Active Management of Timing Guardband to Save Energy in POWER7*

Charles R. Lefurgy[†], Alan J. Drake[†], Michael S. Floyd[‡], Malcolm S. Allen-Ware[†],
Bishop Brock[‡], Jose A. Tierno[+], and John B. Carter[†]

IBM[†]
11501 Burnet Rd.
Austin, TX, USA

IBM[‡]
11400 Burnet Rd.
Austin, TX, USA

IBM[+]
1101 Kitchawan Rd.
Yorktown Heights, NY, USA

{lefurgy,adrake,mfloyd,mware,bcbrock,tierno,retrac}@us.ibm.com

ABSTRACT

Microprocessor voltage levels include substantial margin to deal with process variation, system power supply variation, workload induced thermal and voltage variation, aging, random uncertainty, and test inaccuracy. This margin allows the microprocessor to operate correctly during worst-case conditions, but during typical conditions it is larger than necessary and wastes energy. We present a mechanism that reduces excess voltage margin by (1) introducing a *critical path monitor* (CPM) circuit that measures available timing margin in real-time, (2) coupling the CPM output to the clock generation circuit to adjust clock frequency within cycles in response to excess or inadequate timing margin, and (3) adjusting the processor voltage level periodically in firmware to achieve a specified average clock frequency target. We implemented this mechanism in a prototype IBM POWER7 server. During better-than-worst case conditions our guardband management mechanism reduces the average voltage setting 137-152 mV below nominal, resulting in average processor power reduction of 24% with no performance loss while running industry-standard benchmarks.

Categories and Subject Descriptors

B.8 [**Hardware**]: Performance and Reliability

General Terms

Measurement, Performance, Design, Reliability, Experimentation

Keywords

Timing margin, energy-efficient, critical path, digital phase-lock loop, feedback control, POWER7

1. INTRODUCTION

Server microprocessors must operate reliably across a wide range of environmental conditions and workloads. The timing margin for circuits in the microprocessor is affected by manufacturing process, thermal fluctuation, frequency changes, voltage slewing, and aging. Selecting an operating voltage that is high enough to account for worst-case conditions, without violating power budgets and thereby limiting performance, is a growing challenge [18].

Typically operating voltage is determined during the manufacturing test and characterization process based on the chip's intended operating frequency range and environment. Extra margin or *voltage guardband* is added to the operating voltage to guarantee proper circuit timing even during worst-case voltage droop events. Additional guardband is included to cover for unknown variables and test inaccuracy.

Under typical conditions, the voltage droop experienced by circuits is much smaller than under worst-case conditions, and the circuits could operate correctly with a smaller guardband. Using an unnecessarily large voltage guardband wastes energy.

Critical Path Monitors (CPM) are on-chip sensors that measure the timing margin available to circuits on the chip [5]. In this work, we propose using CPMs to measure available timing margin dynamically and to adjust the operating voltage to maintain a fixed *timing guardband* determined during worst-case characterization. The resulting mechanism reduces power consumption for typical workloads, while still allowing worst-case workloads to operate at the maximum frequency used in the characterization process.

In a prototype POWER7 server [10], we implemented two cooperating feedback controllers to operate the microprocessor with a fixed timing guardband, illustrated in Figure 1. The first feedback controller, implemented in the POWER7 microarchitecture, prevents timing errors at short time scales by lowering the processor core clock frequency via the digital phase-locked loop (DPLL) circuit when the CPMs sense a loss of timing guardband below a calibrated limit. The second feedback controller, implemented in a power-management microcontroller's firmware, adjusts the processor voltage to achieve a desired performance level (clock frequency) on a longer time scale. We have evaluated many workloads running on this server and measured an average processor power consumption reduction of 24% without loss of performance with active guardband management.

The main contributions of this paper are:

- Developing an architecture for controlling the amount of timing guardband dynamically.

MICRO 11, December 3-7, 2011, Porto Alegre, Brazil

* This work was supported in part by the Defense Advanced Research Projects Agency under contract #HR0011-07-9-0002.

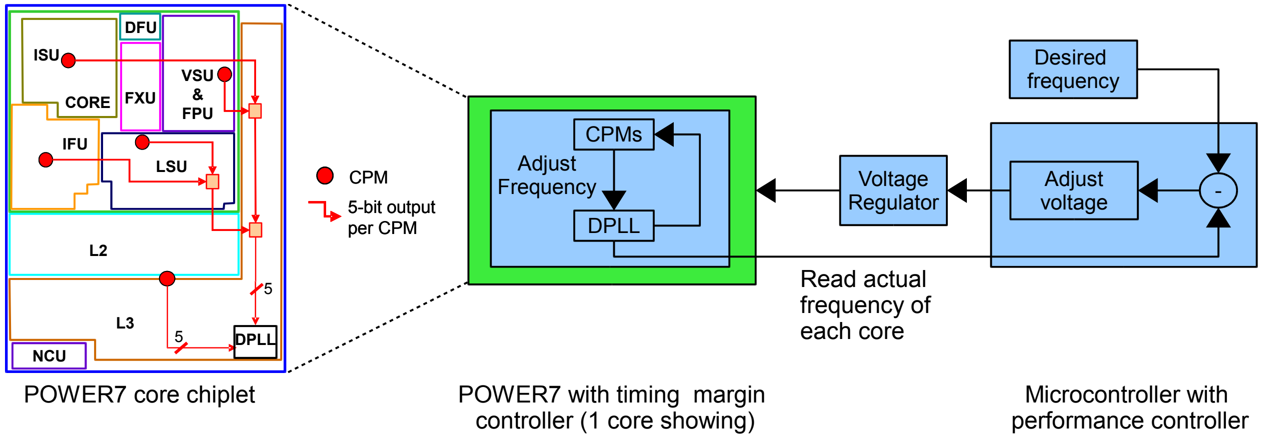


Figure 1: Undervolting architecture.

- Developing a method for calibrating the CPMs to a desired level of guardband.
- Developing a timing margin controller capable of protecting the processor against short-term noise events and validating its behavior during worst-case noise.
- Developing a performance controller that dramatically reduces microprocessor power consumption without harming performance by converting excess timing margin into a voltage reduction.
- Implementing a prototype server that manages microprocessor timing guardband and validating it saves energy using industry-standard benchmarks.

In Section 2 we provide background on traditional characterization methods to determine voltage, discuss how the power supply load-line typically present in systems affects the microprocessor voltage, and compare our approach with the well-known Razor mechanism [6, 3]. In Section 3 we present our architecture for timing guardband management. In Section 4 we provide the experimental results of our prototype. We review the relevant literature in Section 5 and conclude in Section 6.

2. BACKGROUND

2.1 Traditional Determination of Voltage

Operational voltages in microprocessors are set to ensure correct functionality at a given frequency over the lifetime of the microprocessor. Each processor generation has a number of frequency targets, or *sorts*, defined by an acceptable level of performance for a given power dissipation. The voltages for processors in a sort are not set uniformly, since process variation between processors may still be considerable. To find the operational voltage for a particular processor at a given sort frequency, a margin (expressed as a percentage of the sort frequency) is added to the sort frequency. The voltage is then adjusted while running a self-checking power exerciser program until the voltage limit of correct operation is determined. This voltage and the sort frequency (without the margin described above) constitute the operating point of that particular microprocessor.

While the operating margin above is expressed in terms of increased sort frequency, it can be expressed as an increase in required voltage – mandating that the voltage be set higher than necessary to run the microprocessor at the sort frequency absent any failure-causing noise. By extension, margin can be reduced by increasing the operating frequency or by reducing the operating voltage. Throughout this section, margin typically will be expressed as a voltage margin.

Voltage margins compensate for noise processes (such as process variation, system power supply variation, workload-induced thermal and voltage variation, load-line overcorrection, long-term wear-out, and random uncertainty and test inaccuracy) that directly affect latch-to-latch path delay. Critical paths are those timing paths in which the noise-induced delay changes are sufficient to cause data failures. For a given microprocessor, there are usually several critical paths inherent in its design that can cause hardware failures depending on the type of noise present and the workload being executed. These paths limit performance since they dictate the maximum allowable frequency and/or minimum allowable voltage for a desired power budget. To ensure no data loss ever occurs, voltage margins must be large enough to provide adequate timing under the worst noise profile.

Most noise processes are systematic and caused by changes in temperature and workload-induced voltage droop. By adjusting voltage and/or frequency to track the systematic noise of the processor, it becomes possible to save energy during low-temperature and low-activity periods while guaranteeing performance during high-temperature and high-activity periods. The faster a system detects a noise event and compensates for it, the more its margins can be reduced to only the amount needed to protect against the fastest random events and testing uncertainty.

2.2 Microprocessor Power Supply

The voltage levels experienced by server microprocessor circuits vary significantly with workload, even when the voltage settings provided to the Voltage Regulator Modules (VRMs) are constant. As workload activity increases, the average voltage experienced by circuits on the chip tends to drop. The fact that circuit voltages are actually higher under lower load conditions is one of the major effects that can be exploited by an active guardband management mechanism to lower system power, since at a given frequency and temperature a lighter workload can operate correctly at lower circuit voltages than a heavier workload.

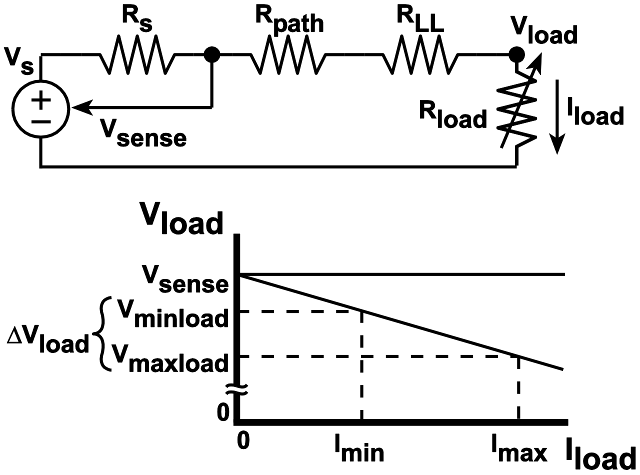


Figure 2: Load line.

Figure 2 is a simplified diagram of a VRM distribution circuit, including output resistance, sense point, path resistance, and load-line resistance. The figure illustrates the causes of voltage variability. First, there is a physical IR voltage drop from the path through the board, package, and on-chip power grid resistance, R_{path}. Variation in average current drawn by a workload is due to variation in the number and types of chip resources in use, stalls on memory, and even the particular data values being operated on. Given the low operating voltages and high performance of modern server processors, operating currents that vary by 75A depending on workload and power management state are possible. This leads to 7.5-37.5 mV variability through typical 100-500 μΩ board and package resistances.

The second cause of voltage variability is the *load line*, resistor R_{LL} often implemented by the power supply. The load line specifies a narrow range of allowed power supply output voltages that varies with load [13]. The load line typically models a resistive path between the power supply and the load, thus voltage at the load drops as current increases. Load-line slopes of 0.5-1.5 mV/A are typical, leading to 37.5-112.5 mV variability at the circuits for a 75A current swing. The effect on circuit voltage of a combined load line plus path resistance is computed from Ohm's law:

$$\Delta V = \Delta I * (R_{LL} + R_{path}) \qquad \text{(Eqn. 1)}$$

The load line increases guardband at lower load levels, where the negative effects on power of slightly higher operating voltages are reduced. The load line also provides a level of protection against voltage droop due to load spikes. Since a workload at a low load level is operating at a higher voltage than the final voltage under the heavier load, on-chip de-coupling capacitance has a higher charge to provide current to offset a voltage droop and the load line allows the VRM time to respond to an instantaneous increase in load without the circuits experiencing a voltage violation.

System firmware that controls the power supplies models the combined effects of the load line plus the path resistance, setting the voltage at the VRM to a level that guarantees a safe circuit voltage under worst-case load conditions. In Section 3 we describe how an environmentally-driven frequency controller is also able to respond to load variation, effectively taking over some of the responsibility for guardband management that had previously been implemented by the load line. Instead of using higher voltages and bulk charge storage to protect against voltage droop, our timing margin controller uses rapid frequency changes to protect circuits against voltage-induced failures.

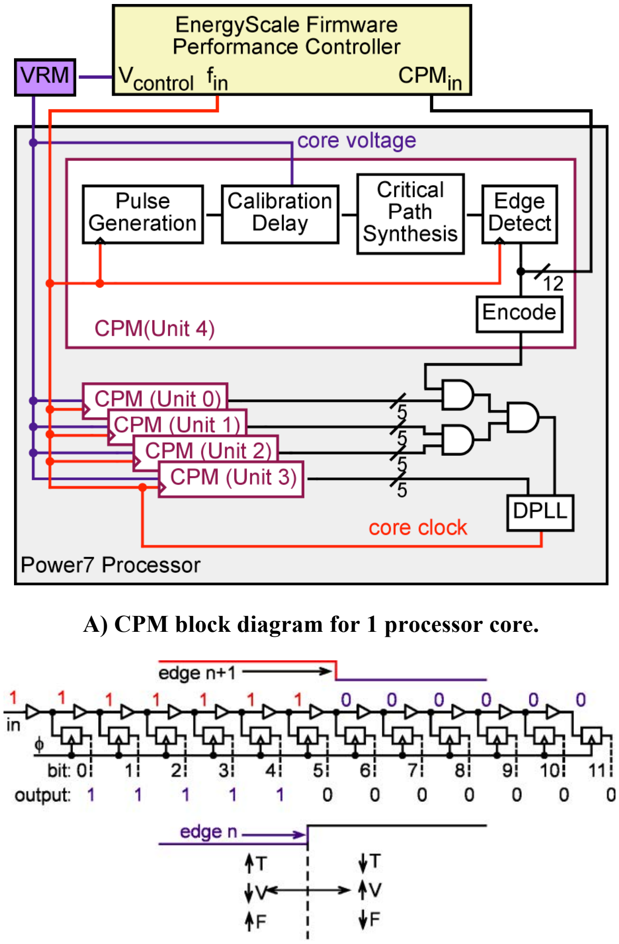
2.3 Contrasting Razor and CPMs

Probably the best-known technique for adaptively reducing circuit voltage guardbands in the face of process, voltage, temperature, and workload variation is *Razor* [6, 7, 2, 3]. Razor's inventors propose two mechanisms that add logic along critical paths to detect soft errors induced by inadequate supply voltage, and rollback and replay operations when soft errors are detected. RazorI replicates flip-flops along critical paths and samples the output of the critical path twice. A *speculative value* is sampled at the end of each cycle and a *known-good value* is sampled when the inputs to the second flip-flop are guaranteed to be stable. During normal operation, the speculative output is used by the next pipeline stage and the circuit operates at the target frequency. However, if the speculative and known-good values differ after any cycle, a global recovery signal aborts the speculative execution of the next pipeline stage, replaces the speculative value with the known-good value, restores the pipeline to its correct state, and re-executes the subsequent pipeline stage. RazorII addresses soft errors with an architecture-level replay mechanism. Errors are detected by flagging spurious transitions at critical path endpoints and via traditional mechanisms for detected soft errors in logic and registers. RazorII has a higher transistor and performance overhead than RazorI, but is more stable and avoids relying on a metastability-detector and timing-critical pipeline recovery path. RazorII overhead is significant because checking/retry must be implemented on paths that normally are not susceptible to soft errors. In conventional processors, normally only SRAMs, register files, and I/O interface must be protected.

Razor can be more aggressive in eliminating guardband than our CPM-based approach, because it can recover when too much guardband is removed. The tight coupling between CPMs and the DPLL circuitry allows us to react very quickly to voltage droop, but we maintain a small guardband to handle the most extreme dI/dt situations. As a result, Razor can reduce power by 33-50% in a 120 MHz, 130-180 nm processor [3], while CPMs reduce power 24% in a 3.864 GHz, 45 nm POWER7 processor. However, Razor has a larger impact on area, performance, and overall design and verification effort. Razor adds a 1-3% area overhead, whereas the CPM circuits and DPLL changes add only 0.046 mm^2 of silicon, representing 0.2% of the processor core (21.04 mm^2) or 0.12% of the processor core plus associated L2 and L3 caches (39.5 mm^2). Razor adds a 1-3% performance penalty to normal operation, while our approach has no performance overhead. Finally, and perhaps most importantly, we believe that the CPM approach has less impact on the overall chip design and verification effort. It requires localized circuit skill to model accurate representations of the critical path and the proportional-integral control loop filter in the variable clock source, but does not impact the design or verification of existing circuit paths. The remainder of the chip can be designed assuming functional correctness is maintained at all times and only those areas otherwise susceptible to soft errors require the overhead of error recovery. In contrast, Razor involves modifying existing circuit paths, which we expect will require a larger dedicated design team and/or have a larger impact on existing design and verification efforts.

3. ACTIVE GUARDBAND MANAGEMENT

Our architecture for guardband management requires three components: (1) a sensor to measure existing timing margin (Section 3.1), (2) a fast, hardware controller that protects against



A) CPM block diagram for 1 processor core.

B) Edge detector in CPM.

Figure 3: Critical path monitor.

reducing timing margin to unsafe levels (Section 3.2), and (3) a controller to adjust voltage to convert excess timing margin into energy savings (Section 3.3)

3.1 Measuring Timing Margin

We use Critical Path Monitors (CPM) in the POWER7 to measure available timing margin. These CPMs are refinements of the CPMs implemented in previous POWER chip technologies [5]. A block diagram of the CPM and its place in the system control are shown in Figure 3A. The CPM consists of a pulse generation circuit to generate the timing edge, a calibration delay used to offset process variation and to adjust timing margin as described later, a critical path synthesis block, edge detector, and output encoder (just a window function, bits 4 through 8 are passed through to the DPLL). Four timing paths are implemented in the POWER7 CPM synthesis block, but only three are used in determining core timing: an inverter delay path, a pass-gate delay path, and a wire delay path. There are 5 CPMs on each of the 8 cores in POWER7. The CPMs for each core are combined as shown in Figure 3A so that the CPM indicating the least timing margin will dominate.

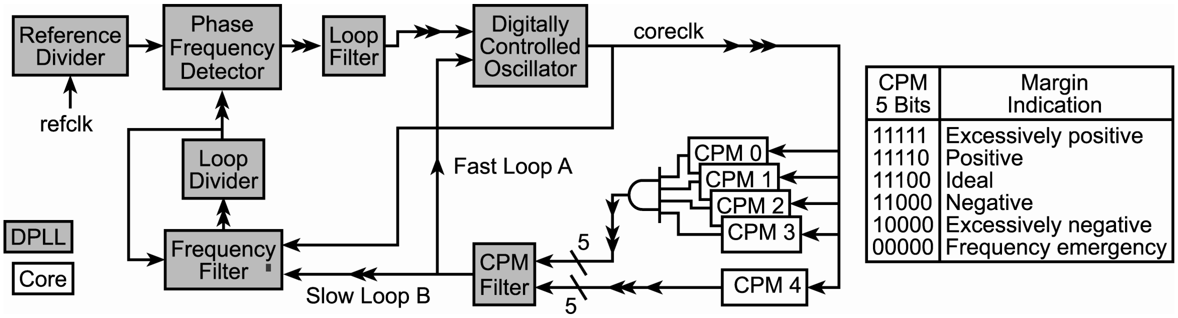
Each clock cycle, a rising edge is launched from the pulse generator into the delay paths. At the same time, the previously launched edge is captured in the 12-bit edge detector, shown in Figure 3B. The edge-detector output bits are numbered 0 through 11, where 0 is the first bit that can receive the timing edge. The edge detector output is a string of 1s followed by 0s with the location of the 1 to 0 transition indicating the timing edge as a function of the clock frequency. Ideally, the edge is located in the middle of the edge detector for maximum sensor visibility. If the voltage drops or the temperature increases, the circuit path feeding the edge detector slows, so the edge will not propagate as far into the edge detector and the edge moves toward bit 0. If the frequency increases, there is less time for the edge to propagate, so the edge also moves toward bit 0. Conversely, increased voltage, reduced temperature, and reduced frequency will cause the edge to move toward bit 11. The movement of the edge in the edge detector is a measure of the change in any noise process affecting timing (voltage, temperature, process corner, workload, clock jitter, skew, etc.). The CPM samples the timing each clock cycle, so it will detect noise events large enough to cause a timing shift on the next cycle boundary after the noise event begins to occur. By locating the CPMs in the power-dense regions of the microprocessor, they experience the operating point and process variation as the actual critical paths. As long as the CPMs respond in a similar fashion to noise processes as the real critical paths, they will approximate the critical path behavior and act as an effective timing monitor. Measurements of the POWER6 CPM described in [5] demonstrate that this CPM design tracks critical path well enough for effective clock control although some amount of margin will still need to be maintained to account for mis-matches.

During CPM design, each of the delay paths is adjusted to have the same pulse launch to edge-detector output delay (as indicated by an edge in bit 6 or an output of 111111000000) at the nominal voltage and frequency timing target. Bit position 6 has a special role since the timing margin controller (described in Section 3.2) relies on this bit to indicate the timing guardband setpoint. Timing guardband in the controller is set by the calibration delay: increasing calibration delay synthesizes a longer critical path which slows the DPLL down, adding timing margin. CPM calibration is the process of re-centering the CPM edge of each CPM to bit position 6 to compensate for process variation, including intra-core variability, while adding calibration delay to provide the desired guardband. Calibration is performed at nominal voltage with the guardband included in the frequency while running a heavy workload as described in Section 2.1. Due to quantization error in the delay line adjustment, the calibrated position may actually be bit 7 or 8. In practice, calibration would be performed once during manufacturing test using the processor sort workload. After calibration a reduction in voltage or an increase in temperature will cause the edge to move toward bit 0 and indicate a loss of timing margin to the DPLL. The opposite movement indicates an increase in timing margin.

3.2 Protecting Timing Margin

A DPLL [19, 20, 21] is present in each POWER7 core, providing per-core dynamic frequency scaling while the core continues to execute code. The DPLL is capable of a near-continuous set of adjustments between 50-125% of nominal frequency with a controlled slew rate and no skipped cycles. These features limit the power supply drop caused by dI/dt and prevent timing hazards during frequency changes.

The normal operating mode of the DPLL is to have the frequency controlled by a central power management controller. In order to improve the latency of frequency changes in response to operating condition changes, a secondary operating mode was added to the



CPM 5 Bits	Margin Indication
11111	Excessively positive
11110	Positive
11100	Ideal
11000	Negative
10000	Excessively negative
00000	Frequency emergency

Figure 4: DPLL block diagram operation using CPM feedback.

DPLL that uses the CPMs to directly control the DPLL frequency. The calibrated CPM sensors and DPLL work together to operate the microprocessor at a desired level of timing guardband by driving the output of the CPM edge detector towards the middle bit position 6. This causes the microprocessor to run at the fastest, safe frequency selected by the guardband characterization and the current operating environment.

Figure 4 is a block diagram showing this CPM/DPLL mode (the normal operating mode in which the DPLL frequency is controlled by the EnergyScale microcontroller is not shown). The DPLL receives a 5-bit window composed of bits 4 through 8 in the 12-bit CPM thermometer code described in Section 3.1. The 5 processor core CPMs are ANDed together so that the lowest timing margin measurement is presented to the DPLL. This code is a direct proxy measurement of the available timing margin in the core and it is interpreted as shown in Figure 4. When the edge in the edge detector is outside positions 4-8, the input to the DPLL is clipped to the extreme values shown in Figure 4.

When the DPLL detects the edge in the first 2 bits in the 5-bit window, it will slow down to increase the amount of time the edge has to propagate until the edge moves back to the middle bit position. Conversely, if voltage increases or temperature decreases, the delay line will be faster and the edge may move to the last two bits in the 5-bit window. In this case, there is extra timing margin so the DPLL speeds up to reduce the amount of time the edge has to propagate until the edge moves back to the middle bit position.

Inside the DPLL there are two feedback loops that use the CPM encoding. In the first loop, Loop A, the CPM value is filtered and immediately presented to the Digital Controlled Oscillator (DCO) where it adjusts the frequency up for positive margin and down for negative margin. The round-trip time of this loop is 8-10 processor cycles from DCO change, through the clock distribution, to the CPM, back to the PLL, and into the DCO with the new value. This fast loop reduces frequency up to 7% or increases it up to 5% in about five nanoseconds to respond to fast noise events. The second loop, Loop B, is a slower loop that uses the normal DPLL feedback loop. It detects that the CPM is requesting a frequency change and will slowly shift the frequency accordingly. This feedback loop is capable of 50 MHz/ms frequency changes. The proportionality and slew rates of both feedback loops can be adjusted to change the sensitivity of the loop to current timing margin changes to improve clock stability.

A similar approach (local noise detection actuating local frequency) to local clock control was used by Intel for the Montecito chip. In their design, they used regional voltage detectors to sense voltage changes and then adjusted the output divider of the local clock divider to compensate for the new operating condition [9]. They did not use the regional voltage detectors to adjust the PLL. While they report response times of 1.5 cycles to noise events, the extra stability we get using the proportional feedback of the DPLL is worth the slightly lower latency of our design.

3.3 Converting Excess Timing Margin to Energy Savings

Now that the timing margin controller holds the timing margin constant and protects against short-term noise, the frequency and voltage settings of the microprocessor may be adjusted to increase energy-savings. Left to its own devices, the timing margin controller will overclock the system to remove excess timing margin when the microprocessor is not running under worst-case conditions. To save energy, we add a performance controller that detects when the average clock frequency selected by the timing margin controller is above the level promised by the customer-set energy policy of the server. When this occurs, the performance controller reduces voltage to save energy.

The performance controller shown in Figure 1 takes a desired frequency target as input. On POWER7 systems, the user selects an energy management policy, which either sets a fixed frequency target or dynamically determines the frequency target based on system utilization [12]. The performance controller operates on a 32 ms interval. During each interval, it measures the average frequency output of the DPLL and compares this to the target frequency. If the measured frequency is higher than the target frequency, the controller steps the voltage down 6.25 mV (one step). The CPM senses this voltage reduction as a loss of timing margin which causes the DPLL to lower its output frequency toward the frequency target. Conversely, if the measured frequency is lower than the target frequency, the controller steps the voltage up 6.25 mV (one step). This results in additional timing margin which causes the DPLL to raise its output frequency toward the frequency target. Each POWER7 core has independent DPLLs. Since all cores share the same voltage in POWER7, the performance controller adjusts voltage so that each core runs at least as fast as its target frequency.

In practice, we observe that one core in a processor runs at the requested frequency while the other cores generally run at a higher frequency. This is because each core experiences a different voltage droop due to within-die leakage and workload variation. Since the goal is to save energy for a given performance target, we modified the performance controller to use a POWER7 hardware feature that sets a frequency cap on each DPLL to hold the maximum frequency output to no more than one frequency step (28 MHz) above the target frequency. This limits the timing margin controller from wasting energy on short time scales and allows the performance controller a 28 MHz window above the target frequency to sense opportunities for voltage reduction.

4. EXPERIMENTAL RESULTS

We implement the managed guardband architecture in a prototype IBM Power 750 Express Server (32 cores, 64GB). The timing margin controller is in the POWER7 microprocessor hardware. The performance controller is implemented in a prototype version of the EnergyScale firmware, which runs on an independent on-board microcontroller and is responsible for power management of the system [10]. During benchmark runs the firmware has a target frequency of 3864 MHz (the product's "Turbo" frequency), but frequency is only changed by the timing margin controller while voltage is adjusted by the performance controller. In all experiments the processor temperature is controlled by dynamically adjusting fan speed and the ambient temperature is 22º – 24º C.

The workload used for deriving CPM calibrations and CPM sensitivity experiments is called *HotTrash*. It is used for the traditional voltage selection of POWER7 chips. It is loaded directly into the POWER7 memory cache and runs without an operating system present. HotTrash induces a high power consumption on the processor, exercises all functional units, and checks the results. DAXPY, a floating-point intensive program, is used to validate guardband management under an operating system where HotTrash cannot run. We use the industry-standard SPEC CPU2006 benchmark suite to evaluate energy savings.

System power and component power are measured by the EnergyScale firmware using on-board power sensors. The power of a processor and its associated memory buffers is measured on a single power sensor and cannot be separated. Temperature is measured by digital thermal sensors on the POWER7 processor. There is a unique thermal sensor associated with each CPM.

4.1 Sensitivity of CPMs

We conducted experiments to understand the sensitivity of the CPMs to noise caused by temperature, voltage, and clock frequency. While running HotTrash, we varied either temperature, voltage, or frequency and observed the average CPM movement over 1024 sample readings. The nominal point around which the operating parameters are varied is 3864 MHz processor core clock frequency, the normal product turbo voltage setting, and 70° C average processor temperature. Measurements across different temperatures were accomplished by changing the fan speed. The voltage was varied by manually setting the POWER7 Vdd voltage regulator operating point. The frequency was varied by manually setting an internal POWER7 register that controls the DPLL. Our measurements show a movement of 1 bit position in CPM output corresponds on average to a change of 17 mV, 48 MHz, or 8.6° C. CPM resolution is dependent on operating voltage and is dependent on workload only so far as the workload causes large voltage excursions. In the POWER7 measurements, workload does not cause large enough excursions to impact the CPM resolution.

4.2 Calibration of CPMs

We calibrate and test our prototype server using two different CPM calibration settings that represent different timing guardband setpoints and are used in the following experiments.

The first calibration setting attempts to eliminate all guardband to determine an upper bound on the energy savings benefit. The calibration methodology is based on the traditional voltage selection procedure discussed in Section 2.1. We replicate the same environment by operating the processor at the voltage selected by the traditional method, overclocking the frequency by the frequency margin used during voltage selection, cooling the processor to 85° C, and running the HotTrash workload. The calibration delay is tuned until all the CPM edges line up as close as possible to bit position 6. The CPM reading varies slightly over time as the calibration procedure is running due to inherent and expected variations in the workload itself. Therefore we use the "sticky mode" of the CPM, which records the lowest edge position (closest to bit 0) since the last read operation, during calibration to capture the reading with the least amount of timing margin. The calibration procedure performs CPM read operations over a period of time (e.g. 250 ms) for each delay setting and then performs multiple samples at the final delay setting to ensure the tuning is correct.

We call this first calibration the *no guardband* calibration. The use of this setting has not resulted in a system failure despite running stressful workloads for days. When we replicate the above procedure using any higher frequency setting, the more aggressive calibrations will cause the processor to eventually fail. Therefore, we believe this tuning essentially eliminates all timing guardband.

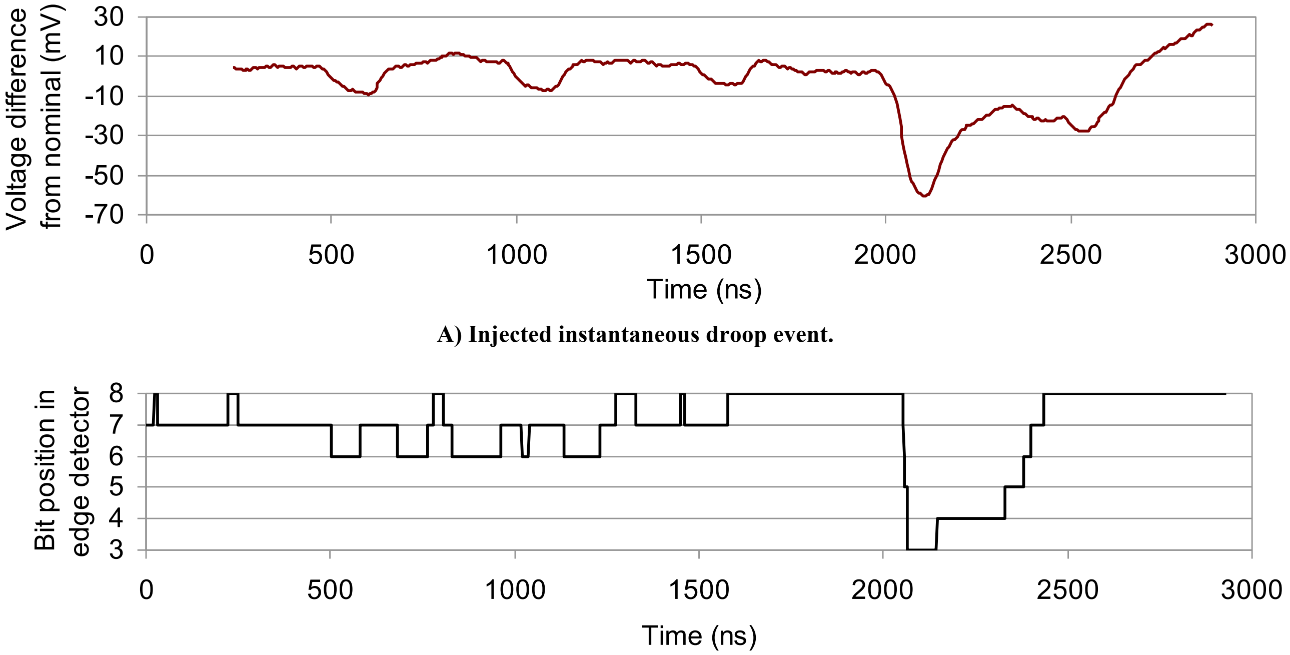
We developed a second calibration setting called *50% guardband* that we generate by removing only half of the frequency margin used in the traditional voltage selection. This represents a guardband with some additional timing margin over the *no guardband* calibration. While we have shown the *no guardband* calibration runs reliably in our prototype, the calibration point that should be used in a commercial product has additional considerations. A product must take into account test uncertainty, CPM accuracy, long-term aging effects, and the assumption that the worst-case noise workload is indeed the worst case – all perceived to be within acceptable business risk. Therefore, the *50% guardband* calibration may be a more acceptable target for commercial purposes.

Our baseline system does not use CPMs, but could be considered to be operating at a *100% guardband* calibration since it has 100% of the traditional voltage guardband.

4.3 Validation of Timing Margin Controller

We now validate that the timing margin controller avoids timing failures by testing systems both with and without the controller enabled. These tests focus on workload transients since they happen faster than the power supply voltage can be adjusted. We have observed workload-induced voltage droops in our system with a time period of about 50 ns. In contrast, system voltage regulators take at least a few microseconds to adjust voltage by even 1%. Fortunately, the timing margin controller provides a faster response. An unfiltered output from the CPM allows for very rapid reductions of the DPLL frequency of up to 7% in about five nanoseconds in response to voltage droops.

For the system under evaluation, a technique was developed that creates what we believe to be the most stressful functional change possible from a duration, magnitude, and rate-of-change perspective. The technique broadcasts an instruction pipeline throttle request simultaneously to all 8 cores on the chip. For the stressful workload being tested, the instructions per second (IPS) rate of execution dropped by a factor of 75x, essentially turning



A) Injected instantaneous droop event.

B) Measured CPM response with timing margin controller disabled. Different droop event from 5A.

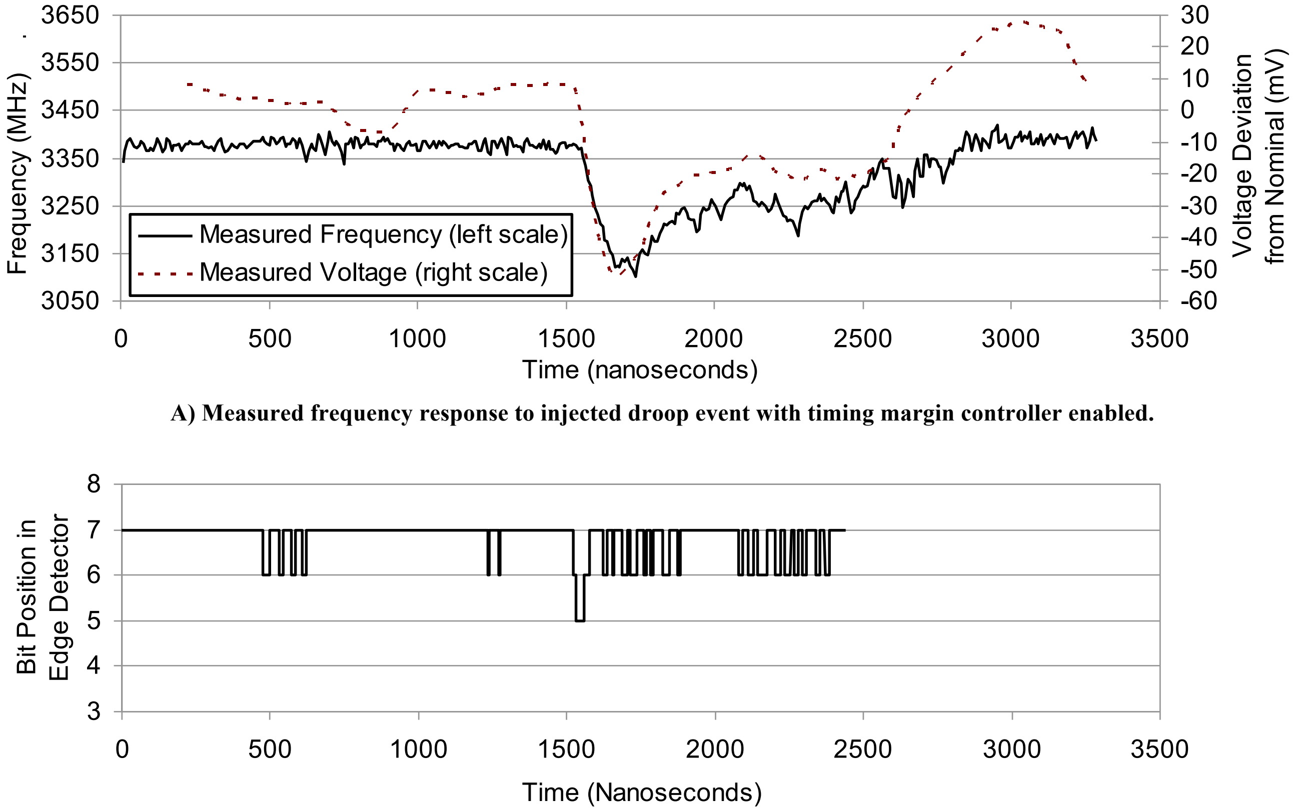
Figure 5: Response to worst-case noise event with timing margin controller DISABLED.

the most stressful workload into almost the lightest possible workload the chip could ever be executing. Later in time, the corresponding un-throttle command is broadcast which releases the stressful workload to its original IPS rate, or a 75x sudden increase. Note that the effect of these transitions from light to heavy is intensified by the inclusion of active clock gating and array power-down mechanisms in the hardware design. Releasing the throttle causes an instantaneous large change in current as the instruction pipeline refills and reengages the clocked-off and powered-down circuitry. The corresponding voltage droop event seen by the chip can occur on the order of 50 ns, which is more than an order of magnitude faster than any possible voltage change response.

Figure 5A shows an illustrative voltage droop event as captured by a scope on a hardware test bench setup using an internal chip voltage sense line wired directly out of the chip onto a debug connector. The droop event begins around 2000 ns into the trace. In this experiment we removed the VRM load-line to enhance the effect of the induced droop so that we could measure the response of the timing margin control system. We ran a steady state maximum power workload at nominal (frequency, voltage, and temperature) and calibrated the CPMs at these same conditions to maximize visibility. Figure 5B shows a cycle-by-cycle internal trace of a very similar, but different instance of, the induced voltage droop event as seen by the CPMs on one of the eight processor cores in this same chip when the timing margin control loop was disabled, also aligned to begin at 2000 ns. Since only the 5-bit thermometer code is available to the trace (and to the DPLL as mentioned previously in this paper), we are unable to resolve cycle-to-cycle readings above edge position 8 or below edge position 3 so the graph clips to those bounds.

Figure 6A illustrates the ability of the timing margin control loop to instantly adjust the DPLL frequency in response to the "un-throttle" voltage droop event. This figure combines simultaneous voltage and frequency traces taken on the same processor core as in Figure 5. This droop event occurs 1520 ns into the trace. During this experiment, the frequency cap of the DPLL was programmed to a nominal frequency of 3360 MHz due to our particular test bench setup. Figure 6B is a cycle-by-cycle internal trace of the CPM edge readings taken for the same droop event in Figure 6A. Dropping to bit 5, which is an excursion of only one additional bit into the edge detector, demonstrates how the mechanism is able to preserve almost all of the circuit guardband despite the magnitude of the voltage droop event. Also note that the edge position saturates at bit 7 when the voltage is above nominal since the frequency cap setting prevents the DPLL from overclocking based on CPM feedback, resulting in extra timing margin.

The data in Figure 6 show how the timing margin control loop responds to a single noise event. Figure 7 shows the results of a longer validation test run on a system with a traditional load line to demonstrate that the timing margin controller allows for significant undervolting of the system while protecting it from failure even while large operating point changes are occurring. Without the timing margin controller present, one can undervolt the system to such an extent that eventually there is a timing failure that causes the chip to malfunction. However, with the timing margin controller enabled, the undervolting amount can be taken well beyond the non-controlled system's minimum voltage while maintaining safe operation, albeit, at a reduced performance level. The validation test consists of the following conditions run with and without the timing margin controller enabled: set the DPLL frequency target to the turbo frequency, run a stressful steady-state workload, periodically throttle and un-throttle the workload to generate significant on-chip noise, and walk the voltage down to demonstrate the undervolting capabilities of the timing margin controller.



A) Measured frequency response to injected droop event with timing margin controller enabled.

B) CPM bit position with timing margin controller enabled. Same droop event as 6A.

Figure 6: Response to worst-case noise event with timing margin controller ENABLED.

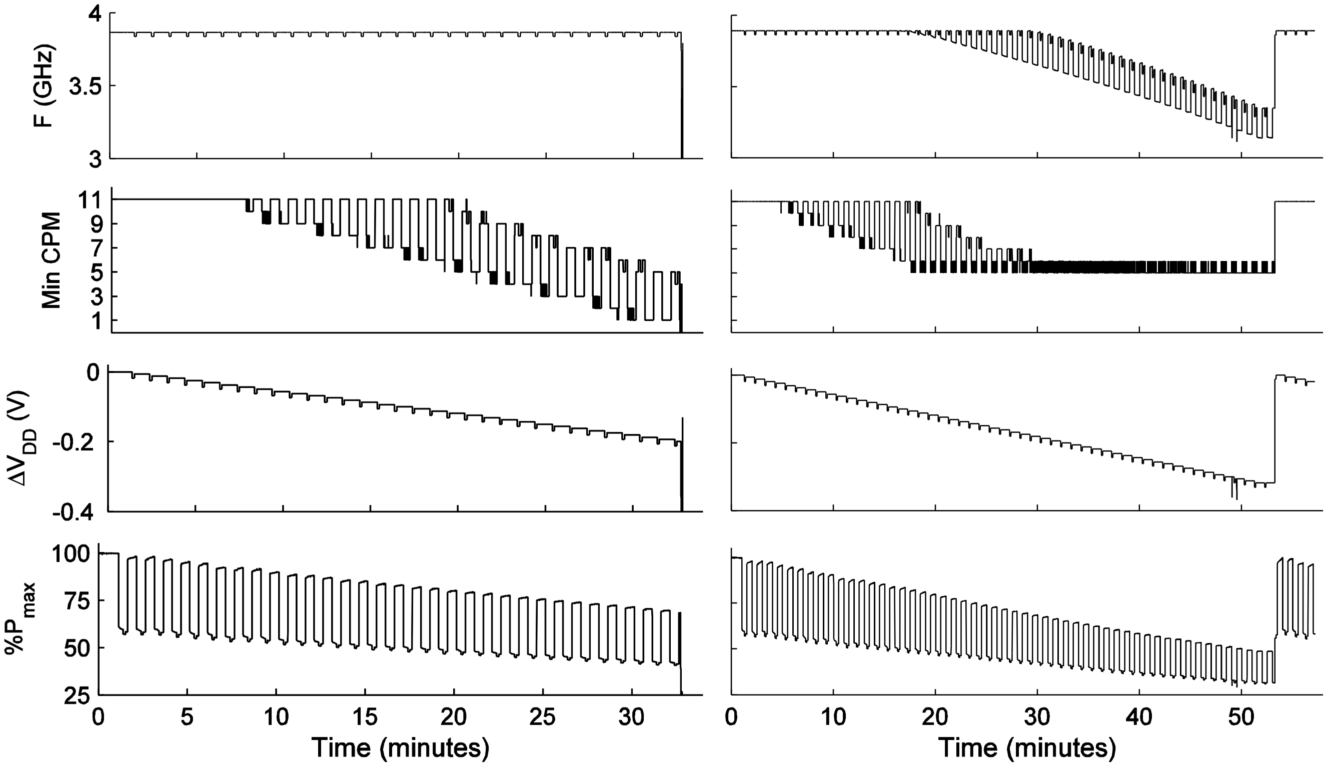
The stressful steady-state workload used for this test is *DAXPY*, which is a floating-point intensive workload. It is configured to hold all data within the L1 and L2 caches and no data is stored in system memory. This allows DAXPY to achieve over 2 floating point instructions per cycle on each core. It is run at a fixed turbo frequency and voltage, and the hottest cores operate at 70° C.

Once it is running, DAXPY is periodically throttled and un-throttled every 30 seconds so that some amount of thermal settling and adjustment can occur between events. When un-throttled at turbo frequency, turbo voltage, and 70° C core temperature, the CPMs read out at approximately bit position 11 using the *50% guardband* calibration. After throttling, and the 75x reduction in IPS throughput, the CPMs output is clipped at 11 due to the limited 12 bit range so the change in CPM value due to workload reduction is not visible at the turbo operating point. The cores cool off by 6° to 7° C when severely throttled but still at turbo voltages. If left for more than 30 seconds, the system's adaptive fan control algorithm will slow down the fans to conserve power and the cores will heat back up to 70° C. This effect is minimized by un-throttling after 30 seconds to heat the cores up again quickly before the fans slow down too much. The oscillation in the data in Figure 7 is a result of the periodic throttling of the workload.

Figure 7A plots the results of the test when the timing margin controller is disabled. The chip frequency, minimum CPM value (indicating worst case timing margin since the previous read of the CPM), voltage change, and percent power change are all plotted. A fixed turbo frequency is held while the voltage is walked down from the turbo voltage in 6.25 mV steps, with each drop occurring in the middle of the throttling of the chip (15 seconds into the 30 seconds of throttling). This gradual drop in voltage eventually causes a timing failure that is unrecoverable at the 33 minute mark when it is undervolted by 206.25 mV. At this point, due to the *50% guardband* calibration of the CPMs, the CPM bit position 0 is reached indicating there is essentially no timing margin left in the chip on the worst case paths.

Figure 7B plots the results of the test when the timing margin controller is enabled. The timing controller loop is calibrated at the turbo frequency with a *50% guardband* calibration of the CPMs. The frequency now shows the impact of the timing margin controller at work. From 0 to 20 minutes, the frequency is capped by the timing loop controller at the turbo frequency even though the CPM indicates extra margin is available. Near the 20 minute mark, when the undervolting has reached 125 mV, the frequency starts to drop below the turbo frequency as the timing margin controller holds the average minimum CPM bit position at 5.5 (bouncing back and forth between bits 5 and 6). The voltage continues to be reduced well below the failure point of 206.25 mV where the lack of a timing margin controller resulted in a chip timing failure. Once undervolting reaches 312.5 mV, it is reset back to 0 mV to repeat the test.

In conclusion, the timing margin controller is able to protect the processor from a timing failure due to dynamic voltage adjustments (undervolting) and rapid workload transients



A) No Timing Margin Controller.

B) Timing Margin Controller.

Figure 7: Voltage Reduction Test.

(throttling and un-throttling of the pipelines in all the cores) that happen much faster than voltage regulator can be adjusted. The results show that the chip will have a timing failure if there is no timing margin controller present during dynamic voltage adjustments and rapid workload transients. The impact of the timing margin controller is a frequency and performance reduction, but continued safe operation of the chip. Only by adding the additional outer-loop performance controller can voltage adjustments be used as a means to preserve performance for a given operating condition.

4.4 Saving Energy

We now demonstrate the significant energy savings that can be achieved by allowing the performance controller to dynamically set the voltage to obtain a target frequency at the calibrated timing guardband.

The baseline system (*100% guardband*) is the unmodified server using the normal turbo voltage and frequency, running the industry-standard SPEC CPU2006 benchmark suite with no feedback controllers enabled. The system power, component power, and workload performance are recorded for multiple runs of each workload in the suite.

For comparison, the feedback controllers are turned on and the workloads are run again using each of the calibration settings from Section 4.2. In these runs, the performance controller senses excess timing margin by observing that the DPLL is using a frequency higher than Turbo. It responds by reducing voltage as described earlier. For all workloads, the measured performance (not shown) was substantially identical to the baseline system and within norms for SPEC CPU2006 run-to-run variation. This is expected since the performance controller attempts to run the processors at the same Turbo clock frequency.

Power consumption results are shown in Figure 8. On average, the *50% guardband* calibration reduces processor and memory buffer power by 20% and overall system power by 18%. The peak processor power across all workloads was reduced by 13%. Individual processors undervolt the traditional voltage from 113 mV to 140 mV on average.

The *no guardband* calibration reduces power even further. Processor and memory buffer power is reduced 24% and overall system power is reduced 21%. The peak processor power measured across all workloads was reduced by 17%. Individual processors undervolt the traditional voltage by 137 mV to 152 mV on average.

There are two main reasons why *no guardband* and *50% guardband* produce somewhat similar power reductions. First, the CPM delay setting step sizes in POWER7 are very coarse-grained. While the names "no guardband" and "50% guardband" precisely describe the frequency margin used to derive the calibration, the final timing margin provided by the calibration is not as precise. Second, active processor power is proportional to the voltage squared, so the initial voltage reductions from using *50% guardband* cause most of the power reduction.

While most of the system power reduction comes from the processor power, a small portion comes from reduced fan power since the dynamic fan controller is able to maintain the normal processor 70° C operating point at reduced fan speed due to the processor power reduction. For both calibration settings, fan power was reduced by about 50%. The firmware has a built-in

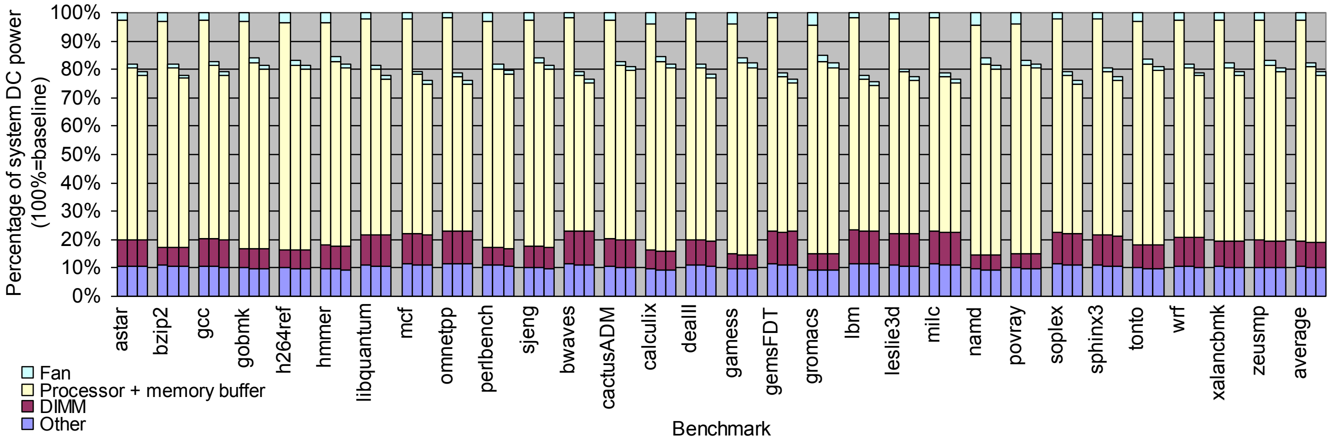


Figure 8: Average Power Consumption.
First bar is baseline system. Second bar uses timing margin control with 50% guardband calibration. Third bar is timing margin control with no guardband calibration.

minimum fan speed that prevented an even larger fan power reduction.

5. RELATED WORK

Our contribution is in the area of microarchitecture hardware and system firmware. The most closely related work, Razor, was discussed in detail in Section 2.3.

Voltage margin. Reddi et al. [18] study voltage margin in an Intel Core 2 Duo processor and suggest scheduling algorithms to reduce voltage droop events. They also present a good overview of voltage margin. Hardware-based and software-based methods that roll-back and recover after voltage margin is lost have been previously studied [1, 11]. We present an alternative solution for performance-sensitive systems that has a reduced implementation cost and does not cause timing failures resulting in potential performance loss from roll-back recovery. A trade-off is that our solution has reduced energy savings compared to roll-back-recover methods since it has a fixed timing guardband setpoint learned during chip manufacturing.

Timing margin. Ring oscillators have been extensively used to study process variation, measure timing margins, and control voltage [16, 15]. One drawback is that they can take on the order of a microsecond to many milliseconds to sense subtle variations in timing. By comparison, our work builds on critical path replicas [5] which provide timing margin measurements every clock cycle to speed the response of our hardware-based controller.

Additionally, our work builds on the large body of prior work on methods to manage dynamic power in the face of process variation, aging, and workload, including [4, 17, 1, 14, 22, 8].

6. CONCLUSION

In this paper, we present an active timing guardband management mechanism that employs critical path monitors (CPM) to continuously measure available timing guardband. The CPM outputs are fed to the DPLL clock generation circuit to form a hardware-based timing margin controller, which adjusts processor frequency within several cycles if the timing guardband is outside specified limits. We couple the timing margin controller with a firmware-based performance controller that monitors the average frequency achieved and adjusts the supply voltage, within safety limits, to achieve a long-term average frequency target.

To demonstrate that this approach can be implemented in a production-scale commercial server, with acceptable design and verification overhead, and to determine its effectiveness in saving energy while running realistic workloads, we implemented our solution in a prototype IBM POWER7 server. During typical conditions, the timing guardband management mechanism reduces voltage 137-152 mV below nominal and achieves an average processor power reduction of 24% without performance loss. Due to processor peak power reduction of 17%, provisioned bulk power supply and cooling could be reduced in capacity to save manufacturing cost. We believe this demonstrates that our solution is highly effective and commercially feasible.

7. ACKNOWLEDGMENTS

The authors thank Philip Restle, Alexander Rylyakov, Daniel Friedman, and Daniel Beece of the IBM Watson Research Lab who helped lay the groundwork for the CPM-clock feedback control loop and did extensive modeling to validate the CPM-DPLL feedback implemented in the POWER7 chip.

The authors thank Richard Willaman of the IBM Austin POWER Systems Lab for his assistance in collecting the instantaneous droop scope data.

8. REFERENCES

[1] Bowman, K.A.; Tschanz, J.W.; Kim, N.S.; Lee, J.C.; Wilkerson, C.B.; Lu, S.-L.L.; Karnik, T.; De, V.K. 2009. Energy-Efficient and Metastability-Immune Resilient Circuits for Dynamic Variation Tolerance. *IEEE Journal of Solid-State Circuits*. 44, 1 (Jan. 2009), 49-63. DOI=10.1109/JSSC.2008.2007148.

[2] Das, S.; Tokunaga, C.; Pant, S.; Ma, W.-H.; Kalaiselvan, S.; Lai, K.; Bull, D.M.; Blaauw, D.T. 2009. RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance. *IEEE Journal of Solid-State Circuits,* 44, 1 (Jan. 2009). IEEE, 32-48. DOI=10.1109/JSSC.2008.2007145.

[3] Das, S. 2009. *Razor: A Variability-Tolerant Design Methodology for Low-power and Robust Computing*. Doctoral Thesis. University of Michigan.

[4] Dighe, S.; Vangal, S.; Aseron, P.; Kumar, S.; Jacob, T.; Bowman, K.; Howard, J.; Tschanz, J.; Erraguntla, V.; Borkar, N.; De, V.; Borkar, S. 2010. Within-die variation-

aware dynamic-voltage-frequency scaling core mapping and thread hopping for an 80-core processor. *International Solid-State Circuits Conference Digest of Technical Papers* (Feb. 7-11, 2010). ISSCC 2010. IEEE, 174-175. DOI=10.1109/ISSCC.2010.5433997.

[5] Drake, A.; Senger, R.; Deogun, H.; Carpenter, G.; Ghiasi, S.; Nguyen, T.; James, N.; Floyd, M.; Pokala, V. 2007. A Distributed Critical-Path Timing Monitor for a 65nm High-Performance Microprocessor. *International Solid-State Circuits Conference Digest of Technical Papers* (Feb 11-15, 2007). ISSCC 2007. IEEE, 398-399. DOI=10.1109/ISSCC.2007.373462.

[6] Ernst, D.; Kim, N.S.; Das, S.; Pant, S.; Rao, R.; Pham, T.; Ziesler, C.; Blaauw, D.; Austin, T.; Flautner, K.; Mudge, T. 2003. Razor: a low-power pipeline based on circuit-level timing speculation. In *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture* (Dec 3-5, 2003). MICRO-36. 7-18. DOI=10.1109/MICRO.2003.1253179.

[7] Ernst, D.; Das, S.; Lee, S.; Blaauw, D.; Austin, T.; Mudge, T.; Kim, N.S.; Flautner, K. 2004. Razor: circuit-level correction of timing errors for low-power operation. *Micro*, 24, 6 (Nov.-Dec. 2004), IEEE, 10-20. DOI=10.1109/MM.2004.85.

[8] Fick, D.; Liu, N.; Foo, Z.; Fojtik, M.; Seo, J.; Sylvester, D.; Blaauw, D. 2010. In situ delay-slack monitor for high-performance processors using an all-digital self-calibrating 5ps resolution time-to-digital converter. *International Solid-State Circuits Conference Digest of Technical Papers* (Feb. 7-11, 2010). ISSCC 2010. IEEE, 188-189. DOI=10.1109/ISSCC.2010.5433996.

[9] Fischer, T.; Anderson, F.; Patella, B.; Naffziger, S. 2005. A 90nm variable-frequency clock system for a power-managed Itanium®-family processor. *International Solid-State Circuits Conference Digest of Technical Papers* (Feb. 2005). ISSCC 2005. IEEE. 294-299. DOI=10.1109/ISSCC.2005.1493985.

[10] Floyd, M.; Allen-Ware, M.; Rajamani, K.; Brock, B.; Lefurgy, C.; Drake, A.J.; Pesantez, L.; Gloekler, T.; Tierno, J.A.; Bose, P.; Buyuktosunoglu, A. 2011. Introducing the Adaptive Energy Management Features of the Power7 Chip. *IEEE Micro*, 31, 2, (March-April 2011), 60-75. DOI=10.1109/MM.2011.29.

[11] Gupta, M.S.; Rangan, K.K.; Smith, M.D.; Gu-Yeon Wei; Brooks, D. 2008. DeCoR: A Delayed Commit and Rollback mechanism for handling inductive noise in processors. In *IEEE 14th International Symposium on High Performance Computer Architecture* (Salt Lake City, UT, February 2008). HPCA 2008. IEEE, 381-392. DOI=10.1109/HPCA.2008.4658654.

[12] IBM. 2011. IBM POWER7 Technology and Systems. *IBM Journal of Research and Development*. 55, 3 (May-June 2011). DOI=10.1147/JRD.2011.2128750.

[13] Intel Corporation. 2009. *Voltage Regulator Module (VRM) and Enterprise Voltage Regulator-Down (EVRD) 11.1 Design Guidelines*. Reference Number 321736, Revision 002, September, 2009.

[14] Isci, C.; Contreras, G.; Martonosi, M. 2006. Live, Runtime Phase Monitoring and Prediction on Real Systems with Application to Dynamic Power Management. In *Proceedings of the 2006 39th Annual IEEE/ACM International Symposium on Microarchitecture* (MICRO 39). IEEE Computer Society, Washington, DC, USA, 359-370. DOI=10.1109/MICRO.2006.30.

[15] Kang, I.; Ethirajan, K.; Severson, M. 2005. *Dynamic Voltage Scaling for Portable Devices*, U.S. Patent Application 2005/0218871 A1.

[16] Keane, J.; Kim, C.H. 2001. An odometer for CPUs. *IEEE Spectrum*. 48, 5 (May 2011), 28-33. DOI=10.1109/MSPEC.2011.5753241.

[17] Meterelliyoz, M.; Goel, A.; Kulkarni, J.P.; Roy, K. 2010. Accurate characterization of random process variations using a robust low-voltage high-sensitivity sensor featuring replica-bias circuit. *International Solid-State Circuits Conference Digest of Technical Papers* (Feb. 7-11, 2010). ISSCC 2010. IEEE, 186-187. DOI=10.1109/ISSCC.2010.5433991.

[18] Reddi, V.; Kanev, S.; Kim, W.; Campanoni, S.; Smith, M.D.; Wei, G.-Y.; Brooks, D. 2010. Voltage Smoothing: Characterizing and Mitigating Voltage Noise in Production Processors via Software-Guided Thread Scheduling. In *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture* (MICRO 43). IEEE Computer Society, Washington, DC, USA, 77-88. DOI=10.1109/MICRO.2010.35.

[19] Rylyakov, A.; Tierno, J.; English, G.; Sperling, M.; Friedman, D. 2008. A wide tuning range (1 GHz-to-15 GHz) fractional-N all-digital PLL in 45nm SOI. In Proc. *2008 IEEE Custom Integrated Circuits Conference* (Sept. 21-24, 2008). CICC 2008. 431-434. DOI=10.1109/CICC.2008.4672113.

[20] Rylyakov, A.; Tierno, J.A. 2010. *Method and Apparatus for Low Latency Proportional Path in a Digitally Controlled System.* U.S. Patent Application 2010/0017690 A1. January 21, 2010.

[21] Tierno, J.A.; Rylyakov, A.V.; Friedman, D.J. 2008. A Wide Power Supply Range, Wide Tuning Range, All Static CMOS All Digital PLL in 65 nm SOI. *IEEE Journal of Solid-State Circuits*. 43, 1 (Jan. 2008), 42-51. DOI=10.1109/JSSC.2007.910966.

[22] Tiwari, A.; Torrellas, J. 2008. Facelift: Hiding and slowing down aging in multicores. In *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture* (MICRO 41). IEEE Computer Society, Washington, DC, USA, 129-140. DOI=10.1109/MICRO.2008.4771785.