RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance

Shidhartha Das, *Member, IEEE*, Carlos Tokunaga, *Student Member, IEEE*, Sanjay Pant, *Member, IEEE*, Wei-Hsiang Ma, *Student Member, IEEE*, Sudherssen Kalaiselvan, Kevin Lai, David M. Bull, and David T. Blaauw, *Member, IEEE*

***Abstract*—Traditional adaptive methods that compensate for PVT variations need safety margins and cannot respond to rapid environmental changes. In this paper, we present a design (RazorII) which implements a flip-flop with *in situ* detection and architectural correction of variation-induced delay errors. Error detection is based on flagging spurious transitions in the state-holding latch node. The RazorII flip-flop naturally detects logic and register SER. We implement a 64-bit processor in 0.13 μm technology which uses RazorII for SER tolerance and dynamic supply adaptation. RazorII based DVS allows elimination of safety margins and operation at the point of first failure of the processor. We tested and measured 32 different dies and obtained 33% energy savings over traditional DVS using RazorII for supply voltage control. We demonstrate SER tolerance on the RazorII processor through radiation experiments.**

***Index Terms*—Adaptive circuits, dynamic voltage and frequency scaling (DVFS), process variations, self-tuning processor, single event upsets.**

I. Introduction

RISING design uncertainties at advanced process nodes make it increasingly difficult to meet aggressive performance targets under strict power budgets. These uncertainties can be attributed to worsening inter- and intra-die process variations, temperature hotspots, supply voltage fluctuations, signal integrity concerns and aging effects such as Time Dependent Dielectric Breakdown [9] and Negative Bias Temperature Instability [8]. Traditionally, design uncertainties are addressed by operating the die at conservative voltage and frequency points such that sufficient safety margins exist. As process geometries shrink, the unacceptable performance and power impacts of pessimistic design margining has led to an increased interest in adaptive techniques. Adaptive techniques eliminate a significant portion of safety margins by dynamically adjusting system parameters such as supply voltage, body bias [15], [16] and operating frequency to account for variations in environmental conditions and silicon grade.

Manuscript received April 08, 2008; revised July 31, 2008. Current version published December 24, 2008. This work was supported in part by ARM Ltd.

S. Das and D. M. Bull are with ARM Ltd., Cambridge CB1 9NJ, U.K. (e-mail: shidhartha.das@arm.com).

C. Tokunaga, D. T. Blaauw, and W. S. Ma are with the Advanced Computer Architecture Laboratory, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA.

K. Lai is with Intel, Hillsboro, OR 97124 USA.

S. Pant is with Advanced Micro Devices, Fort Collins, CO 80528 USA.

S. Kalaiselvam is with Advanced Micro Devices, Sunnyvale, CA 94085 USA.

Digital Object Identifier 10.1109/JSSC.2008.2007145

The traditional methods of adaptive design have used look-up tables [6], [7] or so-called "canary" circuits [1]–[4]. In the look-up table based approach, the design is pre-characterized to obtain voltage and frequency pairs for which correct operation is guaranteed. This approach exploits periods of low CPU utilization by dynamically scaling voltage and frequency, thereby obtaining energy savings. However, each operating point must be suitably margined to guarantee computational correctness in the worst-case combination of process, voltage and temperature (PVT) conditions.

The canary-circuit based approach eliminates a subset of these worst-case margins by using a delay-chain which mimics the critical path of the actual design. The propagation delay through this replica-path is monitored and voltage and frequency are scaled until the replica-path just about fails to meet timing. The replica-path tracks the critical-path delay across inter-die process variations and global fluctuations in supply voltage and temperature, thereby eliminating margins due to global PVT variations. However, the replica-path does not share the same ambient environment as the critical-path because its on-die location differs. Consequently, margins are added to the replica-path in order to budget for delay mismatches due to on-chip variation and local fluctuations in temperature and supply voltage. Margins are also required to address fast-changing transient effects such as coupling noise which are difficult to respond to in time, with this approach. Furthermore, mismatches in the scaling characteristics of the critical-path and its replica require additional safety margins. These margins ensure that the processor still operates correctly at the point of failure of the replica-path.

To eliminate worst-case safety margins, we proposed a novel voltage management technique, for Dynamic Voltage Scaled (DVS) processors, based on *in situ* error detection and correction, called Razor [13]. In this technique, we use a delay-error tolerant flip-flop on critical paths to scale the supply voltage to the point of first failure (PoFF) of a die for a given frequency. Thus, all margins due to global and local PVT variations are eliminated, resulting in significant energy savings. In addition, the supply voltage can be scaled even lower than the first failure point into the sub-critical region, deliberately tolerating a targeted error rate, thereby providing additional energy savings. Thus, in the context of Razor, a timing error is not a catastrophic system failure but a trade-off between the overhead of error-correction and the additional energy savings due to sub-critical operation. We use this distinction throughout the remainder of the paper wherein "error" refers to a timing violation recoverable

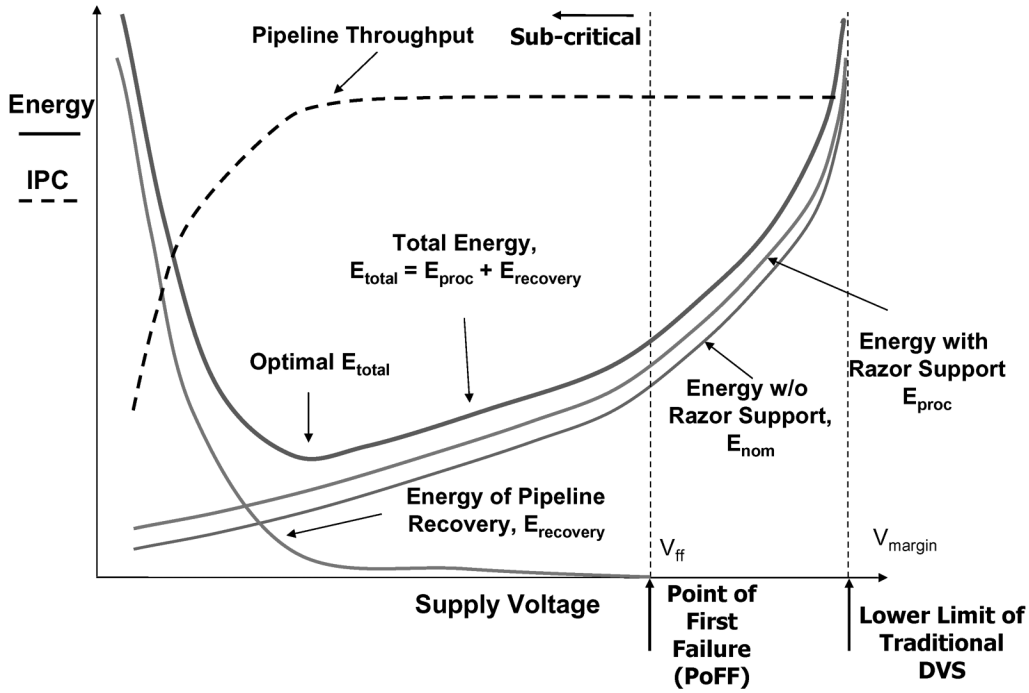


Fig. 1. Qualitative relationship between supply voltage, energy and IPC.

through Razor error correction and a "system failure" refers to unrecoverable pipeline corruption.

The key trade-off between the overhead of error-correction and sub-critical operation in Razor is qualitatively illustrated in Fig. 1 [1]. The voltage at the PoFF of the processor $(\mathrm{V_{PoFF}})$ and the minimum allowable voltage of traditional DVS techniques $(\mathrm{V_{margin}})$ are also labeled in the figure. $\mathrm{V_{margin}}$ is much higher than $\mathrm{V_{PoFF}}$ under typical conditions, since safety margins need to be included to budget for worst-case operating conditions. Razor relies on *in situ* error detection and correction capability to operate at $\mathrm{V_{PoFF}}$, rather than at $\mathrm{V_{margin}}$. The total energy of the processor $(\mathrm{E_{tot}})$ is the sum of the energy required to perform standard processor operations $(\mathrm{E_{proc}})$ and the energy consumed in recovery from timing errors $(\mathrm{E_{recovery}})$. Implementing Razor incurs power overhead due to which the nominal processor energy $(\mathrm{E_{nom}})$ *without* Razor technology is slightly less than $\mathrm{E_{proc}}$. As the supply voltage is scaled, the processor energy $(\mathrm{E_{proc}})$ reduces quadratically with voltage. However, as voltage is scaled below the PoFF $(\mathrm{V_{PoFF}})$, the error rate and the recovery energy $(\mathrm{E_{recovery}})$ 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 $(\mathrm{E_{tot}})$ shows an optimal point where the rate of change of $\mathrm{E_{recovery}}$ and $\mathrm{E_{proc}}$ offset each other.

From our initial Razor experiments [5] (henceforth referred to as RazorI) implemented on a 64 bit processor with 0.18 micron technology, we obtained an average energy savings of 50% over the worst-case by operating at the optimal voltage point, at a fixed frequency of 120 MHz. We used delay-error tolerant RazorI flip-flops only on the critical paths of the processor. The total power overhead due to RazorI flip-flops was found to be 3% of the total chip power. A key finding from these measurements is that the error rate at the PoFF is extremely low, $\sim$1 error in 10 million cycles, making the recovery energy negligible at this operating point. However, it was also found that beyond the PoFF, the error rate increases exponentially at one decade per 10 mV supply voltage increase. Hence the energy gain from operating substantially below the PoFF was small ($\sim$10%) compared to the energy gain from eliminating the PVT margins ($\sim$35 to 45%) [5].

In this paper, we take advantage of these findings and propose a new technique called RazorII wherein the processor is intended to operate near the PoFF and recovery from a timing error occurs by a conventional architectural replay mechanism. Replaying an erroneous instruction incurs greater Instructions Per Cycle (IPC) overhead than the counter-flow pipeline recovery technique used in [5]. However, as error-rates are extremely low at the PoFF, the increased IPC overhead from using architectural replay has a negligible impact on the overall energy efficiency. Architectural replay greatly simplifies the error recovery path, thereby making RazorII significantly more amenable to high-performance microprocessors compared to RazorI. We introduce a novel timing-error detecting flip-flop (RazorII flip-flop) based on flagging spurious transitions at the state-holding node. As we show in Section III, the design of the RazorII flip-flop naturally allows it to detect Single Event Upsets (SEU) within the flip-flop and in the combinational logic. We present the design and measurement results from a 64 bit processor that uses RazorII for SEU tolerance and low-energy operation through dynamic supply adaptation. In [12], the authors propose a similar approach for high-performance computing.

The remainder of the paper is organized as follows. Section II provides a small overview of RazorI and discusses the motivation for RazorII. Section III explains the key concepts and the transistor-level design of the RazorII flip-flop. The micro-archi-

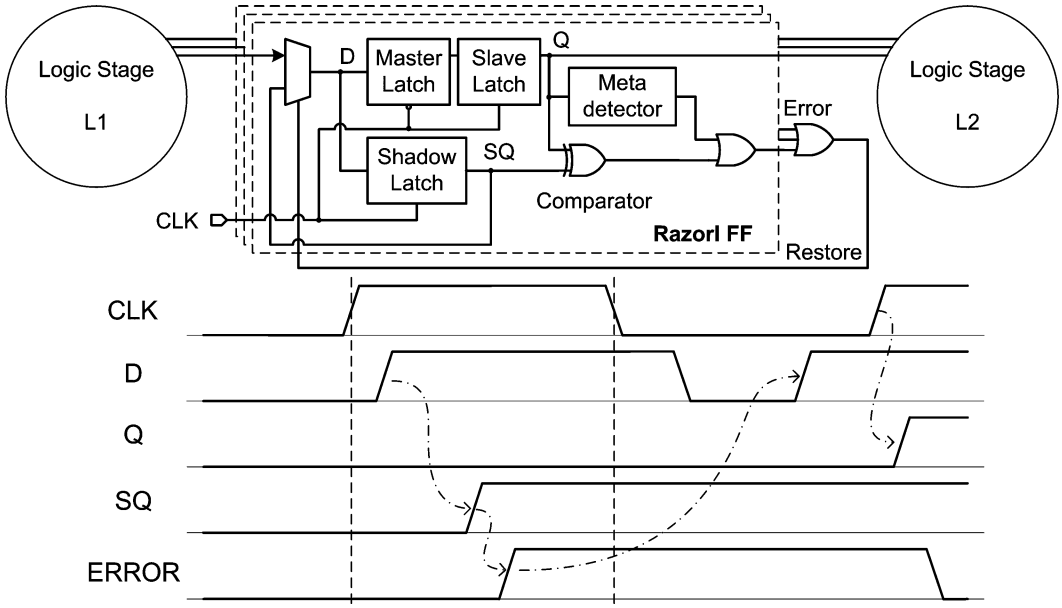


Fig. 2. RazorI flip-flop and conceptual timing diagrams.

tectural features of the RazorII processor and the pipeline replay mechanism are described in Section IV. We discuss methods for setting safe operating limits for a Razor processor in Section V and present our measurement results in Section VI. Finally, we conclude with remarks on the direction of future research on Razor in Section VII.

II. RazorI Overview

Fig. 2 conceptually describes the architecture of the delay-error tolerant RazorI flip-flop for error-detection in critical paths. The key concept in this scheme is to sample the input data of the flip-flop at two different points in time. The earlier, speculative sample is stored in a conventional positive-edge triggered, master-slave flip-flop. This main flip-flop is augmented with a so-called "shadow latch" which samples at the negative edge of the clock. Thus, the shadow-latch gets additional time equal to the high-phase of the clock to capture the correct state of the data. An error is flagged when data captured at the main flip-flop differs from the shadow-latch data. As the setup and hold constraints for the main flip-flop are allowed to be violated, an additional detector is required to flag the occurrences of metastability at the output of the main flip-flop. The error-pins of individual RazorI flip-flops are then "OR"-ed together to generate a pipeline *restore* signal which overwrites the correct data in the shadow-latch into the main flip-flop, at the next positive edge of the clock.[1] Since the shadow latch data is used to overwrite state in the main flip-flop, it is required to ensure using conventional worst-case techniques that the data in the shadow latch is always correct.

There are key design issues that complicate the deployment of RazorI in high-performance, aggressively-clocked microprocessors. The primary difficulty is the generation and propagation of the pipeline *restore* signal. The *restore* signal is evaluated at the output of a high fan-in OR-tree and is suitably buffered and routed to every flip-flop in the pipeline stage before the next rising edge of the clock. This imposes significant timing constraints on the *restore* signal and the error recovery path can itself become critical when the supply voltage is scaled. This limits the voltage headroom available for Razor, especially in aggressively clocked designs. The design of the metastability detector is also difficult under rising process variations as it is required to respond to metastable flip-flop outputs across all process, voltage and temperature corners. Consequently, it requires the use of larger devices which adversely impacts the area and power overhead of the RazorI flip-flop. There is the additional risk of metastability at the *restore* signal which can propagate to the pipeline control logic, potentially leading to system failure.

[1]For simplicity, the conceptual diagram in Fig. 2, uses a mux at the data input for this purpose, however a more efficient implementation is given in [5].

III. Key Concepts of RazorII

In order to effectively address the design and timing issues in RazorI, we propose an improved alternative, called RazorII, which moves the responsibility of recovery entirely to the micro-architectural domain. The RazorII approach introduces two novel components which are as follows:

1) Instead of performing both error detection and correction in the flip-flop, RazorII performs only detection in the flip-flop, while correction is performed through architectural replay. This allows significant reduction in the complexity and size of the Razor flip-flop, although at the cost of increased IPC penalty during recovery. Architectural replay is a conventional technique which often already exists in high-performance microprocessors to support speculative operation such as out-of-order execution and branch prediction. Hence, it is possible to overload the existing framework to support replay in the event of timing errors. In addition, this technique precludes the need for a pipeline *restore* signal, thereby significantly relaxing the timing constraints on the error-recovery path. This feature makes Ra-

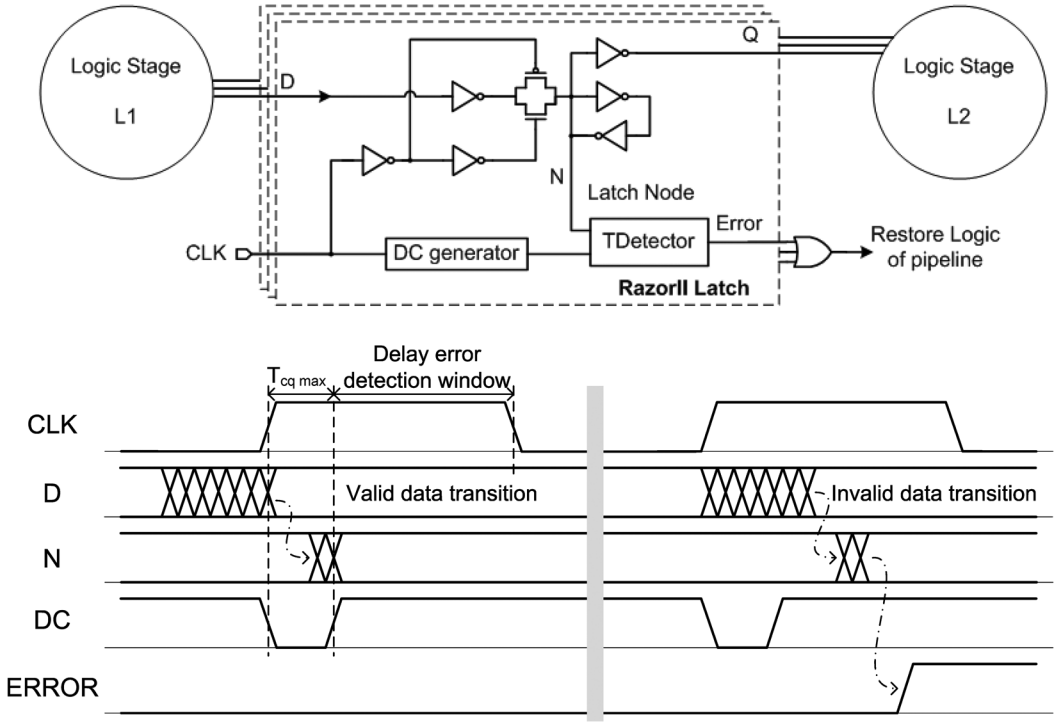


Fig. 3. RazorII flip-flop and conceptual timing diagrams.

zorII highly amenable to deployment in high-performance processors.

2) The design of the RazorII flip-flop uses a positive level-sensitive latch instead of a master-slave flip-flop. The flip-flop operation is enforced by flagging any transition on the input data in the positive clock-phase as a timing error. Elimination of the master latch significantly reduces the clock-pin capacitance of the flip-flop bringing down its power and area overhead. In addition, it also allows the RazorII flip-flop to naturally detect Single Event Upsets (SEU) in the logic and registers without additional overhead. In the following sub-sections, we discuss in detail how timing error detection and SEU tolerance are simultaneously achieved using the RazorII flip-flop.

A. Transistor Level Design of the RazorII Flip-Flop

The architecture and the principle of operation of the RazorII flip-flop are illustrated in Fig. 3. It uses a single positive level-sensitive latch, augmented with a transition-detector controlled by a detection clock (DC). Timing errors are detected by monitoring the internal latch node for spurious transitions. A legitimate transition occurs when data is setup to the latch input before the rising edge of the clock. In this case, the output Q of the latch transitions at the rising edge after a delay equal to the clock-to-Q (CLK-Q) delay of the latch, to reflect the state of data being captured. In order to prevent legitimate transitions being flagged as timing errors, a short negative pulse on the detection clock is used to disable the transition detector for at least the duration of the CLK-Q delay after the rising edge, as shown in the figure. However, if the input data transitions after the rising clock edge, during transparency, the transition of latch node, N, occurs when the transition detector is enabled and results in assertion of the error signal. The error signal engages the architectural replay mechanism to restore correct state within the pipeline.

The circuit schematic of the RazorII flip-flop, the detection clock generator and the transition-detector are shown in Fig. 4. The transition-detector (Fig. 4(b)), uses a delay-chain to generate an"implicit" pulse out of a rising or a falling transition at the latch node, N. The pulse is then captured by a dynamic OR gate to generate the error signal. Two pulse-generators are required to capture transitions in both directions. The AND gates required for the pulse generation are built as a part of the evaluation tree of the OR-gate and have as inputs, the monitored node and its delayed version. For example, the pulse-generator for the rising transition at node N, uses the inverter, I3, and the long-channel transmission gate, TG2, to create the required delay. The inputs to the corresponding AND gate are the nodes d1 and the d3, as labeled in the figure. Similarly, the pulse-generator for the falling transition uses gates I2 and TG1 and the corresponding inputs to the AND gate are d0 and d2. For silicon test-and-debug purposes, the delay-chain for each pulse-generator can be controlled by tuning the gate voltage of long-channel transmission gates (TG1 and TG2) in the delay-chains through the TD-TG Vdd pin. However, it was found that the test chip was fully functional without the need for tuning.

The error-reset signal pre-charges the dynamic node in the OR-gate enabling it to capture subsequent transitions on the latch node. Error-reset is generated during architectural recovery in the event of a timing error. Using the error-reset signal instead of the clock for precharge, reduces the total clock-pin capacitance. Thus, the dynamic node is conditionally

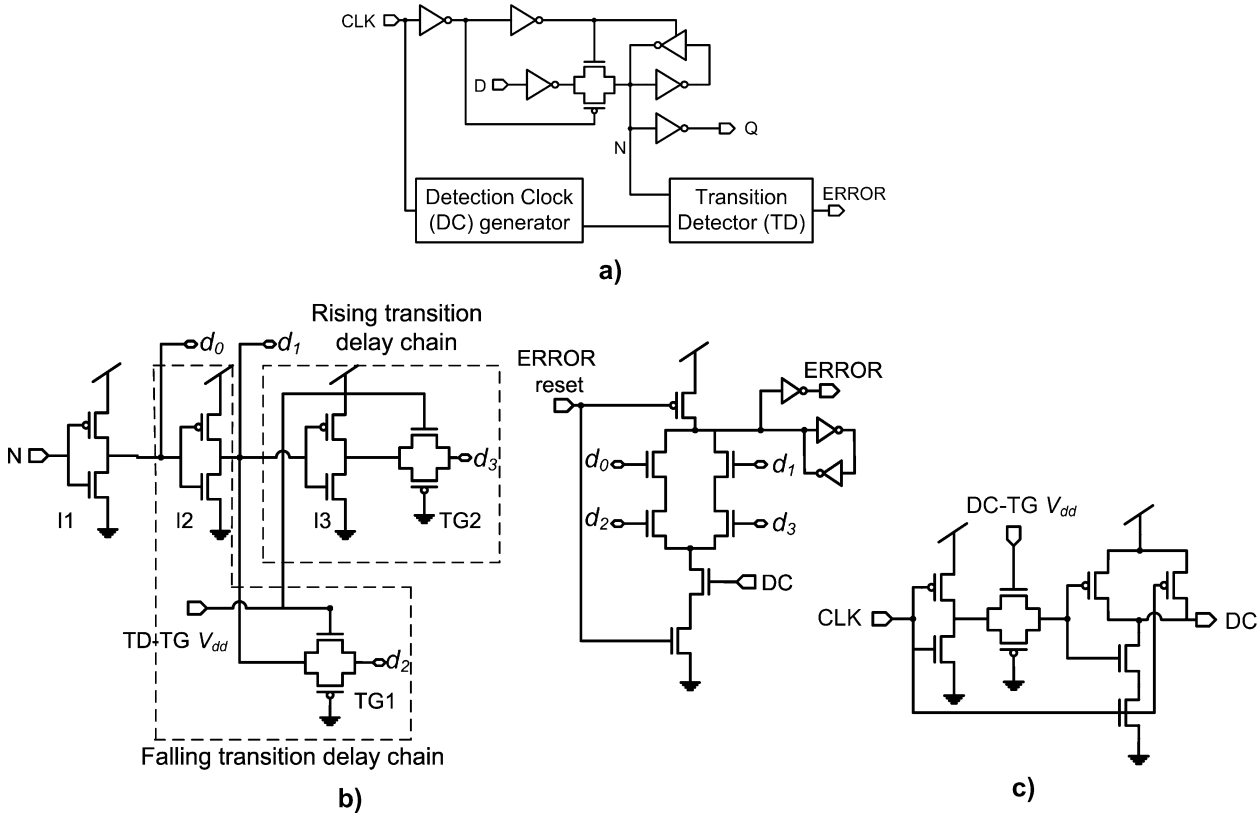


Fig. 4. Circuit-level schematic of the RazorII flip-flop. (a) RazorII flip-flop circuit schematic; (b) transition-detector; (c) detection clock generator.

precharged during recovery, in the event of a timing error. A cross-coupled inverter pair is used as a latch structure to protect the dynamic node from discharge due to leakage.

B. Impact of Intra-Die Process Variability

As explained previously, the low-pulse temporarily disables the transition-detector thereby preventing legitimate transitions at the latch node from being flagged as errors. For correct functionality, it is required that the *minimum* width of the low pulse at the DC clock is greater than the *maximum* CLK-Q delay of the main latch across all PVT corners. The width of the DC pulse is determined by the delay through the delay-chain in the DC generator. We used conventional worst-case sizing of the transistors in the DC generator to satisfy this constraint on silicon. Achieving this in the face of rising intra-die process variability at 45 nanometer technology node and below, may require the use of Monte-Carlo sampling techniques. For a 3-sigma yield target, it is required to ensure that the 3-sigma increase of the CLK-Q delay of the latch is still covered by the 3-sigma reduction in the DC pulse-width. The relevant timing diagram with process variation is illustrated in Fig. 5.

In order to enable post-manufacture tuning and to account for process-variation mismatches between the latch delay and DC pulse-width, the delay-chain in the DC generator is made tunable by controlling the gate voltage of the transmission gate through the DC-TG Vdd pin. Again, tuning was not required for the normal operation of the chip. The DC-TG Vdd pin of individual RazorII flip-flops were routed as conventional signal nets with an input pad serving as a common driver. The TD-TG

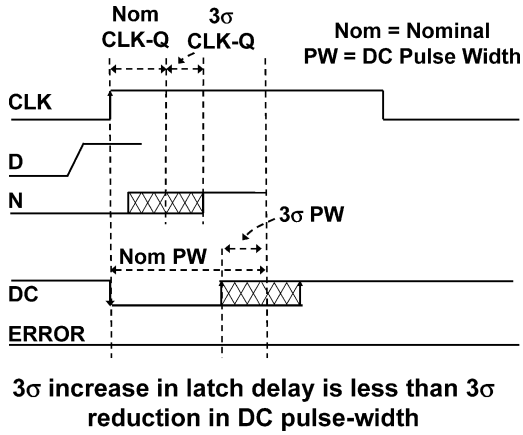


Fig. 5. Timing constraints with intra-die process variations.

Vdd pin was also routed in a similar manner. These pins have relaxed timing constraints since they are only meant for post-manufacture tuning. The analog tuning voltages (DC-TG Vdd and TD-TG Vdd) are generated using external regulators which form a part of the test-gig. During testing, they were set at their default setting of 1.2 V (nominal supply voltage for the technology used).

The difference between the CLK-Q delay and the DC pulse-width represents the duration when a transition on N goes undetected. This allows dynamic time-borrowing in the RazorII flip-flop wherein a critical computation gets extra time from the next cycle to complete, without flagging a timing error. Of course, this reduces the available time for the succeeding

pipestage. However, if a time-borrowing critical computation is followed by a non-critical computation in the succeeding pipeline stage, then no timing errors will be flagged. Thus, the pipeline can potentially operate at a frequency greater than what is dictated by the critical path of the circuit. A larger value of the DC pulse-width allows greater scope for dynamic time-borrowing, although at the expense of reducing the available time for error detection. Thus, the DC pulse-width represents the trade-off between dynamic time-borrowing versus timing speculation available on the critical path of the circuit.

The duration of suppression of the transition-detector, denoted by the DC pulse-width needs to be greater than the CLK-Q delay of the actual latch across all PVT corners. In the prototype processor, we ensured this through conservative sizing during design time. At the slow corner, the DC pulse width is 350 ps and the actual CLK-Q delay of the latch is 200 ps leading to a margin f 150 ps at the slow corner. As explained previously, a higher value of the DC pulse-width reduces the Point of First Failure through dynamic time-borrowing at the expense of reducing the speculation time available. Thus, reduction in energy savings due to reduced speculation is counter-balanced by the energy gain due to a lower Point of First Failure. Hence, it is unlikely that the margin in the DC pulse width has a significant impact on the total energy savings of the processor.

C. Fundamental Minimum-Delay Trade-Off

The RazorII error detection window lies between the rising edge of DC and the falling edge of CLK and can be controlled with the duty cycle of CLK. This constrains the minimum propagation delay for a combinational logic path terminating in a RazorII flip-flop to be at least greater than the duration of the high clock phase. Delay buffers are required to be inserted in those paths which fail to meet this minimum path delay constraint. The insertion of delay buffers incurs power overhead because of the extra capacitance added. A longer clock high-phase requires a greater number of delay buffers to be inserted, thereby increasing the power overhead. However, a smaller high-phase implies that the voltage difference between the PoFF and the point where error-detection fails is less and, thus, reduces Razor timing speculation.

The duration of the positive phase of the propagated clock can be configured as required so as to exploit the above trade-off. A key observation is that the hold constraint only limits the *maximum duration of the positive clock phase* and does not affect the clock frequency. Thus, the pipeline can still be operated at any frequency as required as long as the positive clock phase is sufficient to meet the minimum delay constraint. In the processor that we present in this paper, timing critical flip-flops had a clock with a 40% duty cycle resulting in a 25 F04 detection window while non-critical flip-flops had a 13% clock duty cycle to minimize buffer insertion. A total of 1924 buffers were added to meet hold time constraints which added a 1.3% power overhead.

Since the supply voltage is never lowered to the point where the latch transitions at the falling clock edge, meta-stability of the latch node is avoided. However, a transition at the falling edge of DC can cause partial discharge of the dynamic OR gate in the TD, leading to mestability at the error output. A key observation is that when the error signal becomes metastable the actual latch node is still correct. Hence, such an event is benign from the perspective of the data path. To avoid the metastable error signal from propagating through the error recovery logic, it is double-latched before being forwarded to the architectural replay unit. Depending on how the metastable signal resolves, the replay unit can potentially interpret the event as a valid error. In this case, the pipeline is flushed and a replay event occurs. Thus, this case is a "false positive" wherein replay occurs even though the pipeline data is still correct.

D. SEU Detection Using the RazorII Flip-Flop

The transition detector is always enabled except for the period after the rising edge of the clock where valid transitions occur. This naturally allows the RazorII flip-flop to detect and flag Single Event Upsets (SEU) on the latch node as well as in the combinational logic that fans in to it. The voltage pulse due to a SEU event in the combinational logic can possibly propagate to a RazorII flip-flop in the transparent phase of the clock, leading to a glitch in the latch node. If the glitch is sufficiently wide, the transition detector interprets the glitch as a combination of two transitions and flags it as a timing error. Of course, in the low-phase of the clock, the SEU pulse is benign, as it can never propagate to the latch node. A particle strike on the latch node, N, leads to a single transition causing a state flip to occur when the clock is low, as shown in Fig. 6(a). A high energy particle strike at N (Fig. 6(b)) leads to a pulse when the strike occurs in the transparent phase of the clock. In both the cases of Fig. 6(a) and (b), the TD successfully detects the event and flags an error. A weak pulse due to SEU in the transparent clock phase, shown in Fig. 6(c), can go undetected by the TD but it can possibly cause a glitch in the output, Q. If the glitch is amplified by the downstream logic, it will still be flagged by a RazorII flip-flop in the succeeding stage, leading to an error.

An interesting case to consider is when the SEU pulse occurs while the detection clock is low. There can be three possible scenarios that follow as illustrated in the timing diagrams in Fig. 7:

Case I: The pulse occurs and dies before the detection clock is enabled as shown in Fig. 7(a). In such a case, the transition detector does not flag an error since the latch node, N, is restored to its correct state before the detection clock is enabled. Since no state corruption occurs, this is essentially benign.

Case II: The pulse on N initiates when the detection clock is low but N recovers correct state after the rising edge of the detection clock (Fig. 7(b)). In such a case, the transition detector responds to the trailing edge of the pulse and flags an error. While the pulse does revert back to its correct state, it does so after the DC has reengaged and hence, it is correctly interpreted as a timing error.

Case III: Fig. 7(c) illustrates the case when a SEU pulse actually causes state corruption to occur at the latch node, N, without an error being flagged. This is a special condition which occurs when the width of the DC pulse is equal to the width of high phase of clock. As shown in the figure, if the SEU pulse occurs just before the falling edge of the

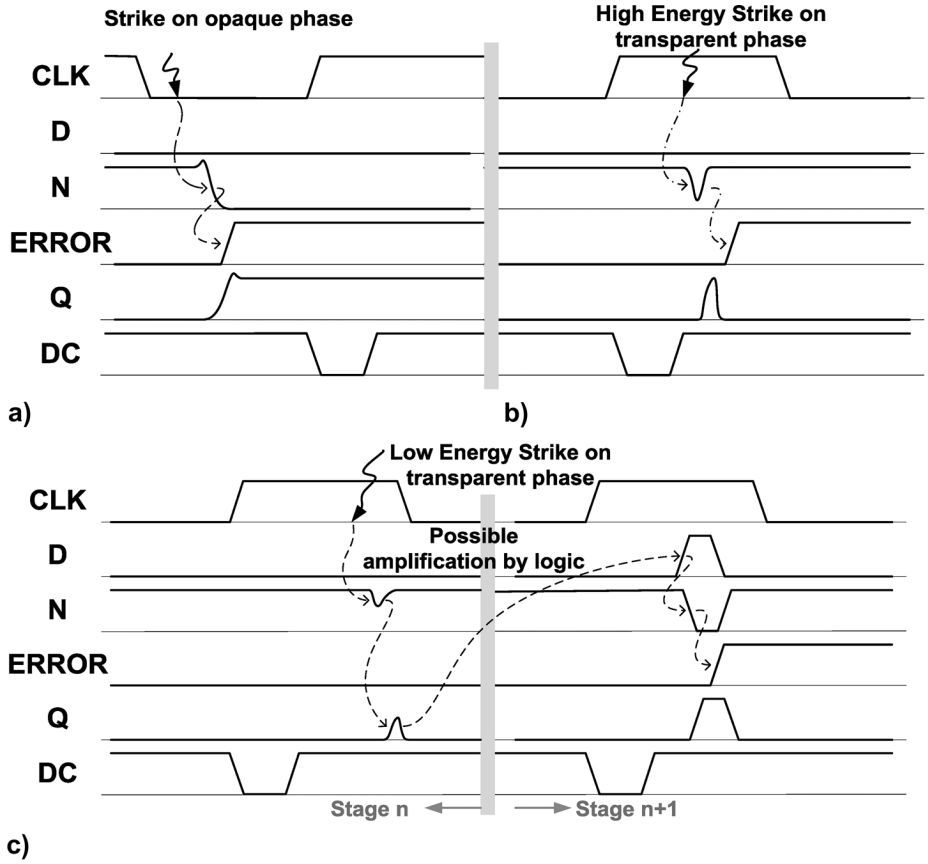


Fig. 6. Conceptual timing diagrams showing SEU detection when DC is high. (a) Strike on the opaque phase of the clock; (b) high-energy strike on the transparent phase of the clock; (c) low-energy strike on the transparent phase of the clock with amplification by logic.

clock, then the latch node, N, samples the leading edge of the pulse and is unable to revert back to its correct state. This is because the main latch enters into its opaque phase just before the trailing edge of the pulse occurs. In essence, the SEU pulse degenerates into a single transition at the latch node, N. Since this transition occurs when the detection clock is low, the transition detector does not flag this as a timing error, leading to system failure.

Case III sets a lower bound on duration of the high phase of the clock in relation to the pulse-width of the detection clock. In order to prevent system failure from occurring in case III, it is imperative to allow node N to recover correct state before the falling edge of the clock. Hence, the clock needs to be high for at least half of the SEU pulse width *after* the rising edge of the detection clock. This allows N to follow the trailing edge of the pulse and achieve correct state as shown in Fig. 7(d). For the processor that we present in this paper, we chose this minimum overlap to be 100 ps at the typical corner. The trailing edge of the pulse occurs after the transition-detector has been enabled and hence is flagged as an error, thus maintaining correct operation in the pipeline.

E. Comparative Analysis With the RazorI Flip-Flop

The RazorI flip-flop detects late-arriving data by comparing the flip-flop state with that of a "shadow" latch which samples off the negative edge of the clock. In total it consists of three latches (master, slave and shadow), a comparator and a meta-stability detector. The implementation requires 76 transistors. Compared to a conventional flip-flop, the RazorI flip-flop has a worse CLK-Q delay for the same drive strength due to the extra loading of the metastability detector and the error comparator. Its setup time is the same or slightly worse than a library flip-flop. It consumes 25% extra power when data does not switch (due to transitions on clock) and 70% extra power when data switches. Thus, for a 10% activity rate, the total power overhead of the RazorI flip-flop is 30% when compared to a conventional flip-flop. For our implementation of the RazorI processor in [5], we needed to use the RazorI flip-flop only for timing-error protection on the critical paths. Hence, the net power overhead of using RazorI flip-flops was less than 3% of the total chip power.

The elimination of the master latch and the metastability detector in the RazorII flip-flop are significant simplifications that lead to improvements in delay, power and area. The RazorII flip-flop uses 47 transistors in total. The elimination of the master latch leads to slightly improved CLK-Q delay compared to a conventional flip-flop. In addition, it completely eliminates the setup time constraint at the positive edge of the clock. Thus, the RazorII flip-flop can be effectively modeled as having 0 ps setup time. The power overhead compared to a conventional flip-flop of the same drive strength for a 10% activity factor is 28.5%. The total power overhead due to insertion of RazorII flip-flops in the processor was 1.2%. Metastability risks at the positive edge of the clock for the latch data-path are completely

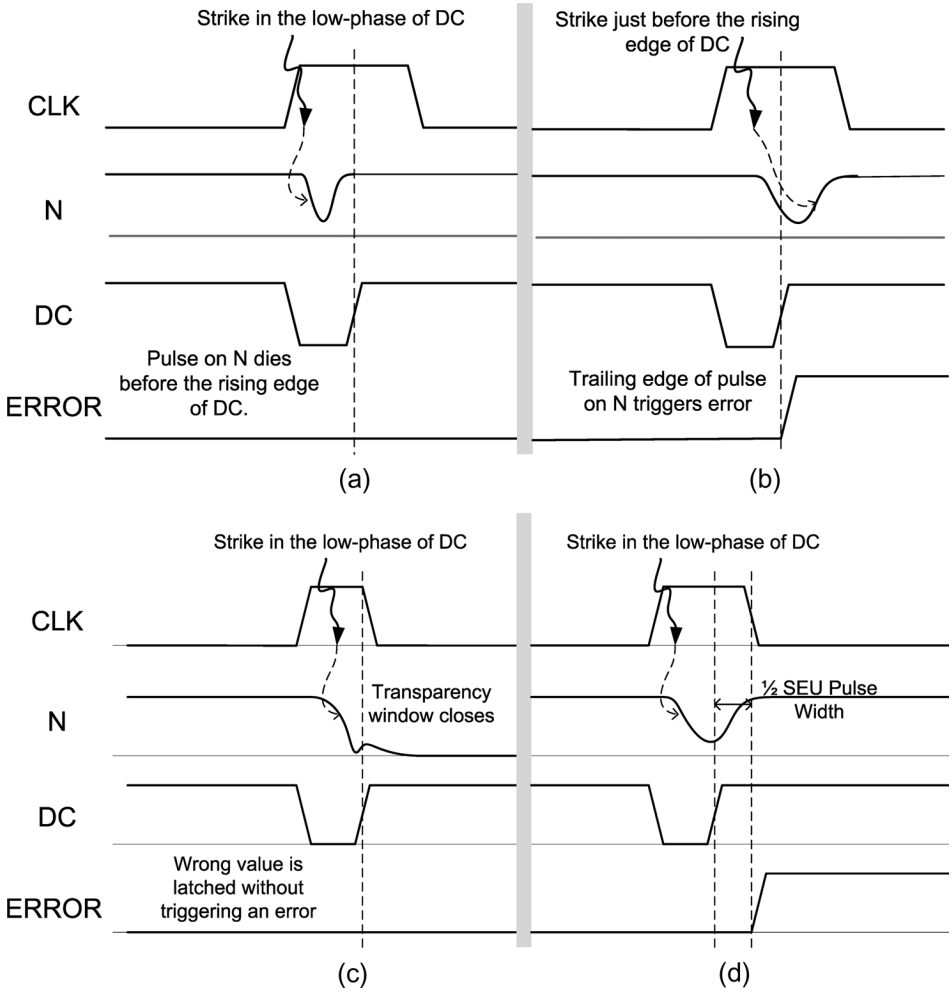


Fig. 7. SEU detection when DC is low.

eliminated, thereby precluding the need for a metastability detector.

As mentioned before, the generation of the detection clock could be easily shared across multiple RazorII flip-flops. Such an implementation requires 39 transistors in total as opposed to 76 transistors used for the RazorI flip-flop. In addition, it requires just a single additional transistor on clock for error evaluation. In total, it has 5 transistors on a clock as compared to 14 clock transistors in the RazorI flip-flop. Thus when compared to a conventional flip-flop, for an activity factor of 10%, it consumes 11% less power based on a simulation analysis. This is a significant reduction compared to the RazorI flip-flop which consumes 30% extra power. In the actual implementation of the processor, the detection clock was generated locally within each RazorII flip-flop and was not shared.

IV. Pipeline Design of the RazorII Processor

RazorII was incorporated in a 64 bit, 7 stage, in-order Alpha processor in a 0.13 μm technology for timing error detection and SEU tolerance. The architecture, shown in Fig. 8, can be broadly divided into a speculative domain which is timing-critical and a non-speculative domain with sufficient timing slack. The speculative domain consists of a two-stage fetch stage (IF1 and IF2), instruction decode (ID), an integer execution unit (EX) and the memory access (MEM) stage. All pipeline registers in the speculative domain require RazorII protection against state corruption due to SEU. The error pins of all the RazorII flip-flops in each pipeline stage are ORed together and the result is propagated and ORed with that of the next stage. This allows the composite error signal for the entire pipeline to be evaluated on a per-stage basis. This relaxes the timing constraint on the error generation path. The write-back (WB) stage was designed to be non-critical to stabilize the speculative pipeline output before it was committed to storage in the non-speculative domain.

The non-speculative domain stores the architectural state of the processor and consists of the caches (instruction and data), the Register File and the program status registers. Read and write combinational paths to these units are non-critical and hence do not require timing error protection. Error Correcting Codes (ECC) is used to recover from SEU in the caches and the Register File. The program status registers are protected using Triple Module Redundancy [10] (TMR). The key concept of TMR is to use three blocks of logic for the same computation. A majority voting circuit is then used to forward the final result to the pipeline. A TMR error is flagged when the outputs of the redundant logic blocks mismatch. Thus, high degree of reliability against SEU can be achieved although at the cost of redundant logic and registers. TMR allows SEU errors in the architectural

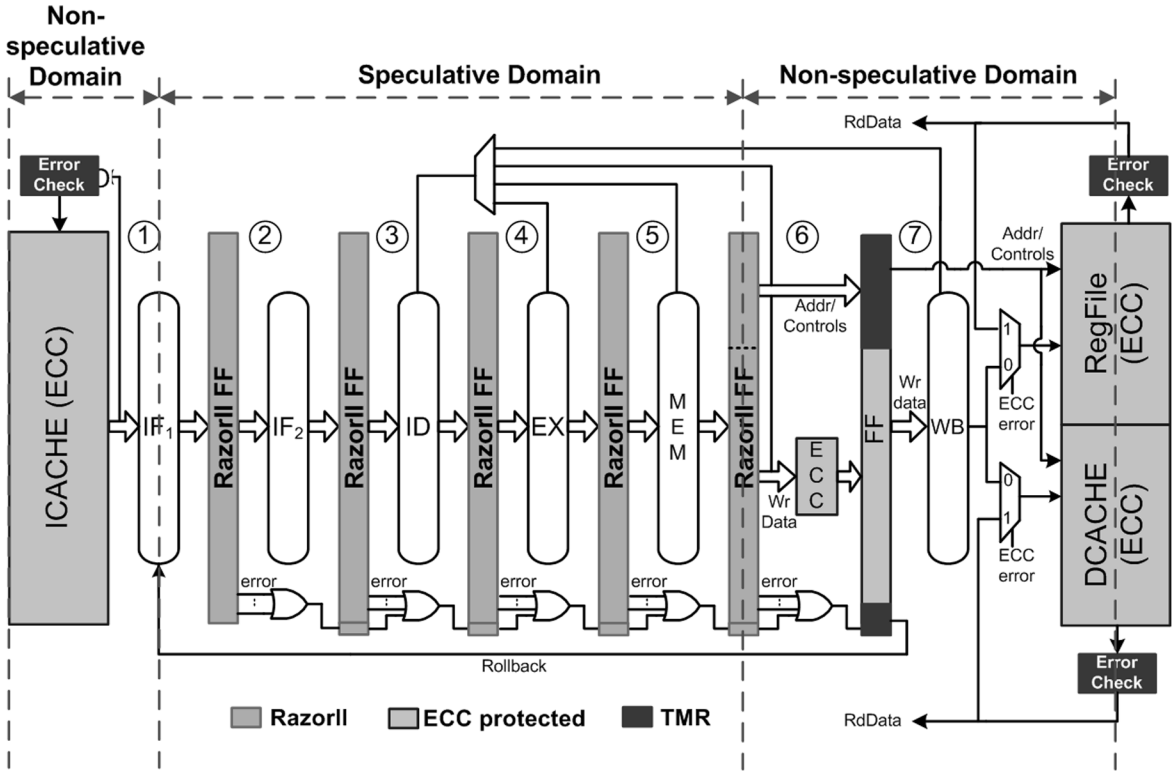


Fig. 8. RazorII processor: Pipeline design.

state registers to be corrected on-the-fly and no extra recovery mechanism is required.

Data from the speculative pipeline is encoded before the write-back stage. The data word and the corresponding redundant bits are then written into storage. Thus, the ECC encoder adds 22 redundant bits to the 64 bit data for a Dcache and Register File write access and 17 redundant bits for a 32 bit Icache write access. On a read access, the ECC decoder operates on the obtained code word and determines if a state flip has occurred, while in storage. In the event of an error, the data word is corrected and forwarded into the pipeline and the corrected word is recommitted to storage. The ECC decoder can correct one random bit flip and up to 4 consecutive bit-flips in the data word. Both ECC and TMR errors are corrected in-place and do not engage the pipeline recovery mechanism.

Replay is achieved by check-pointing the Program Counter (PC) register in the WB stage of the pipeline. The Program Counter (PC) register is passed along the Razor pipeline. When an error is detected, the entire pipeline is flushed and the PC in the fetch stage is overwritten with the PC in the WB stage. Normal instruction execution resumes from then on. The PC in the WB stage is protected from SEU through TMR. Since an erroneous instruction is re-executed through the pipeline during replay, the same instruction can suffer repeated timing errors. Hence, it is required to monitor the instruction being replayed to detect a deadlock situation. When the number of replay iterations for the same instruction reaches a certain threshold, called the "replay limit", the clock frequency is halved for 8 cycles to allow guaranteed completion. Thus, for a replay limit of 1, every timing error is accompanied by recovering at half the clock frequency. For a replay limit of "n", an errant instruction is replayed "n-1" times at the same frequency, if required, before the frequency is halved for the "n"th iteration.

A majority of timing errors at the PoFF are actually caused due to transient events, such as cross-coupling noise, which disappear during replay. Hence, it is expected that for most timing errors, replaying the erroneous instruction just once will be sufficient for completion, without having to reduce the clock frequency. This observation is borne out from our silicon measurement results, described in Section VI, where 60% of failing instructions are executed to completion in the first replay iteration without reducing the clock frequency. The replay limit is externally programmable.

V. Setting Limits to Voltage and Frequency Scaling

Voltage scaling in RazorII based systems is limited to the point where the error detection window is sufficient to detect and flag timing errors. At this safe limit, shown in Fig. 9, the critical-path computation finishes before the negative clock-edge of the next cycle and causes the internal latch-node, N, to transition while meeting the setup time of the level-sensitive latch at this clock-edge. If the critical path transition occurs after this setup time, the latch will be opaque and the transition will not be visible to the transition detection to flag an error. The Razor error-detection window $(\mathrm{T_{spec}})$ defined between the rising edge of the Detection Clock and the falling clock-edge is also shown in the figure. In the RazorII based processor, the system controller monitors error-rates and tunes itself to the PoFF where the error-rates are extremely low. However, the risk with using error-rates for self-tuning is that for an idle processor, where the observed error-rate is zero, the processor voltage and frequency

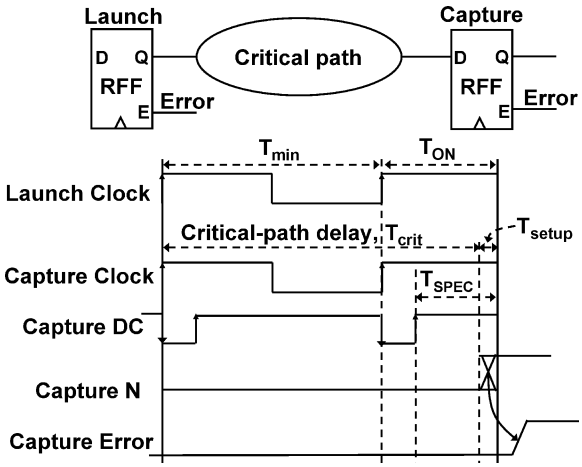


Fig. 9. The limit of safe operation.

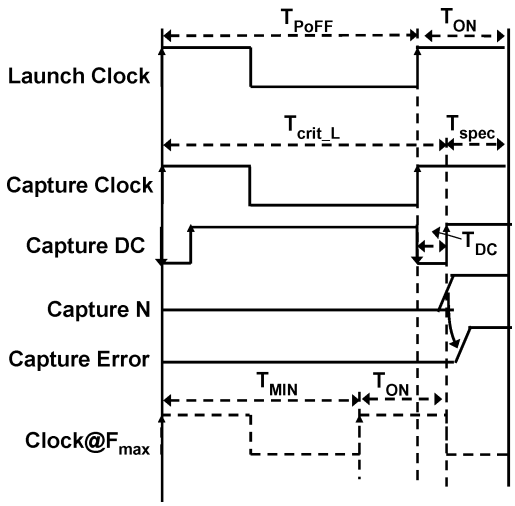


Fig. 10. Maximum frequency of operation.

can be potentially scaled too aggressively, beyond this safe limit. Thus, if the idle period is followed by a critical-path operation, the latency of the computation may exceed the error detection window, leading to system failure. Hence, it is imperative that the system is able to limit itself to a known, safe operating point even when the monitored error-rate is actually zero.

The critical path delay can dynamically vary due to changes in the ambient conditions (voltage and temperature) or ageing effects. This can cause a shift in the safe operating limit. Consequently, it may be required to periodically tune the processor to obtain this limit. In the following, we discuss several ways in which the safe operating limit can be obtained.

A. Conservative Estimation From Static Timing Analysis

In this scheme, static timing analysis is used to obtain the maximum frequency of operation. Contrary to conventional practice, timing analysis is performed with respect to the negative edge of the clock. At the maximum frequency, the critical path delay $(\mathrm{T_{crit}})$, the setup time of the latch at the negative clock-edge $(\mathrm{T_{setup}})$, the cycle time $(\mathrm{T_{min}})$ and the high phase of the clock $(\mathrm{T_{ON}})$ are related by the following equation (Fig. 9):

$$T_{crit} + T_{setup} = T_{min} + T_{ON}.$$

Thus, maximum frequency of operation $(\mathrm{F_{max}})$ is obtained as follows:

$$F_{\max} = \frac{1}{T\min} = \frac{1}{T_{crit} + T_{setup} - T_{ON}}. \tag{1}$$

The duration of the high phase of the clock, $\mathrm{T_{ON}}$, is a function of the minimum delay constraint, as explained in Section III-B, and is independent of the clock frequency. $\mathrm{T_{setup}}$ is obtained by the characterization of the RazorII latch during design time. $\mathrm{T_{crit}}$ is obtained through static timing analysis on the entire design, at a given voltage of operation. Thus, these parameters can be used in (1) to obtain the $\mathrm{F_{max}}$. At $\mathrm{F_{max}}$, it is guaranteed that all timing errors will be detected and flagged by Razor even in the worst-case condition. Thus, $\mathrm{F_{max}}$ values at different operating voltages can be stored in a look-up table.

The key advantage of this technique is its simplicity. The key disadvantage is that its reliance on static timing analysis incorporates worst-case margins in the estimation of $\mathrm{F_{max}}$. An important observation to make here is that $\mathrm{F_{max}}$ is measured with respect to the negative edge of the clock. Thus, in this technique worst-case margins are still eliminated from the positive edge but are now moved to the negative edge. It is expected that for a reasonably large value of $\mathrm{T_{ON}}$, the PoFF of the processor will be attained before $\mathrm{F_{max}}$ is reached. Thus, the conservative margins do not have any performance impact. This is in contrast with conventional look-up table approaches where margins are added to the positive clock-edge and hence have significant impact on performance and energy efficiency.

B. Worst-Case Vector Based Tuning

In this technique, the processor has two modes of operation. The normal operating mode is interrupted to enter the tuning mode wherein worst-case vectors are executed through the pipeline. In the tuning phase, the frequency of operation is adjusted until the PoFF for the worst-case vectors is reached. The worst-case vectors exercise the critical ath of the processor and can be used to obtain $\mathrm{F_{max}}$ as shown in Fig. 10. At the PoFF of the worst-case vectors, a critical-path computation causes a transition at the internal latch node, N, of the capture RazorII flip-flop just at the rising edge of the Detection Clock. Thus, the cycle time $(\mathrm{T_{PoFF}})$ at this point is related to the critical path and the pulse width of the Detection Clock $(\mathrm{T_{DC}})$ by the following equation:

$$T_{crit\_L} = T_{PoFF} + T_{DC}. \tag{2}$$

Note that in (2), $\mathrm{T_{crit\_L}}$ refers to the delay required to transition the internal latch node. Thus, it is the sum of the propagation delay through the critical path and the internal delay through the latch.

$\mathrm{T_{DC}}$ can be expressed as a function of the Razor error detection window $(\mathrm{T_{spec}})$ and the high-phase of the clock $\mathrm{T_{ON}}$ in the following equation:

$$T_{DC} = T_{ON} - T_{spec}. \tag{3}$$

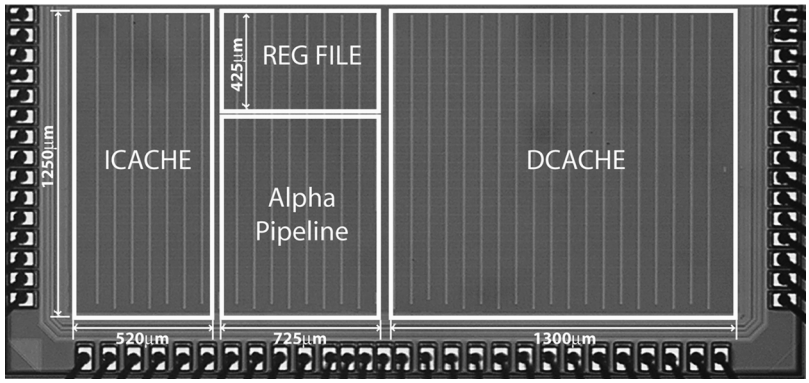


Fig. 11. Die photograph of the RazorII processor.

Thus, from (2) and (3), we can obtain $\mathrm{T_{crit\_L}}$ as a function of $\mathrm{T_{PoFF}}$ as follows:

$$T_{crit\_L} = T_{PoFF} + T_{ON} - T_{spec}. \tag{4}$$

At the maximum frequency of operation ($\mathrm{F_{max}}$), the internal latch node transitions before the negative edge of the clock. Thus, the cycle time at $\mathrm{F_{max}}$ ($\mathrm{T_{min}}$) can be expressed as a function of $\mathrm{T_{crit\_L}}$ and $\mathrm{T_{ON}}$ as

$$T_{crit\_L} = T_{\min} + T_{ON}. \tag{5}$$

Thus, from (4) and (5), we can obtain $\mathrm{F_{max}}$ as follows:

$$F_{\max} = \frac{1}{T_{\min}} = \frac{1}{T_{crit\_L} - T_{ON}} = \frac{1}{T_{PoFF} - T_{spec}} \tag{6}$$

Safety margins can then be empirically added to the $\mathrm{F_{max}}$ thus obtained. The key advantage of this scheme is that it uses the pre-existing Razor flip-flops for *in situ* delay monitoring. Thus, conservative worst-case margining at the negative edge of the clock is avoided. The key disadvantage is that the processor needs to be periodically interrupted for tuning. However, this can be achieved on-the-fly by the Operating System, especially when the system is waiting on long-latency events such as servicing a cache miss or when waiting for an interrupt.

The complexity of this approach requires further investigation before it can be qualified on silicon in a functional system. Moreover, this approach is complicated by the difficulty in obtaining worst-case vectors and replaying them deterministically. Worst-case vectors can change with variations in operating conditions. This requires infrastructure for capturing the failing vectors, storing and replaying them. We have ongoing research efforts focused on addressing these and related issues.

In the above Sections V-A and V-B, we have discussed methods for obtaining the maximum frequency of operation at a constant supply voltage. However, it is also possible to keep the operating frequency constant and scale the supply voltage to obtain the limits of error-detection. This can be achieved by a voltage control loop. The controller can be run as a software routine on the "Razor"-ized processor core. The processor can then sample the error register and instruct a regulator to increase/decrease the supply voltage according to the observed error-rate. The voltage control infrastructure is required to be a part of the entire system and can be present either on-die or on-board.

TABLE I
PROCESSOR IMPLEMENTATION DETAILS

Technology Node	CMOS 0.13µm
Dimensions	2700µm * 1250µm
Number of Critical RazorII Flip-flops	121 of 826
RazorII Power Overhead	1.2%
Number of buffers added for hold-time fixing	1924
Buffer insertion power overhead	1.3%
Total number of transistors	260K
Operating voltage	1.2V
Frequency	185MHz @ 1.2V
Power@185MHz	94.3mW

VI. SILICON MEASUREMENT RESULTS

We designed and built a 64 bit processor executing a sub-set of the Alpha instruction set in 0.13 micron technology which uses RazorII for supply voltage control. The die photograph of the processor is shown in Fig. 11 and the relevant implementation details are provided in Table I. 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. We achieved fully functional silicon across a range of voltage from 0.8 V to 1.2 V. A 32-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.

A. RazorII Clocking Scheme

The core frequency of the processor is controlled by an internal Clock Generation Unit (CGU). The CGU generates clock frequencies in the range between 50 MHz to 370 MHz. The CGU has a separate voltage domain that is not scaled. Hence, the core frequency remains constant even when the core voltage is dynamically scaled. The frequency output of the CGU is externally programmable.

The minimum delay constraint for a RazorII flip-flop is defined by the duration of the high clock phase. Since, all pipeline

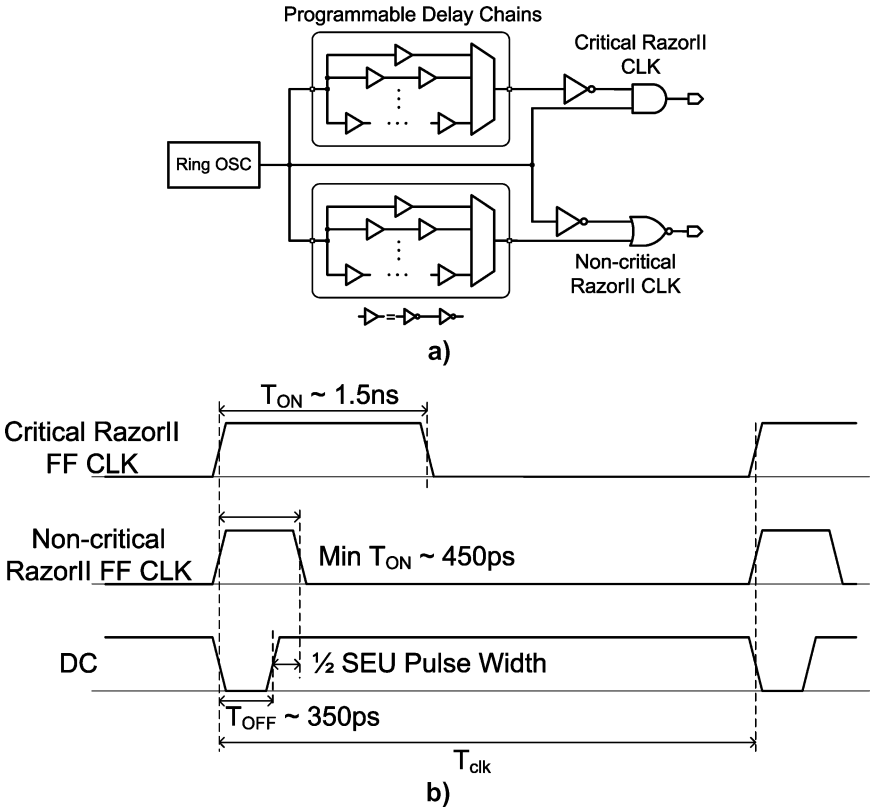


Fig. 12. Clocking scheme in the RazorII processor. (a) Clock generation unit; (b) asymmetric clocks.

registers are required to be RazorII flip-flops for SEU tolerance, the excessive buffer insertion required to satisfy this constraint can have prohibitive area and power impact. We solve this problem by having separate clock trees for timing-critical flip-flops and those that have sufficient slack, as shown in Fig. 12(a). We use a Ring Oscillator (RO) for frequency synthesis. The clock output of the ring oscillator has approximately 50% duty cycle. Delay chains are then used to tune the high phase of the clock to create separate asymmetric clocks for the timing-critical and the non-critical flip-flops. The delay through the chains determines the duration of the high-phase of the individual clocks. The frequency output of the RO and the high-phase of the asymmetric clocks are all separately tunable via an external interface.

The conventional design flow was modified to account for the specific clocking requirements of the critical RazorII flip-flops. This was achieved in two stages. In the first stage, conventional clocking is used to synthesize a single clock tree for all the flip-flops in the design. Static timing analysis is performed on the routed and extracted netlist to identify the critical flip-flops. In the second stage, we mark the critical flip-flops identified in the previous stage in the original placed design, without the routing information. We re-synthesize the clock-trees with the asymmetric clocks and route the clock with the longer high-phase to the critical flip-flops. Buffer insertion is then performed to fix the minimum-delay violations in the design. Thereafter, the design is routed and the parasitics extracted. Sign-off is achieved through timing verification in the final routed and extracted netlist.

Fig. 12(b) illustrates the asymmetric clocks used for different flip-flops in the design, at the slow corner. We use larger speculation windows for timing-critical RazorII flip-flops whereas non-critical flip-flops require much smaller speculation duration. We chose to route the clock with the larger speculation window to the top-15% most-critical RazorII flip-flops. These represented 121 out of a total of 876 flip-flops. Thus, in this processor, the timing critical flip-flops had a clock with a 40% duty cycle resulting in a 25 F04 detection window while non-critical flip-flops had a 13% clock duty cycle to minimize buffer insertion. The high phase, TON, of the critical RazorII flip-flops was tunable with a range from 850 ps to 1500 ps while TON for the non-critical flip-flops was tunable from 450 ps to 700 ps. A total of 1924 buffers were added to meet the hold time constraint, which added a 1.3% power overhead.

As explained in Section III-C, the minimum overlap between the high phase of the clock and the detection clock (DC) is required to be half of a SEU pulse width. An additional concern that affects the duration of the high phase is the risk of attenuation during propagation through the clock-tree. Attenuation can occur due to asymmetric rise or fall times through the buffers in the clock-tree or power-supply jitter affecting the clock-tree. This can be an issue for the non-critical clock, which uses a narrower high-phase compared to the critical clock (450 ps versus 1.5 ns at the slow corner). In such a case, a wider high-phase maybe required for the non-critical clock. In our implementation, we added 100 ps as the difference between the DC pulse-width and the high-phase of the non-critical clock. This was sufficient for the correct operation of the chip. How-

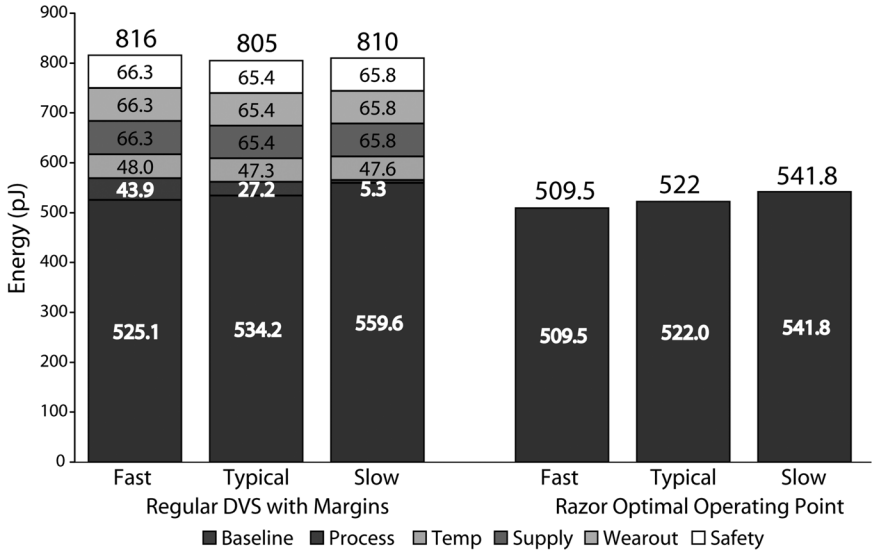


Fig. 13. Distribution of Razor percentage energy savings. In all, 33 dies were tested and measured.

ever, we also provided capability for post-silicon tuning of the high-phase (from 450 ps to 750 ps) which was not required to be exercised for the chip to operate. Attenuation is not a concern for the DC clock pulse since it was locally generated within each RazorII flip-flop.

An important observation to make here is that the only restriction imposed by the RazorII clocking scheme is on the duration of the high phase required to meet the minimum delay constraint at the destination RazorII flip-flops. Hence, all the conventional clocking techniques such as clock-gating and useful skew insertion can be used, as is, with RazorII without any modification to the existing methodology. Of course, such techniques may worsen the minimum-path constraints. This can be addressed in the conventional manner through additional buffer insertion on the violating paths.

The impact of clock uncertainties which affect the rising clock-edge, such as cycle-to-cycle jitter, impacts RazorII technology in the same way as it affects conventional clocking techniques. However, RazorII, being naturally resilient against timing uncertainties, can tolerate a higher level of such jitter, compared to conventional techniques. Phenomena that affect both clock edges, such as duty-cycle jitter, have an impact on the high-phase of the clock at the destination flip-flop and hence, affect the minimum-path constraint. Additional buffer insertion may be required to account for minimum-path violations caused due to duty-cycle jitter.

Since the asymmetric clocks were digitally generated, care was taken to reduce the jitter induced by power-supply noise in the CGU. The CGU had additional metallization in the power-grid and had sufficient number of decoupling capacitors to reduce supply voltage ripple to the high-frequency RO. Alternatively, analog techniques can also be used for jitter compensation within the CGU.

B. Total Energy Savings

We measured the energy savings from RazorII based DVS on 32 different dies at 185 MHz operating frequency. Fig. 13 shows the energy savings for three different chips at the fast, slow and the typical corners respectively. These chips are labeled according to the corner that they represent.

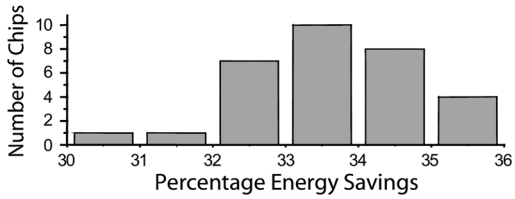


Fig. 14. Total energy savings with RazorII.

The first set of bars show the energy consumption when Razor error correction is disabled and the chips are operated at the worst-case voltage. We obtain the worst-case operating voltage by adding margins to the PoFF of the slowest chip of the lot (1.21 V). We added an estimated 5% of the nominal operating voltage (1.2 V), or 60 mV, as margin for power supply uncertainties, wear-out effects and safety, respectively. Temperature margins were measured by the shift in the PoFF of the worst-case chip at 85C versus that at 25C. The PoFF of the worst-case chip at 25C and 85C was measured to be 1.21 V and 1.26 V respectively, a shift of 50 mV. Thus, by adding these margins to the PoFF of the slowest chip, we obtained the worst-case operating voltage to be 1.44 V. At this operating voltage, correct operation is guaranteed for all the tested dies, across all operating conditions. A key observation to make here is that these margins are optimistic since the actual process spread is significantly worse than what we can obtain with a limited sample size of 32 dies.

The first set of bar graphs in Fig. 13 show the Energy Per Instruction (EPI) of the chips-under-test when operating at 1.44 V. For each chip, we measured the contribution of each category of margins to the overall energy consumption. The energy due to process variations margin was measured by the difference in energy consumption when operating at the PoFF of the worst-case chip (1.21 V) versus operating at its own PoFF, at 25C. For the "fast" chip, this was measured to be 44 pJ per instruction. This is significantly greater than the process variations margin for the "slow" chip which was measured to be 5.3 pJ. This is because

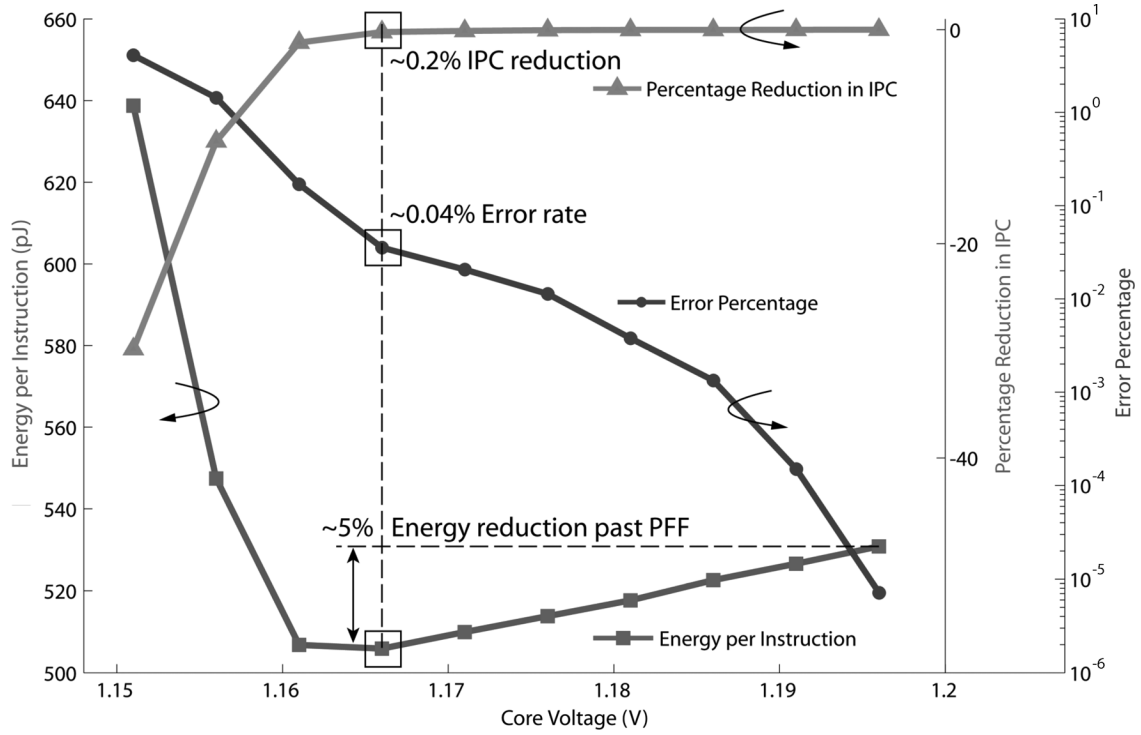


Fig. 15. Sub-critical operation in RazorII.

the PoFF of "slow" chip (1.205 V) is very close to the PoFF of the worst-case chip (1.21 V). The energy due to temperature margins was measured by the difference in energy when operating at the PoFF of the worst-case chip at 85C (1.26 V) versus operating at 1.21 V. At the worst-case voltage, the "fast" chip consumes 816 pJ per instruction. The "typical" and the "slow" chips consume 805 pJ and 810 pJ per instruction, respectively.

For the second set of bar graphs shown in Fig. 13, we enabled Razor error correction and operated the chips at their own optimal operating voltages (Fig. 1). At this voltage, the exponential increase of recovery energy compensates for the quadratic reduction of the pipeline operations energy. The EPI of the "fast" chip was measured to be 509.5 pJ at the optimal voltage. This translated to a net saving of 37.5% when compared to 816 pJ consumed at the worst-case voltage. The energy savings for the "typical" and the "slow" chips were measured to be 35% and 33%, respectively. The elimination of higher process variations margin for the"fast" chip compared to the rest, leads to greater overall energy savings at the optimal point. We obtain, on an average, 33% energy savings over the worst-case for all the chips tested, as shown in the histogram in Fig. 14.

C. RazorII Sub-Critical Operation

Fig. 15 shows the measured error-rate, IPC and energy-per-instruction of the "fast" chip as a function of the supply voltage in the sub-critical voltage regime at 185 MHz. The voltage at the PoFF is 1.197 mV. At this voltage, the error-rate is extremely low of the order of 10^{-6}. As the supply voltage is reduced below the PoFF, the error-rate increases exponentially. Initially, the error-rate is still extremely low and the pipeline operations energy dominates over the recovery energy. Consequently, the overall energy-per-instruction reduces quadratically with the supply voltage. At the optimal operating voltage of 1.165 V, the EPI is measured to be 509.5 pJ for an error-rate of 0.04% and an IPC degradation of 0.2%.

Beyond the optimal energy point, the recovery energy for the exponentially increasing error-rates dominates the overall processor energy. Hence, both the IPC degradation and the energy-per-instruction of the processor show an exponential trend. This greatly complicates the design of a voltage controller which can dynamically tune the supply voltage for the optimal operating point. In addition, the energy savings at the optimal point compared to the PoFF is not significant (5%). Hence, it is beneficial to operate at the PoFF rather than at the optimal voltage.

Fig. 16 shows the measured statistics on the number of replay iterations required to commit an erroneous instruction. An erroneous instruction can suffer repeated timing errors during replay. A possible deadlock situation can be avoided by recovering at half the frequency to guarantee completion for a repeatedly failing instruction. As explained in Section IV, the number of replay iterations allowed for a failing instruction before frequency is halved, is called the "replay limit". In Fig. 16, we plot the number of times the replay limit is reached as a function of the replay limit. The runtime for the code being executed is also plotted against the replay limit. We see that the runtime for the replay limit of 1 is much higher than that for the replay limit of 2. This is because for the replay limit of 1, recovery for every timing error occurs at half the clock frequency. On the contrary, for a replay limit of 2, most instructions complete when re-executed at the same frequency.

Those instructions that don't complete in the first replay keep failing repeatedly for subsequent replay iterations until the re-

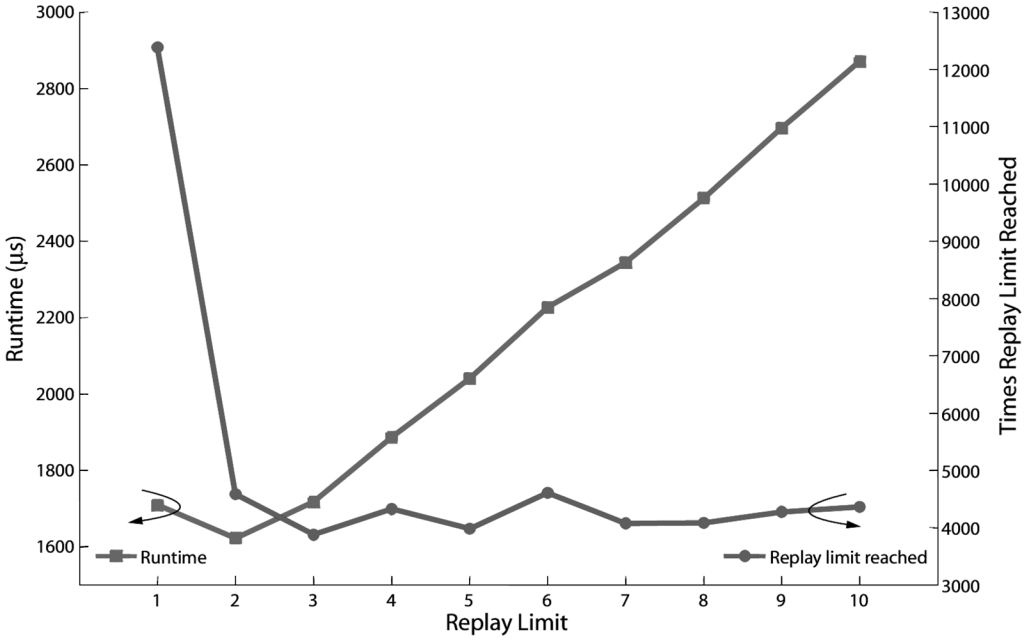


Fig. 16. Run-time versus replay trade-off.

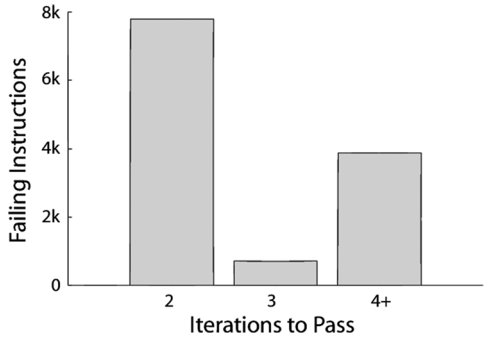


Fig. 17. Histogram of instructions as a function of replay iterations.

play limit is reached. Therefore, as the replay limit is increased beyond 2, the runtime also linearly increases. These observations are also borne out from the histogram in Fig. 17. As can be seen, 60% of erroneous instructions require a limit of 2 to complete. This number drastically reduces as the limit is increased. For higher values of replay limits, the numbers of times the replay limit is reached is almost constant.

D. Soft Error Rate Radiation Tests With RazorII

Fig. 18 shows the setup used for accelerated radiation tests performed on the processor to quantify the Soft Error Rate (SER) tolerance provided by RazorII. These tests were performed in the Breazeale Nuclear Reactor at the University of Pennsylvania. Thermal neutrons with a neutron flux of 3.5×10^7 neutrons/cm^2 were used for irradiating the chip. SER protection in the SRAM arrays and Register File is provided by ECC. Other architectural state registers are protected by TMR and all the pipeline registers are protected using RazorII error correction.

Table II lists the radiation tests performed on the processor. In the first test (Test 1), we completely disable error correction by

Fig. 18. SER test setup.

Razor, ECC and TMR. The test code is scanned into the Icache, error protection is disabled and the code is executed while the processor is being simultaneously irradiated. This test is performed at 0.8 V with sufficiently low operational frequency such that timing errors do not occur. Thus, all the observed errors are due to bit-flips in the state-holding nodes due to particle strikes. As expected, the final execution output is incorrect. When error detection is enabled (Test 2) the processor is able to detect and correct the SER induced errors. This was verified for different operating voltages (0.8 to 1.0 V). The soft-error rate was recorded for memory and pipeline elements. In Test 3, we allow the processor to execute when the frequency of operation is increased beyond PoFF causing delay errors to occur in addition to SER. Although the delay errors completely overwhelm errors due to SER, RazorII is able to detect and correct all of them and the processor continues to operate correctly.

TABLE II
SER RADIATION TESTS

Radiation Test Description	Error Detection (TD enabled)	Vdd (V)	Razor Errors	ECC Errors	Delay error rate > 0	Correct program execution
1 – No timing errors and error detection off	Off	0.8	NA*	NA*	No	No
2 – No timing errors with error detection	On	0.8	4	5	No	Yes
	On	0.9	6	4		
	On	1.0	4	6		
3 – Both SER and timing errors with error detection	On	0.8	33M **		Yes	Yes

*When TD is disabled the counter of errors in the ECC and RazorII FF us disabled

**Beyond PoFF (Delay error rate > 0), timing errors overwhelm SER, but they both are detected and corrected by the RazorII mechanisms

VII. CONCLUSION AND FUTURE WORK

In this paper, we presented a new technique for error detection and correction called RazorII which provides energy efficiency as well as SER tolerance. We presented the design of a novel delay-error tolerant flip-flop that relies on detecting spurious transitions at its input in order to flag timing errors on the processor critical paths. We demonstrated this technique on a 64 bit Alpha processor and obtained, on an average, 33% energy savings over the worst-case. In addition, we also demonstrated correct operation of the processor with RazorII when irradiated with high-energy particles in a nuclear reactor. We discussed issues and proposed solutions for setting voltage-scaling limits in Razor DVS. Future research into Razor focuses on building robust and field-deployable systems using Razor as a means of simultaneously satisfying conflicting power and performance requirements in the face of rising silicon uncertainties.

ACKNOWLEDGMENT

The authors are grateful to Prof. Vijaykrishnan Narayanan and Ramakrishnan Krishnan at Pennsylvania State University for their help with the radiation tests. Thanks are also due to ARM Ltd. for funding this work and to MOSIS (www.mosis.org) for fabricating the test chip.

REFERENCES

[1] A. Drake, R. Senger, H. Deogun, G. Carpenter, S. Ghiasi, T. Ngyugen, N. James, and M. Floyd, "A distributed critical-path timing monitor for a 65 nm high-performance microprocessor," in *2007 IEEE ISSCC Dig.*, Feb. 2007, pp. 398–399.

[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] S. Das, D. Roberts, S. Lee, S. Pant, D. Blaauw, T. Austin, T. Mudge, and K. Flautner, "A self-tuning DVS processor using delay-error detection and correction," *IEEE J. Solid-State Circuits*, pp. 792–804, April 2006.

[6] J. Tschanz *et al.*, "Adaptive frequency and biasing techniques for tolerance to dynamic temperature-voltage variations and aging," in *2007 IEEE ISSCC Dig.*, Feb. 2007, pp. 292–293.

[7] B. Stackhouse, B. Cherkauer, M. Gowan, P. Gronowski, and C. Lyles, "A 65 nm 2-billion-transistor quad-core itanium processor," in *2008 IEEE ISSCC Dig.*, Feb. 2008, pp. 92–93.

[8] S. Rangan, N. Mielke, and E. Yeh, "Universal recovery behavior of negative bias temperature instability," in *Proc. IEEE Int. Electron Devices Meeting (IEDM'03)*, Dec. 2003, p. 341.

[9] A. M. Yassine, H. E. Nariman, M. McBride, M. Uzer, and K. R. Olasupo, "Time dependent breakdown of ultrathin gate oxide," *IEEE Trans. Electron Devices*, vol. 47, no. 7, pp. 1416–1420, Jul. 2000.

[10] W. J. Van Gils, "A triple modular redundancy technique providing multiple-bit error protection without using extra redundancy," *IEEE Trans. Computers*, vol. C-35, no. 7, pp. 623–631, Jul. 1986.

[11] S. J. Piestrak, A. Dandache, and F. Monteiro, "Designing fault-secure parallel encoders for systematic linear error correcting codes," *IEEE Trans. Reliability*, vol. 52, no. 4, pp. 492–500, Dec. 2003.

[12] K. A. Bowman, J. W. Tschanz, N. S. Kim, J. C. Lee, C. B. Wilkerson, S. Lu, T. Karnik, and V. De, "Energy-efficient and metastability-immune timing-error detection and instruction-replay-based recovery circuits for dynamic-variation tolerance," in *2008 IEEE ISSCC Dig.*, Feb. 2008, pp. 402–403.

[13] D. Ernst, N. S. Kim, S. Das, S. Pant, T. Pham, R. Rao, C. Ziesler, D. Blaauw, T. Austin, T. Mudge, and K. Flautner, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 36th Annual IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2003, pp. 7–18.

[14] D. Blaauw, S. Kalaiselvan, K. Lai, W. Ma, S. Pant, C. Tokunaga, S. Das, and D. Bull, "Razor II: In situ error detection and correction for PVT and SER tolerance," in *2008 IEEE ISSCC Dig.*, Feb. 2008, pp. 400–401.

[15] T. Kuroda, K. Suzuki, and S. Mita, "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.

[16] J. Kao, M. Miyazaki, and A. R. Chandrakasan, "A 175-mV multiply-accumulate unit using an adaptive supply voltage and body bias architecture," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1545–1554, Nov. 2002.

Shidhartha Das (S'03–M'08) 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 at Ann Arbor in 2005.

His research interests include micro-architectural and circuit techniques for low-power and variability-tolerant digital IC design. Currently, he is a Staff Engineer working for ARM Ltd., Cambridge, UK, in the Research and Development group.

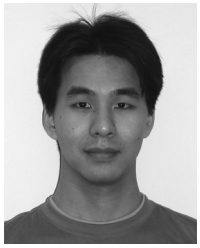
Carlos Tokunaga (S'98) received the B.S. degree in electronics engineering from the University of Los Andes, Bogota, Colombia in 2001, and the M.S degree in electrical engineering from the University of Michigan, Ann Arbor, in 2005, where he is currently pursuing the Ph.D. degree.

In fall 2006 and spring 2008 he was with the Circuits Research Lab, Intel, Hillsboro, OR, where he was a graduate intern. His research interests include VLSI design with particular enphasis on low-power, high-performance and security-based systems circuit design.

Sanjay Pant (M'08) received the B.Tech degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, in 2001, and the M.S. and Ph.D. degrees in electrical engineering from the University of Michigan at Ann Arbor in 2004 and 2007.

In fall 2004 and summer 2005, he was with the Strategic CAD Labs, Intel Corporation, Hillsboro, Oregon, where he worked as a Graduate Intern. Currently, he is a Senior Design Engineer in the Advanced Power Technology Group, Advanced Micro Devices, Fort Collins, Colorado.

His research interests include low power VLSI design and signal integrity issues in power distribution networks.

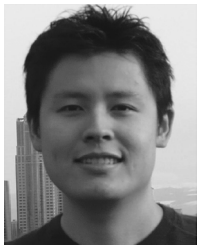


Wei-Hsiang Ma (S'08) was born in Taipei, Taiwan. He received the B.S. degree in electrical engineering from the National Taiwan University in 2002, and the M.S. degree in electrical engineering and computer science in 2007 from the University of Michigan, Ann Arbor, where he is currently working towards the Ph.D. degree.

His research interests include low-power and high-performance circuit technologies and design methodologies.

Sudherssen Kalaiselvan received the B.E. degree with honors in electrical and electronics engineering from Birla Institute of Technology and Science, Pilani, Rajasthan, India, in 2005 and the M.S.E. degree in computer science from the University of Michigan, Ann Arbor, in 2007.

He is now a Design Engineer II with Advanced Micro Devices, Sunnyvale, California. His research interests include low-power VLSI designs, SEU tolerant designs and high-speed circuits.



Kevin Lai was born in Taipei, Taiwan, and emigrated to Canada at 14. He received the B.S. degree in electrical engineering from the University of British Columbia, Canada, in 2005, and the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor, in 2007.

He is now working as a design engineer for the next-generation microprocessor group at Intel in Hillsboro, Oregon.

David M. Bull received the B.Sc. degree in computer science from Royal Holloway College, University of London, U.K., in 1991.

He is a consultant engineer at ARM Ltd., Cambridge, U.K. He joined ARM in 1995, and spent nine years working on various aspects of processor development including micro-architecture and circuits. He has worked on the ARM9 and ARM11 processor families processor, and was the design lead for the ARM926EJ-S. Since 2004, he has focused on research into advanced circuit and micro-architectural techniques, and has lead the ARM RAZOR research project.

David T. Blaauw received the B.S. degree in physics and computer science from Duke University in 1986, and the Ph.D. degree in computer science from the University of Illinois, Urbana, in 1991.

Until August 2001, he worked for Motorola, Inc. in Austin, TX, were he was the manager of the High Performance Design Technology group. Since August 2001, he has been on the faculty at the University of Michigan where he is currently a full Professor. His work has focused on VLSI design and CAD with particular emphasis on circuit design and optimization for high performance and low power applications.

Dr. Blaauw was the Technical Program Chair and General Chair for the International Symposium on Low Power Electronic and Design and was the Technical Program Co-Chair and member of the Executive Committee the ACM/IEEE Design Automation Conference. He is currently a member of the ISSCC Technical Program Committee.