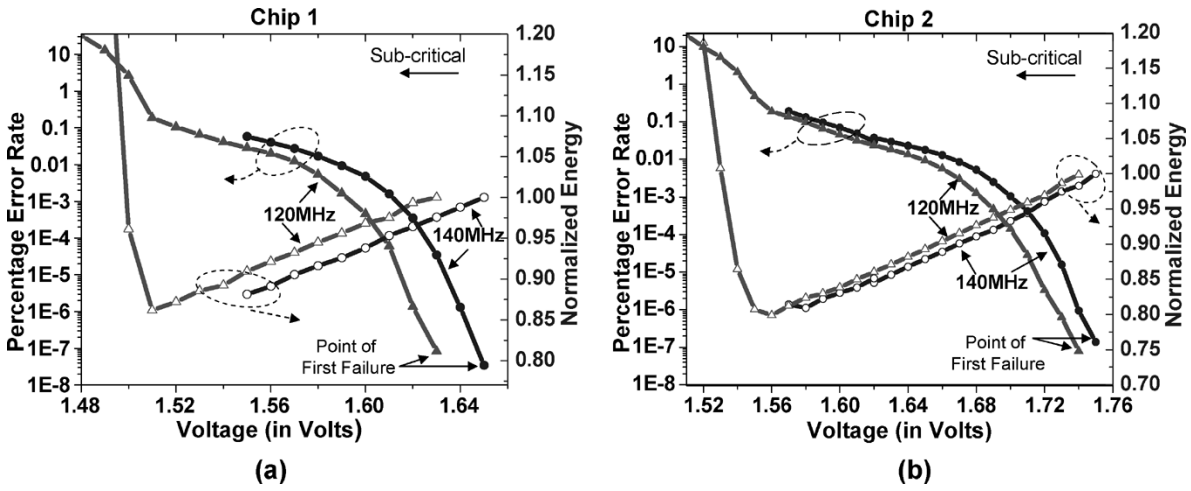


Fig. 8. Error rate and normalized energy measurement for chip 1 and chip 2.

TABLE III
ERROR RATE AND ENERGY/INSTRUCTION AT POINT OF FIRST FAILURE AND POINT OF 0.1% ERROR RATE FOR CHIPS 1 AND 2

	Point of First Failure			Point of 0.1% Error Rate		
	Voltage	Power	Energy per Instruction (Power/IPC /Freq)	Voltage	Power	Energy per Instruction (Power/IPC /Freq)
Chip1	1.63V	104.5mW	870pJ	1.52V	89.7mW	740pJ
Chip2	1.74V	119.4mW	990pJ	1.58V	99.6mW	830pJ

of the slowest (worst case process corner) chip and the chip under test. Temperature margins were computed by the shift in the first failure point for a chip when operating at 105 °C as opposed to operating at 25 °C. In addition, by scaling the supply voltage below the first failure point, we measured the minimum voltage for which error correction is achievable with Razor and the voltage where a 0.1% error rate is attained.

A. Energy Savings From Sub-Critical Operation

Fig. 8 shows the error rates and normalized energy savings versus supply voltage at 120 and 140 MHz for two different chips. Energy at a particular voltage is normalized with respect to the energy at the point of first failure. For all plotted points, correct program execution with Razor error correction was verified.

From Fig. 8, we note that the error rate at the point of first failure is very low, and is on the order of 1.0e-8, because only a few critical paths that are rarely sensitized fail to meet setup requirements and are flagged as timing errors. As voltage is scaled further into the subcritical regime the error rate increases exponentially. The instruction per cycle (IPC) penalty due to the error recovery cycles is negligible for error rates below 0.1%. Under such low error rates, the recovery overhead energy is also negligible and the total processor energy shows a quadratic reduction with the supply voltage. At error rates exceeding 0.1%, the recovery energy rapidly starts to dominate, offsetting the quadratic savings due to voltage scaling. For the measured chips, the energy optimal error rate fell at approximately 0.1%.

Table III shows the measured power at the point of first failure and the energy per instruction for both the chips at the point of first failure and at the point of 0.1% error rate. At 120 MHz, chip 1 consumes 104.5 mW at the first failure point and 89.7 mW at an optimal 0.1% error rate, leading to 14% energy savings with negligible IPC hit. The energy saving for chip 2 is 17%. These savings are in addition to the energy saved just by eliminating voltage margins. Fig. 9 shows the distribution of the percentage normalized energy savings obtained over the first failure point while operating at the 0.1% error rate voltage for all the chips tested. At 120 MHz, the range extends from 5% to 23% and from 5% to 19% at 140 MHz.

Fig. 10(a) shows the distribution of the first failure voltage for the 33 measured chips. At 120 MHz, the measured range of variation of the first failure point is from 1.46 to 1.76 V. The correlation between the first failure voltage and the 0.1% error rate voltage is shown in the scatter plot of Fig. 10(b). The 0.1% error rate voltage shows a net variation of 0.24 V from 1.38 to 1.62 V which is approximately 20% less than the variation observed for the voltage at the point of first failure. The relative "flatness" of the linear fit indicates less sensitivity to process variation when running at a 0.1% error rate than at the point of first failure. This implies that a Razor-enabled processor, designed to operate at the energy optimal point, is likely to show greater predictability in terms of performance than a conventional worst case optimized design. The energy optimal point requires a significant number of paths to fail and statistically averages out the variations in path delay due to process variation, as opposed to the

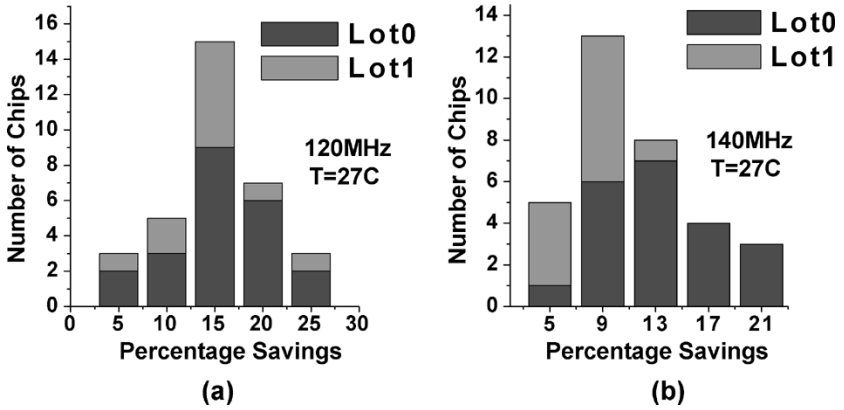


Fig. 9. Distribution of normalized energy savings over first failure point at 0.1% error rate for 33 measured chips.

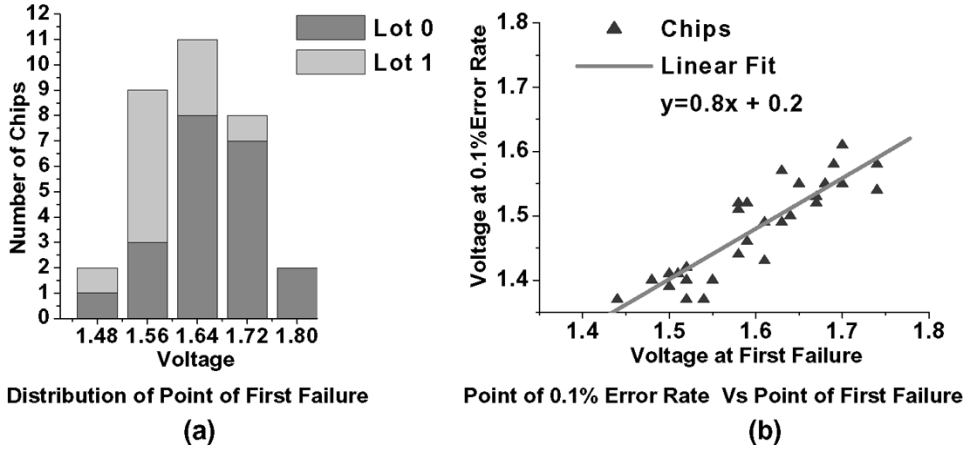


Fig. 10. Distribution of point of first failure and point of 0.1% error rate for 33 measured chips.

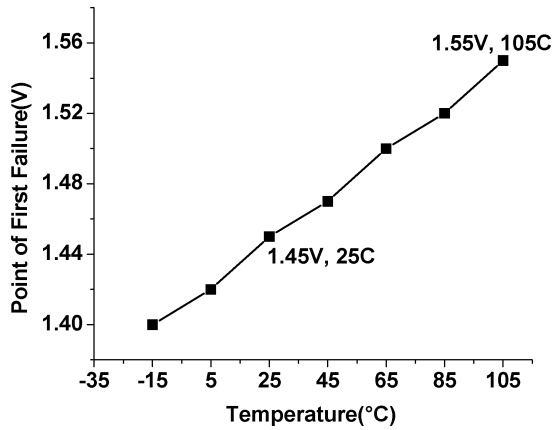


Fig. 11. Temperature margins.

first failure point which, being determined by the single longest critical path, shows higher process variation dependence.

Fig. 11 shows the effect of temperature on the point of first failure for a typical chip. Since critical path delay increases with temperature, the first failure voltage also increases and shifts by 100 mV from 1.45 to 1.55 V for a temperature change from 25 °C to 105 °C.

B. Total Energy Savings With Razor

The bar graph in Fig. 12 shows the energy for chips 1 and 2 when operating at 120 MHz. The first failure voltage for chips 1 and 2, as shown in Fig. 8, are 1.63 and 1.74 V, respectively, and therefore represent typical and worst case process conditions.

The first set of bars shows the energy when Razor is turned off and the chip under test is operated at the worst case operating voltage at 120 MHz, as determined for all the chips tested. This is the minimum voltage which guarantees error-free operation for the slowest process corner silicon at the worst case temperature of 105 °C and a power supply drop equal to 10% of the nominal voltage of 1.8 V. The point of first failure for the slowest chip, among the 33 tested dies, is 1.76 V at 25 °C which increases to 1.86 V at 105 °C, a change of 100 mV. To this, we add an extra 0.18 V (10% of 1.8 V) as safety margin for supply voltage drop, thus obtaining the worst case operating voltage of 2.04 V. Without Razor being enabled, all the chips would need to operate at the worst case voltage in order to ensure correct operation across all dies and operating conditions.

We measure the power consumption of chips 1 and 2 at this voltage and quantify how much of the worst case power is due to process, temperature, and voltage safety margins. We measure the power due to process margins of a chip by measuring the difference in power consumption when operating at its own point of first failure versus that when operating at the first failure

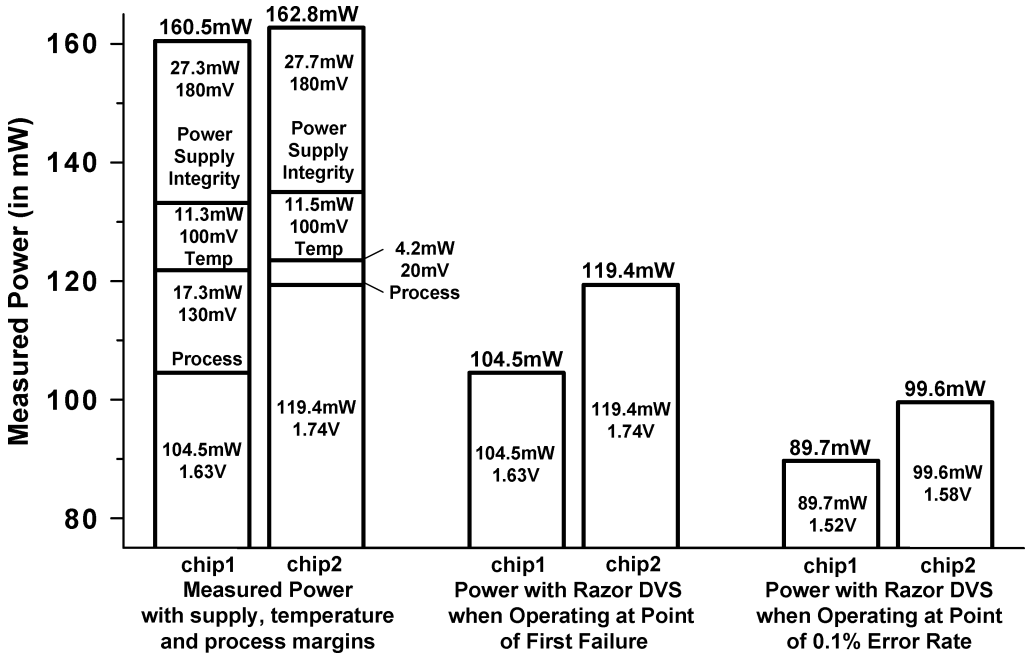


Fig. 12. Razor energy savings.

voltage of the worst case chip. For example, chip 1 consumes 17.3 mW extra when operating at 1.76 V (the point of first failure of worst case chip) as opposed to operating at its own first failure point of 1.63 V. The power due to temperature margins is measured by the difference in power consumption when operating at a voltage of 1.86 V (first failure point of worst case chip at 105 °C) versus operating at 1.76 V. Similarly, the power due to power supply margins is measured by operating the chip at the worst case voltage of 2.04 V versus operating it at 1.86 V. At 2.04 V, chip 1 consumes 160.5 mW of which 27.3 mW is due to safety margin for supply voltage drop, 11.2 mW is due to temperature margin, and 17.3 mW is due to process margin. Chip 2 consumes 162.8 mW at the worst case voltage, as shown in Fig. 12.

The second set of bars shows the energy when operating with Razor enabled at the point of first failure with all the safety margins eliminated. At the point of first failure, chip 1 consumes 104.5 mW while chip 2 consumes 119.4 mW of power. Thus, for chip 1, operating at the first failure point leads to a saving of 55.9 mW which translates to 35% saving over the worst case. The corresponding saving for chip 2 is 43.4 mW (27% saving over the worst case).

The third set of bars shows the additional energy savings due to subcritical mode of operation of Razor. With Razor enabled, both chips are operated at the 0.1% error rate voltage and power measurements are taken. Since the operating frequency is kept constant at 120 MHz and the IPC degradation is minimal at 0.1% error rate, the percentage savings in power is an accurate estimate of the percentage savings in energy. At the 0.1% error rate, chip 1 consumes 89.7 mW of power, which translates to 44% saving over the worst case (14% saving over operating at the point of first failure). Chip 2 consumes 99.6 mW of power at 0.1% error rate, which is a saving of 39% over the worst case (17% saving over the point of first failure). The total energy gains for chip 1 (71 mW, 44%) and chip 2 (63 mW, 39%) are comparable because the greater process margin in chip 1 (13 mW greater) is compensated by increased savings for chip 2 (4 mW extra) due to scaling below the first failure point.

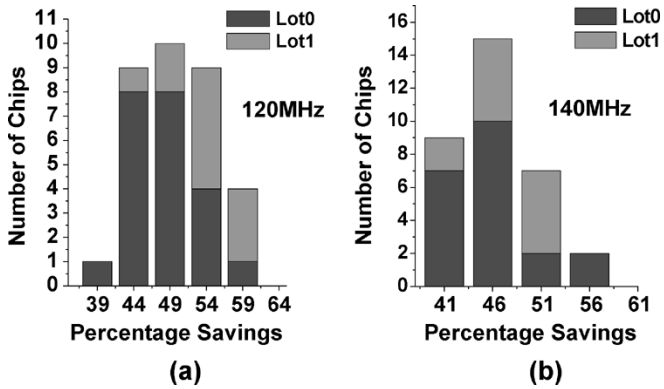


Fig. 13. Distribution of total energy savings over worst case for 33 measured chips.

The distribution of the percentage energy savings over the worst case for all 33 chips at 120 and 140 MHz operating frequencies is shown in Fig. 13. On average, we obtain approximately 50% savings over the worst case at 120 MHz and 45% savings at 140 MHz when operating at the 0.1% error rate voltage.

VI. Razor Voltage Control

Fig. 14 shows the basic structure of the hardware control loop that was implemented for real-time Razor voltage control. The controller reacts to the error rate that is monitored by sampling the error register and regulates the supply voltage to achieve a

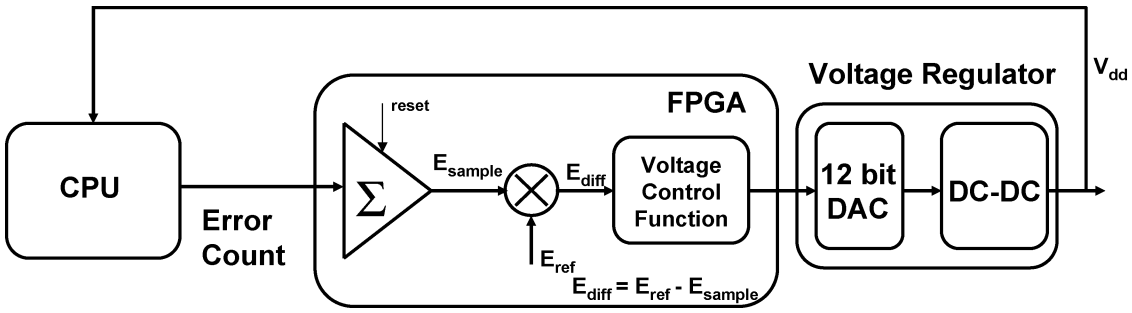


Fig. 14. Razor voltage control loop.

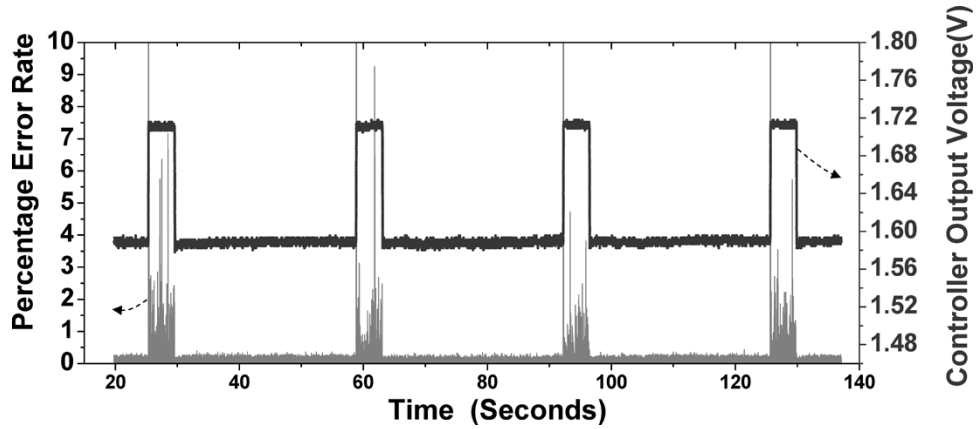


Fig. 15. Run-time response of the razor voltage controller

targeted error rate. The difference between the sampled error rate and the targeted error rate is the error rate differential, E_{diff}. A positive value of E_{diff} implies that the CPU is experiencing too few errors and hence the supply voltage may be reduced. If E_{diff} is negative, then the system is exhibiting too many errors and hence the supply voltage needs to be increased.

The control algorithm is implemented on a Xilinx XC2V250 FPGA, which computes the error rate from the sampled register. The pipeline $restore$ signal, when flagged, increments the error register. Thus, the error register is a measure of the total number of cycles where the Razor recovery mechanism is initiated. The controller on the FPGA reacts to the error-rate by adjusting the supply voltage to the chip through a DAC and DC–DC switching regulator. The DAC outputs an analog reference voltage to the regulator based on the 12-bit control output from the FPGA. The DC–DC regulator has a voltage gain of 1.76 and can source a maximum current of 600 mA. It can easily supply sufficient current to the chip which consumes less than 80 mA at 1.8 V. We tested the controller using a program which has alternating high and low error rate phases. At the high error rate phase, the processor is executing high latency instructions and hence the critical paths of the circuit are being exercised frequently. Therefore, a higher supply voltage is required to sustain the targeted error rate and *vice versa*.

The on-chip error counter is sampled at a frequency of 750 kHz and is accumulated within the field-programmable gate array (FPGA). The algorithm updates the control output at a conservative frequency of 1 kHz. If error rates are too high, voltage is increased at a rate of 1 bit per millisecond. Conversely, a low error rate caused a 1-bit decrease. This corresponds to a voltage change of 2.15 mV at the output of the DC–DC regulator feeding into the chip.

Fig. 15 shows a two-minute portion of the voltage controller response for the two-phase program execution. The targeted error rate for the given trace is set to 0.1% relative to CPU clock cycle count. The controller maintains an average of 0.1% error rate during the low error rate phase. In the high error rate phase, the controller maintains an average of 0.2% error rate although the median for the samples is still at 0.1% error rate. The control target is not achieved in the high error rate phase due to the occasional bursts in the error rate which increase the average error rate beyond that of the target. The error rate is bursty in this phase because a significantly greater number of critical paths are exercised and hence there is a greater sensitivity to noise in the supply voltage which causes the observed bursts. In the low error rate phase, a much smaller number of paths are critical and hence the sensitivity of the error rate to power supply noise is also reduced significantly.

The controller response during a transition from the low-error rate phase to the high-error rate phase is shown in Fig. 16(a). Error rates increase to about 15% at the onset of the high-error phase. The error rate falls until the controller reaches a high enough voltage to meet the desired error rate in each millisecond sample period. During a transition from the high error rate phase to the low error rate phase, shown in Fig. 16(b), the error rate drops to zero because the supply voltage is higher than required. The controller responds by gradually reducing the voltage until the target error rate is achieved. The average voltage maintained

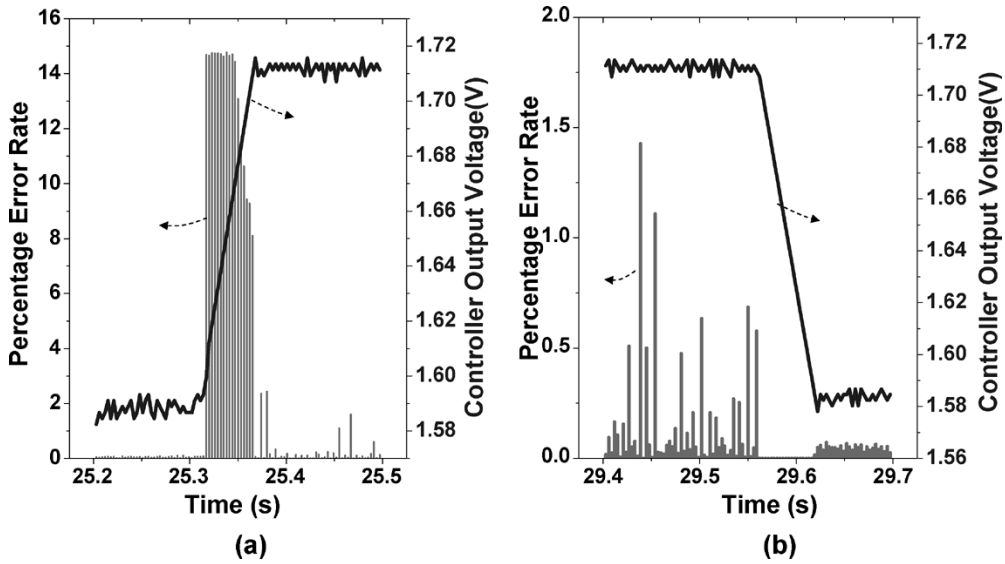


Fig. 16. Razor voltage controller: error-rate phase transition response.

during the low error rate phase is 1.59 V and the average voltage maintained at the high error rate phase is 1.72 V, a difference of 130 mV. More efficient and complex control and error prediction strategies are an area of ongoing research, including automatic optimal error-rate selection.

VII. Conclusion

In this paper, we presented a self-tuning processor with Razor-based DVS. Razor incorporates *in situ* error detection and correction mechanisms to eliminate voltage margins and to operate below the point of first failure. We presented the design of a novel delay-error tolerant flip-flop that detects and recovers from timing errors on the processor critical paths. With Razor-based voltage management, we obtained 50% energy savings over the worst case, on an average across 33 tested dies, by operating at the 0.1% error rate voltage at a constant frequency of 120 MHz. Since the energy-optimal voltage for Razor occurs at moderately low error rates, it motivates design optimization targeted at improving the delay of typically exercised logic paths as opposed to the worst case critical path. As process technology shrinks, Razor provides a solution toward achieving computational robustness and faster design closure in the presence of increasing silicon uncertainties.

Acknowledgment

The authors wish to thank D. Ernst, C. Ziesler, R. Rao, and T. Pham for their helpful suggestions and contributions.

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 *Proc. Int. Symp. Microarchitecture (MICRO-36)*, Dec. 2003, pp. 7–18.

[2] T. D. Burd, T. A. Pering, A. J. Stratakos, and R. W. Brodersen, "A dynamic voltage scaled microprocessor system," *IEEE J. Solid-State Circuits*, vol. 35, no. 11, pp. 1571–1580, Nov. 2000.

[3] M. Nakai, S. Akui, K. Seno, T. Meguro, T. Seki, T. Kondo, A. Hashiguchi, H. Kawahara, K. Kumano, and M. Shimura, "Dynamic voltage and frequency management for a low power embedded microprocessor," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 28–35, Jan. 2005.

[4] K. J. Nowka, G. D. Carpenter, E. W. MacDonald, H. C. Ngo, B. C. Brock, K. I. Ishii, T. Y. Nguyen, and J. L. Burns, "A 32-bit powerPC system-on-a-chip with support for dynamic voltage scaling and dynamic frequency scaling," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1441–1447, Nov. 2002.

[5] T. Kuroda, K. Suzuki, S. Mita, T. Fujita, F. Yamane, F. Sano, A. Chiba, Y. Watanabe, K. Matsuda, T. Maeda, T. Sakurai, and T. Furuyama, "Variable supply-voltage scheme for low-power high-speed CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 33, no. 3, pp. 454–462, Mar. 1998.

[6] V. von Kaenel, P. Macken, and M. Degrauwe, "A voltage reduction technique for battery-operated systems," *IEEE J. Solid-State Circuits*, vol. 25, no. 10, pp. 1136–1140, Oct. 1990.

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

[8] W. Dally and B. Poulton, *Digital System Engineering*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[9] R. Sproull, I. Sutherland, and C. Molnar, Counter-flow pipeline processor architecture Sun Microsystems Rep. SMLI-TR-94-25, Apr. 1994.

[10] MOSIS. [Online]. Available: http://www.mosis.org

[11] S. Das, S. Pant, D. Roberts, S. Lee, D. Blaauw, T. Austin, T. Mudge, and K. Flautner, "A self-tuning DVS processor using delay-error detection and correction," in *Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2005, pp. 258–261.

[12] B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "Theoretical and practical limits of dynamic voltage scaling," in *Proc. 41st Design Automation Conf.*, Jun. 2004, pp. 868–873.

[13] R. Gonzalez, B. Gordon, and M. Horowitz, "Supply and threshold voltage scaling for low power CMOS," *IEEE J. Solid-State Circuits*, vol. 32, no. 8, pp. 1210–1216, Aug. 1997.

[14] T. Mudge, "Power: a first-class architectural design constraint," *Computer*, vol. 34, no. 4, pp. 52–58, Apr. 2001.

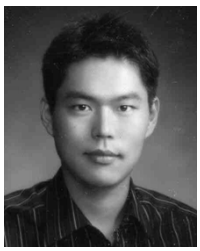
Shidhartha Das (S'03) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Bombay, India, in 2002, and the M.S. degree in computer science and engineering from the University of Michigan, Ann Arbor, in 2005, where he is currently pursuing the Ph.D degree.

His research interests include interconnect modeling and circuit-architectural co-design techniques for low-power digital IC design.

David Roberts (S'04) received the M.Eng. degree in computer systems engineering from the University of Warwick, Coventry, U.K. He is currently pursuing the Ph.D. degree at the University of Michigan, Ann Arbor.

His research interests include low-power and robust computer architectures.

Mr. Roberts is a member of the British Computer Society.



Seokwoo Lee received the B.S.E. degree (*summa cum laude*) in computer science from the University of Michigan, Ann Arbor, in 2003. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Engineering at the University of Michgan, Ann Arbor.

His research interests include computer architecture, variability-aware system design, reliable system design, and low-power system design and computer simulations.

Sanjay Pant received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, in 2001, and the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor, in 2004, where he is currently pursuing the Ph.D. degree.

In fall 2004 and summer 2005, he was with the Strategic CAD Laboratories, Intel Corporation, Hillsboro, OR, where he worked as a Graduate Intern. His research interests include low-power VLSI design and signal integrity issues in power distribution networks.

David Blaauw (M'94) received the B.S. degree in physics and computer science from Duke University, Durham, NC, in 1986, and the M.S. and Ph.D. degrees in computer science from the University of Illinois, Urbana, in 1988 and 1991, respectively.

He worked at IBM Corporation as a Development Staff Member until August 1993. From 1993 to August 2001, he worked for Motorola, Inc., Austin, TX, where he was the Manager of the High Performance Design Technology group. Since August 2001, he has been on the faculty of the University of Michigan as an Associate Professor. His work has focused on VLSI design and CAD with particular emphasis on circuit design and optimization for high-performance and low-power designs.

Dr. Blaauw was the Technical Program Chair and General Chair for the International Symposium on Low Power Electronic and Design in 1999 and 2000, respectively, and was the Technical Program Co-Chair and member of the Executive Committee of the ACM/IEEE Design Automation Conference in 2000 and 2001.

Todd Austin received the M.S. degree in computer engineering from the Rochester Institute of Technology, Rochester, NY, and the Ph.D. degree in computer science from the University of Wisconsin, Madison, in 1996.

He is an Associate Professor of electrical engineering and computer science at the University of Michigan. His research interests include computer architecture, compilers, computer system verification, and performance analysis tools and techniques.

Prof. Austin has earned numerous awards, including the Ruth and Joel Spira Outstanding Teacher Award in 2002 and a National Science Foundation CAREER Award in 2001. He is a member of Association for Computing Machinery (ACM).

Krisztián Flautner (M'03) received the B.S., M.S., and Ph.D. degrees in computer science and engineering from the University of Michigan, Ann Arbor.

He is Director of Advanced Research at ARM Limited, Cambridge, U.K., and the architect of ARM's Intelligent Energy Management technology. His research interests include high-performance, low-energy processing platforms that support advanced software environments.

Dr. Flautner is a member of the Association for Computing Machinery (ACM).

Trevor Mudge (S'74–M'77–SM'84–F'95) received the B.Sc. degree from the University of Reading, U.K., in 1969, and the M.S. and Ph.D. degrees in computer science from the University of Illinois, Urbana, in 1973 and 1977, respectively.

Since 1977, he has been on the faculty of the University of Michigan, Ann Arbor. He recently was named the first Bredt Family Professor of Electrical Engineering and Computer Science after concluding a ten-year term as the Director of the Advanced Computer Architecture Laboratory, a group of eight faculty and about 70 graduate students. He is the author of numerous papers on computer architecture, programming languages, VLSI design, and computer vision. He has also chaired about 33 theses in these areas. His research interests include computer architecture, computer-aided design, and compilers. In addition to his position as a faculty member, he runs Idiot Savants, a chip design consultancy.

Dr. Mudge is a member of the Association for Computing Machinery (ACM), the Institution of Electrical Engineers (IEE), and the British Computer Society.