System Level Analysis for Achieving Thermal Balance and Lifetime Reliability in Reliably Overclocked Systems

Prem Kumar Ramesh, Viswanathan Subramanian and Arun K. Somani
Dependable Computing and Networking Laboratory
Iowa State University
Ames, IA, USA
{pramesh, visu, arun}@iastate.edu

***Abstract*—Advancements in process technology offer continuous improvements in system performance. Technology scaling brings forth several new challenges. In particular, process, voltage, and temperature variations require sufficient safety margins to be added to the clock frequency of digital systems, making it overly conservative. Aggressive, but reliable, dynamic clock frequency tuning mechanisms that achieve higher system performance, by adapting the clock rates beyond worst-case limits, have been proposed earlier. Even though reliable overclocking guarantees functional correctness, it leads to higher power consumption and overheating. As a consequence, reliable overclocking without considering on-chip temperatures will bring down the lifetime reliability of the chip. In [1], we presented a comparative study on the thermal behavior of reliably overclocked systems with non-accelerated systems. In this paper, we elaborate more on the theoretical analysis along with experimental results to establish a safe acceleration zone for such 'better than worst-case' designs by efficiently balancing the gains of overclocking and the impact on system temperature. We analyze how reliable overclocking impacts the on-chip temperature of microprocessors, and evaluate the effects of overheating, due to reliable dynamic overclocking mechanisms, on the lifetime reliability of such systems. First, we theoretically study the possibilities for realizing such a system. We, then, evaluate the effects of thermal throttling, a technique that clamps the on-chip temperature below a predefined value, on system performance and reliability. Our study shows that a reliably overclocked system with dynamic thermal throttling, constrained to operating within 355K, achieves around 25% performance improvement.**

***Index Terms*—Microprocessors, Dependability, Adaptability, Overclocking, Thermal Throttling**

I. Introduction

In pursuit of ever faster execution times, a growing community known as overclockers, are manually accelerating their high performance processors past the manufacturer specified limits. Impressive results have been shown in existing systems that support overclocking. For example, a 2.6 GHz AMD Phenom processor running at speeds of up to 4 GHz using liquid cooling has been achieved. Such is the interest with overclocking enthusiasts that chipset manufacturers are introducing technologies that support overclocking. AMD's Overdrive and Advance Clock Calibration technologies are cases in point [2]. These gains are possible because of the worst-case assumptions used by traditional design methodologies. The clock frequency of a processor is selected to give enough time for the longest delay path, which determines the circuit propagation delay, to stabilize under adverse operating conditions. This propagation delay varies as process, voltage and temperature (PVT) variations are introduced during circuit fabrication and operation; so designers must assume the worst when fixing the system clock frequency. However, the combination of longest delay paths and adverse operating conditions are rare, leading to room for performance improvement that overclockers exploit. Systems running at overclocked speeds cannot be relied upon, as the possibility of system failure is very high. As a result, to account for the timing errors that occur at better-than-worst-case speeds, it is important to overclock the system reliably, in order to reap the benefits of making the common case faster.

The design for worst-case settings provides us an opportunity to improve processor performance to a greater extent through overclocking. When the system is forced to operate beyond this conservative limit, reliable overclocking mechanisms employ proven fault tolerance techniques to detect and recover from timing errors. Although aggressive clocking mechanisms facilitate in improving performance, they adversely impact on-chip temperatures, leading to hot spots. Overclocking enthusiasts invest heavily in expensive cooling solutions to protect the chip from overheating, and such overclocked systems typically have significantly lower lifetime. Additionally, reliable overclocking techniques necessitate additional circuitry, leading to an increase in power consumption. Higher clock speeds and power densities invariably lead to accretion of on-chip temperature over a period of time. As systems operate faster, on-chip temperatures quickly reach and exceed the safe limits. This poses a serious threat to the lifetime reliability of these systems. In [1], we presented our comparative study on the thermal behavior of reliably overclocked systems with non-accelerated systems. In this paper, we elaborate more on the theoretical analysis along with experimental results to establish a safe acceleration zone for such 'better than worst-case' designs by efficiently balancing the gains of overclocking and the impact on system temperature.

We must emphasize that current products from both the leading microprocessor vendors, Intel and AMD, have dynamic thermal monitoring techniques that take necessary corrective action to maintain on-chip temperature [3]–[5]. The corrective actions, in most cases, shut down the system or reduce system voltage and frequency, leading to considerable performance degradation. Our goal in this study is to analyze the temperature pattern of reliably overclocked systems, and

evaluate the lifetime reliability of such reliable aggressive clocking mechanisms. Furthermore, we monitor the on-chip temperature of aggressively overclocked systems that dynamically enhance single threaded application performance. We couple thermal monitoring techniques with reliable overclocking to alleviate lateral issues relating to system power and reliability. While taking feedback from an integrated thermal monitor, we observed an average performance increase of 25%, while operating within temperature 355K. Our work is related to Razor [6], which uses timing error tolerance mechanism to conserve energy while suffering moderate performance degradation. Another relevant study, SPRIT^{3}E [7], uses a similar mechanism to reliably overclock the system to enhance system performance.

First, we theoretically analyze the possibilities for realizing a controlled reliably overclocked system, while maintaining the on-chip temperature. We use SimpleScalar [8] simulator for Alpha EV6 processor, with a built-in power model, namely, Wattch [9] for the experiments. We integrate HotSpot [10] thermal modeling tool to monitor on-chip temperature. In real hardware, this translates to thermal sensors and counters for tracking on-chip temperature, which most of the present day chips support. We explore a broad spectrum of results for SPEC 2000 benchmark suite [11].

The remainder of this paper is organized as follows. Section II provides an overview of how reliable overclocking is performed in processors for performance enhancement. This section also outlines the issues related to thermal and reliability management in processors. We use processors and systems interchangeably in the rest of the paper. Section III explains our experimental framework used for analyzing the thermal impacts in reliably overclocked processors. We present our results in Section IV and Section V concludes the paper.

II. Background

A. Reliable Overclocking

One of the earliest works on aggressive clocking, TEATIME [12], scales the frequency of a pipeline using dynamic timing error avoidance. This technique attempts to achieve better-than-worst-case performance by realizing typical delay operation rather than assuming worst-case delays and operating conditions. TEATIME achieves this by modeling a one-bit wide delay chain that reflects the worst-case critical path of the system, plus a safety margin. A prior work to this called TIMERTOL [13] exists in which, timing error tolerance is achieved by multiple special copies of the pipeline logic. Similar architectures include CTV [14] and X-Pipe [15] that propose timing speculation at pipeline stage level.

The most significant aspect that can be exploited by reliable overclocking is the input data dependency of the worst-case delays. The worst-case delay paths are sensitized only for specific input combinations and data sequences [16]. Typically, the propagation delay of the digital system is much less than the worst-case delay and this can be exploited by overclocking. The benefits of overclocking can be furthered by allowing a tolerable number of errors to occur, and have an efficient mechanism to detect and recover from those errors. In addition to this, systems have different design restrictions, such as power, energy or area constraints. Based on all this, there are numerous architectures that have been proposed over the years.

Timing speculation based architectures that replicate registers in circuit critical path have been proposed. The basic idea is to duplicate latching, using shadow latches that are clocked in such a way to guarantee correctness. When a timing error is detected, it is recovered the following cycle. This technique along with dynamic voltage scaling has been used to improve energy efficiency [6]. Along with adaptive clocking mechanisms, reliable overclocking improves performance drastically [7].

In [17], the trade-off between reliability and performance is studied, and overclocking is used to improve the performance of register files. The conjoined pipeline architecture, proposed in [18], organizes pipeline redundancy in such a way as to improve both performance and reliability. In [19], triple modular redundancy based timing speculative register cells that can handle both soft errors and timing errors have been proposed.

Other works in the domain seek to improve common case performance through functionally incorrect designs [20], [21]. The Selective Series Duplex architecture [22] consists of an integrity checking architecture for superscalar processors that can achieve fault tolerance capability of a duplex system at much less cost than the traditional duplication approach. DIVA [21] uses spatial redundancy by providing a separate, slower pipeline processor alongside the fast processor. The desire for better than worst case designs is much more serious in nanoscale technology. PVT variations within and across the die are causing a bottleneck while selecting the worst-case frequency. ReCycle [23] uses additional registers and clock buffers to apply cycle time stealing from the faster pipeline stages to the slower ones. Another technique, EVAL [24] has been proposed to maximize performance with low power overhead in the presence of timing induced errors.

Apart from these run-time schemes, there are static methods that are specifically developed for better than worst case architectures. The effect of parameter variations and its impact on timing errors has been studied in [25]. BlueShift [26] proposes a design methodology from ground up. The main idea is to identify and optimize the frequently used critical paths, called the 'overshooters', at the expense of the lesser frequent ones.

In this section, we briefly discuss an in-built error detection and recovery mechanism that tolerates timing errors occurring at frequencies past the worst-case limit. We describe the working of *local timing error detection and recovery* circuits that replicate pipeline registers to support reliable overclocking. These circuits were first proposed in [6] to implement energy efficient processors and later used in [7] to enhance performance of superscalar processors. The purpose of these circuits is to detect and correct any resultant timing errors that occur because of overclocking, and to guarantee computational correctness.

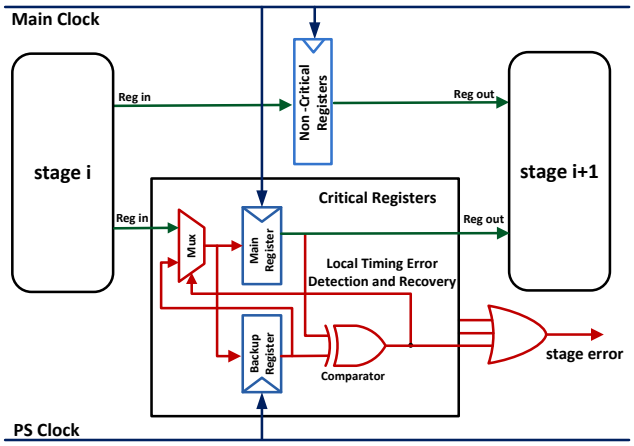

Fig. 1. Typical pipeline stage in a Reliably Overclocked Processor. Local timing error detection and recovery scheme for critical registers is shown in detail.

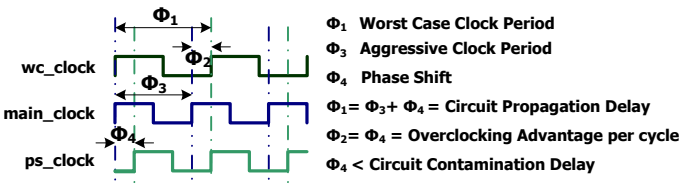

Fig. 2. Timing diagram showing overclocking advantage per cycle, as compared to the worst-case clock

B. Timing Error Detection and Recovery

In a reliably overclocked processor (ROP), to tolerate timing errors, registers in the critical paths of every pipeline stage are augmented with a second time-delayed register. A typical pipeline stage in such a processor, along with local timing error detection and recovery circuit augmentation for critical path registers, is shown in Figure 1. Each combinational logic stage is a dense logic combination with multiple inputs and outputs, and possibly with more than one path from each input to output. The short paths in the logic can operate correctly even during extreme voltage and/or frequency scaling. The paths that are not likely to meet their timing requirements are categorized as critical paths and only their corresponding stage output registers are replaced with timing error detection and recovery circuits.

A brief description of timing error detection and recovery in a ROP is presented from [7]: The main register is clocked ambitiously by the *Main Clock* at a frequency higher than that required for error-free operation. The backup register is clocked in such a way that it is prevented from being affected by timing errors, and its output is considered "golden" [7]. The clock for this register is phase shifted, shown as *PS Clock*, such that the combinational logic is effectively given its full, worst-case propagation delay time to execute. In case of a mismatch between the primary and backup registers, a recovery measure is taken by correcting the current stage data and stalling the pipeline for one cycle. In addition to local recovery, action is also taken on a global scale to maintain correct execution of the pipeline in the event of a timing error. The extent to which systems can be overclocked is limited by the penalty cycles needed to recover from timing errors. A balance must be maintained between the number of cycles lost to error recovery and the gains of overclocking. The achievable performance enhancement per cycle is shown in Figure 2 as Φ_2.

C. Timing Error Based Feedback Control System

Reliably overclocking a processor may not yield an increase in performance at all times. The reason being that the occurrence of a timing error is highly dependent on the workload and the current operating conditions. Therefore, it is beneficial to have an adaptive clock tuning system, which increases or decreases the clock frequency based on a set target error rate. In other words, it is necessary to fix a bound for overclocking, as errors induce additional recovery cycles that imparts a performance overhead.

- Let t_{no} denote the non-overclocked worst-case time period and t_{ov} denote the time period after overclocking.
- Let t_{diff} be the difference in time between the two time periods. Then, to execute n clock cycles, the total execution time is reduced by $t_{diff} \times n$, when there is no error.
- Let S_e, k and t_{pll} denote the fraction of clock cycles affected by errors, error recovery cycles and time taken by Phase Locked Loop (PLL) to lock next frequency respectively.

Then, equation 1 gives the bound on the timing errors that can be tolerated without adding any performance penalty.

$$S_e < \frac{t_{diff}}{t_{ov} \times k} - \frac{t_{pll}}{n \times t_{ov} \times k} \quad (1)$$

Dynamic clock frequency tuning is controlled by a global feedback system based on the total number of timing errors that occur in a specified time interval. Current products, such as IBM PowerPC 750GX processors, use two PLL scheme for clock generation to perform dynamic power-performance scaling [27]. This allows instant frequency switching, when frequency sampling interval is greater than t_{pll}.

The number of errors occurring at each timing error counter sampling interval is continuously monitored. As long as the number of errors is within target limits, the frequency is scaled up, else scaled down. One can apparently construe that the error rate is a monotonically increasing function with respect to frequency. This allows the use of efficient search algorithms to select the next tuned frequency starting from the base frequency.

For our understanding, let us consider the empirical model for circuit delay as given by Eqn (2).

$$Delay = \frac{C.V^2}{2v_{SAT}C_{OX}W(V - V_T)^2} \quad (2)$$

Here, v_{SAT}, C_{OX} and W are technology dependent constants; C specifies the capacitive load the circuit drives; V and V_T

are the system voltage and threshold voltage respectively ($V_T = 0.2398V$ for $45nm$ technology) [28]. Eqn (2) suggests that the time period provided should match this *Delay* in order to avoid timing errors. In traditional designs, the clock frequency is determined off-line during design phase for the worst-case settings, which is too conservative. However in our case, the clock time period is shorter compared to the circuit delay, and this results in timing errors. That is, there is a direct correspondence between the number of timing errors and the circuit slow down. Further, the slowdown can be related to the capacitive load that can be driven for that time period. Rearranging Eqn (2) for t_{no} and t_{ov} yield the following loads that can be driven respectively.

$$K_{no} = \frac{t_{no}(V-V_T)^2}{V^2},\ K_{ov} = \frac{t_{ov}(V-V_T)^2}{V^2}$$

Thus, the percentage slow down for the overclocked frequency with respect to the worst-case frequency is given by Eqn 3.

$$\%SlowDown = \frac{K_{no} - K_{ov}}{K_{no}} \times 100 \tag{3}$$

Finally, the maximum frequency for performance enhancement is theoretically limited by the contamination delay of the circuit. If time period of the new frequency is less than contamination delay of the circuit, timing error certainly occurs during every cycle and the error rate goes to 100%. Frequencies below propagation delay do not cause any timing errors at all (0% error rate). This, however, incurs performance overhead. Earlier studies have indicated that fixing a non-zero target error rate improves performance significantly. However, the system temperature goes up along with the system performance as the target error rate margin increases.

1) Speed-up: Having discussed the importance of timing error throttling, the next step is to re-calculate the speed-up achieved from overclocking. In a pipelined processor, the total number of cycles to process a given set of instructions is mainly divided into instruction execution, branch penalty and memory cycles. During overclocking, the clock frequency of the memory is not scaled, thereby increasing the total number of execution cycles. Let each memory operation take C_m cycles at t_{no} and q be the factor by which the frequency is scaled i.e., ($q = \frac{t_{no}}{t_{ov}}$). Now, after overclocking each memory operation takes $q.C_m$ cycles.

Let us assume that the system takes n clock cycles without considering memory cycles. If α denotes the factor of memory accesses that happen when the system executes n cycles. Then, the new execution time due to reliable overclocking is given by:

$$Ex_{ov} = n.t_{ov} + n.\alpha.q.C_m.t_{ov} + n.S_e.k.t_{ov} \tag{4}$$

To express original runtime (Ex_{no}) from Eqn 4, we replace t_{ov} by t_{no} and substitute $q = 1$ & $S_e = 0$. The overall speed up is calculated as given by Eqn 5.

$$Speedup = \frac{Ex_{no}}{Ex_{ov}} = \frac{q \times (1 + \alpha.1.C_m)}{(1 + \alpha.q.C_m + S_e.k)} \tag{5}$$

In our case, we take one cycle for timing error recovery, that is $k = 1$.

It should be noted that the benefits of reliable overclocking surpass the memory penalties provided the error rate is limited. This is clearly understood from the series of charts (a), (b) and (c) of Figure 3. Here, we depict the speed-up for a spectrum of memory access factors relative to target error rate, S_e, for different values of q.

For performance enhancement, the system must tolerate $20\%, 50\%, 70\%$ and 100% of timing exceptions at the overclocking rates $q = 1.2, 1.5, 1.7$ and 2.0, respectively. In the forthcoming sections, we show that for practical workloads the number of timing errors produced is quite low for smaller values of q, but quickly reaches 100% for higher values.

Also, from the model we deduce that non-memory bound workloads are more beneficial. For typical workloads, α is quite small, as most of the memory operations are shadowed through caching and buffering.

D. Thermal and Reliability Management

Over the last decade, thermal awareness has gained importance distinguishing itself from power awareness. Processor chips began to have thermal sensors in various locations to regularly sample the temperature and to shut down the operation in case of overheating. However, rapid heating and cooling of processor chips create thermal cycles affecting the lifetime reliability of the system [29].

The power consumed by a VLSI chip consists of two parts: dynamic and static. Dynamic power is dependent on capacitance (C), voltage (V), frequency (f) and switching factor (α), and is given by $P_{dyn} = \alpha CV^2 f$. Since dynamic power is directly proportional to the frequency at which the circuit operates, this causes overclocked systems to consume more power, which in turn causes systems to overheat. But solving the thermal problem is not as simple as bringing down the overall power consumed [30].

The problem becomes much more noticeable in designs under $90nm$ technology, where leakage power grows significantly. The leakage power grows exponentially with temperature as given by the empirical relationship, $P_{leak} \propto e^{\beta(T_i - T_0)}$ [31]. Here, β is technology dependent constant (β is 0.036 and 0.017 for $180nm$ and $70nm$ respectively), T_0 is the temperature of a reference point and T_i is the temperature at i^{th} instant with respect to the reference point. We see a positive feedback, wherein, increase in temperature leads to further leakage and increased total power consumption, which in turn leads to increase in temperature. Due to non-uniform switching and leakage, temperature is not distributed uniformly across the chip, creating localized heating in parts leading to hot spots.

Furthermore, overclocking increases the switching activity of the circuits causing more dynamic power dissipation. The dynamic power and energy consumed by the overclocked system are illustrated in Eqn (6) (7), respectively.

$$P_{ov} = \alpha.C.V_{ov}^2/t_{ov} = \alpha.C.V_{ov}^2/(t_{no}.q) \tag{6}$$

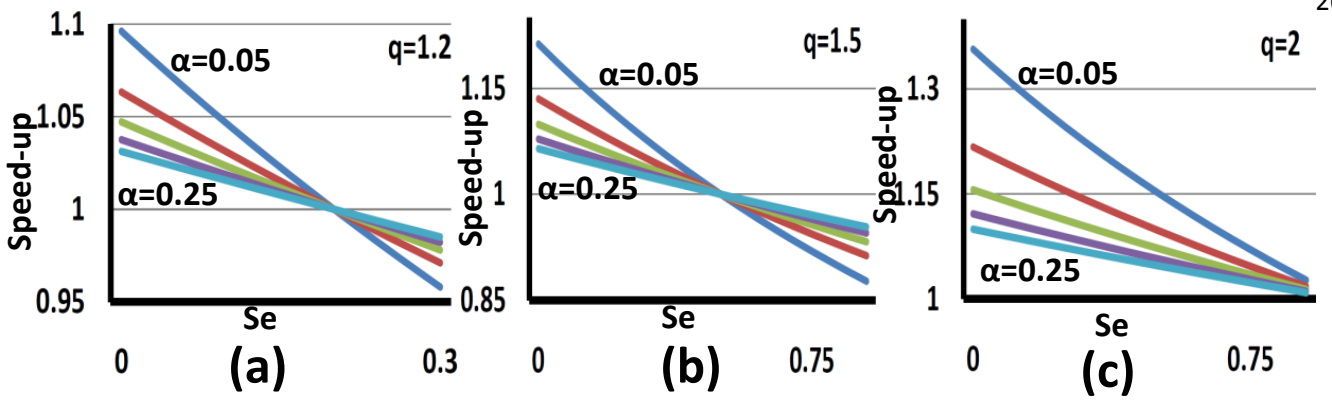

Fig. 3. Performance analysis of overclocking with error rate for different memory bounded workloads (a) Overclocking at 1.2X (b) Overclocking at 1.5X (c) Overclocking at 2X

$$E_{ov} = P_{ov}.n.(1 + \alpha.q.C_m + S_e.k).t_{ov} \quad (7)$$

Higher temperatures not only increase power budget, but also affect the lifetime reliability of the devices. To improve the overall reliability and lifetime of the systems, the thermal performance should be monitored and the average degradation of transistors managed. An initial exploration on thermal throttling through voltage reductions has been proposed in [32]. In this paper, we implement five critical failure mechanisms that are specified in [33] and [29] for our evaluation. A brief description of each of the wear out phenomenon and their respective Mean-Time-To-Failure (MTTF) are described below.

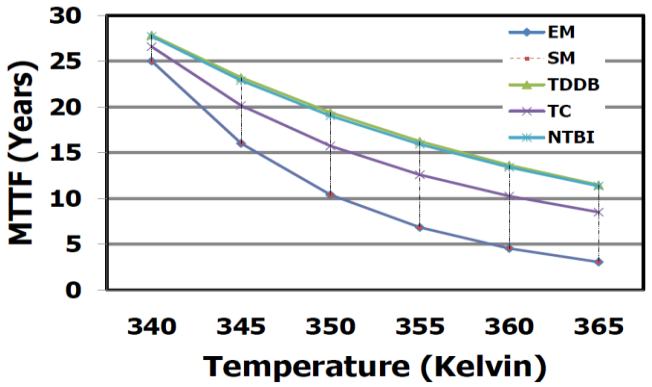

Fig. 4. MTTF for different steady state temperatures

Electromigration (EM) occurs due to transport of material due to gradual movement of the ions in a conductor caused by the momentum transfer between electrons and the diffusing metal. In Eqn (8), J is the interconnect current density. Activation energy, E_{aEM} and n are constants that depend on the interconnect metal used. (Typically, $n = 1.1$, $E_{aEM} = 0.9eV$).

$$MTTF_{EM} \propto (J)^{-n} e^{\frac{E_{aEM}}{kT}} \quad (8)$$

Stress Migration (SM) is a phenomenon that creates voids in the circuit, as a result of hydrostatic stress gradient. These voids may lead to high impedance or even break the circuit. This occurs due to difference in thermal expansion rates of materials. Again, E_{aSM}, m and the metal deposition temperature, T_{metal} are metal dependent constants in Eqn(9). T_{metal} generally assumes a value far higher than circuit operating temperature. (Typically, $m = 2.5$, $E_{aSM} = 0.9$). Although the term $|T_{metal} - T|^{-m}$ increases with T, there is an overall negative impact on the MTTF due to the exponential dependence of temperature.

$$MTTF_{SM} \propto |T_{metal} - T|^{-m} e^{\frac{E_{aSM}}{kT}} \quad (9)$$

Time Dependent Dielectric Breakdown (TDDB), also known as oxide breakdown occurs as a result of destruction of the gate oxide layer, and gradually leads to permanent transistor failure. a, b, X, Y and Z in Eqn (10) are fitting parameters. (Typically,$a = 78$, $b = -0.081$, $X = 0.759eV$, $Y = -66.8eV/K$, $Z = -8.37e - 4eV/K$).

$$MTTF_{TDDB} \propto (\frac{1}{V})^{(a-bT)} e^{\frac{[X+(Y/T)+ZT]}{kT}} \quad (10)$$

Sudden raise or fall in temperature causes **Thermal Cycles (TC)** which ultimately lead to device failure. Thermal cycles are caused by differences in thermal expansion rates across metal layers. Thermal cycling is proportional to the difference between current temperature and the ambient temperature $T_{ambient}$. In Eqn(11), $q = 2.35$, refers to the Coffin-Mason exponent, which is empirically determined material dependent constant. From this definition, one could observe that sudden cooling of devices below $T_{ambient}$ worsens the lifetime reliability.

$$MTTF_{TC} \propto (\frac{1}{T - T_{ambient}})^{q} \quad (11)$$

Finally, **Negative Bias Temperature Instability (NBTI)** is the failure mechanism that takes place in PFET devices.

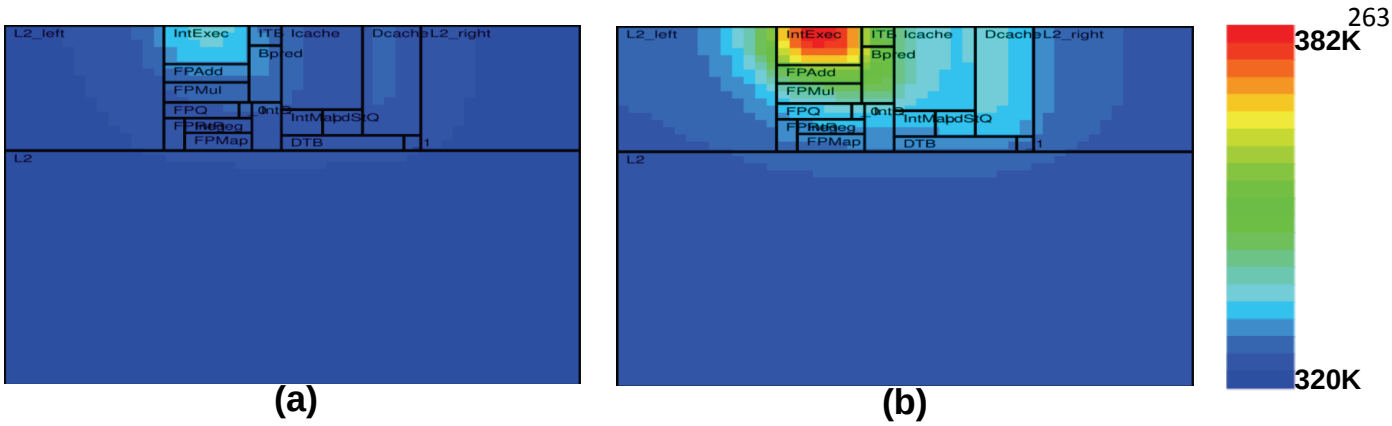

Fig. 5. Steady state temperature analysis (a) Non-overclocked processor settles around 330K (b) Reliably overclocked processor reaches over 380K

NBTI occurs due to timing constraint violations. In Eqn (12), A, B, C, D and β_1 are fitting parameters. (Typically, $A = 1.6328$, $B = 0.07377$, $C = 0.01$, $D = 0.06852$, $\beta_1 = 0.3$).

$$MTTF_{NBTI} \propto \left[\left\{ \begin{array}{c} ln\left(\frac{A}{1+2e^{B/kT}}\right) \\ -ln\left(\frac{A}{1+2e^{B/kT}} - C\right) \end{array} \right\} \times \frac{T}{e^{-D/kT}} \right]^{\beta_1} \quad (12)$$

Here, k is Boltzmann's constant and T is temperature in Kelvin. These wear out phenomena create impedance in the circuits, gradually leading to permanent device failures. Figure 4 shows how the increase in steady state temperature affects the processor lifetime. We use this reliability model to determine the critical temperature, for a target lifetime.

E. Thermal Consequences of Overclocking

Reliable dynamic clock frequency tuning for performance enhancement, described above, is incomplete without considering the thermal effects. Processors cannot be overclocked indefinitely, as this intensifies on-chip temperature. Thermal plots shown in Figure 5 compares a non-overclocked Alpha EV6 processor, running at $1GHz$ with an overclocked one, running at $2GHz$. We observed that steady state for dynamic reliable overclocking reached $380K$, while the non-overclocked settles at around $330K$. This calls the need for an efficient scheme for thermal balance in reliably overclocked processors.

III. Experimental Framework for Estimating On-chip Temperature

Figure 6 presents the entire simulation framework. The individual components are explained below in detail. The figure depicts both timing error based feedback control, and thermal throttle. For our initial evaluation of how on-chip temperatures vary when reliably overclocked, we only observe the temperature, without employing any thermal throttle. We employ dynamic clock tuning beyond worst-case limits, using timing error based feedback control, to adapt system behavior based on workload characteristics. The number of timing errors occurring at a given time is based on the workload being executed by the processor.

A. Modeling the ROP

To evaluate the trends in on-chip temperature, we model a reliably overclocked processor using a functional simulator, which incorporates a random timing error injector based on error profiles obtained by running application binaries on a hardware model. Our base processor, which is an out-of-order 64-bit, 4-way issue Alpha EV6 processor, is derived from the SimpleScalar-Alpha tool set [8]. This processor executes the Alpha AXP ISA. For our workload, we use pre-compiled set of Alpha binaries from the SPEC 2000 benchmark suite [11].

Wattch [9] is an accurate, architecture level power tool that is embedded within the SimpleScalar simulator. Wattch calculates instantaneous power at every cycle, and outputs the total power accumulated over a simulated period of time and the average power. We modified Wattch to track instantaneous power for each functional block.

B. Thermal Modeling

Thermal sensor modeling is done using the HotSpot tool [10]. The instantaneous power trace provided by Wattch power tool is used to calculate temperature. Since Wattch does not account for leakage power due to thermal runaway phenomenon, the temperature from HotSpot is used to calculate leakage power based on the formula presented in Section II. We obtain the $45nm$ Alpha EV6 floor plan from $130nm$ floor plan by assuming scaling is proportional to square of technology [34]. We also estimate the power dissipated by the additional circuitry required to detect timing errors.

C. Incorporating Timing Errors

In order to bring in the aspects of timing error in the SimpleScalar Alpha simulator, which is cycle accurate, but not timing accurate, we analyzed the number of timing errors occurring in the hardware model of a superscalar processor. We

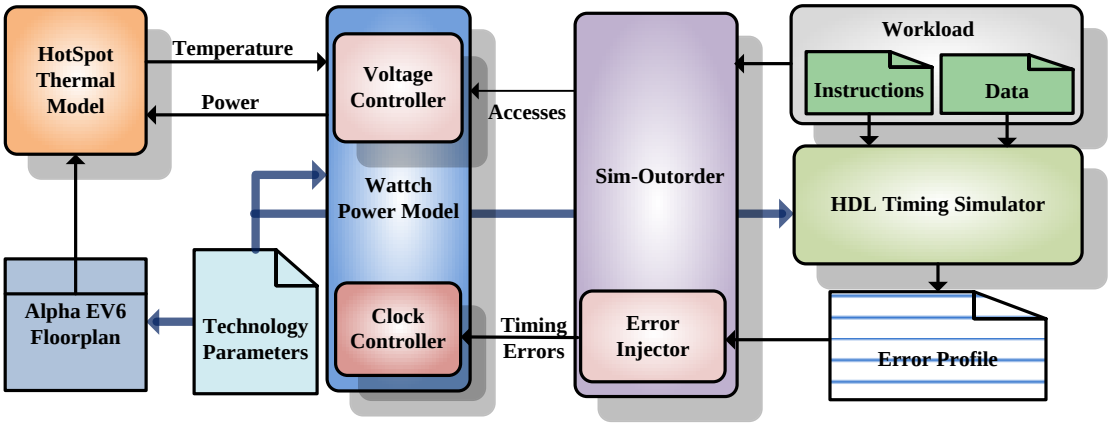

Fig. 6. Evaluation framework

Pipeline Stage	T_{PD} (ns)	T_{CD} (ns)	% Critical Registers	P_{leak} (mW)
Fetch	3.90	0.06	2.1	2.555
Decode	2.76	0.10	0	0.151
Rename	2.88	0.06	0	0.588
Issue	4.89	0.10	89.17	3.507
Execute	6.65	0.08	11.86	1.436
Memory	5.21	0.10	3.21	4.985
Commit	1.94	0.07	0	4.5

TABLE I
SYNTHESIS REPORT OF MAJOR PIPELINE STAGES

performed our study in a superscalar, dynamically scheduled pipeline similar in complexity to the Alpha 21264 [35]. We obtained the Illinois Verilog Model (IVM) from the University of Illinois website, which is a Verilog implementation of an Alpha microprocessor at the Register Transfer Level. The IVM Alpha processor executes a subset of the Alpha instruction set. However, the IVM processor is not fully synthesizable, and the synthesizable model does not support running SPEC 2000 benchmarks.

We designed the following experiment to evaluate the timing errors occurring at various overclocked frequencies. We synthesized individual pipeline stages using Synopsys design compiler. We used the $45nm$ OSU standard cell library for timing estimation [36]. There are altogether 12 pipeline stages in the processor. Table I reports the synthesis results for the major pipeline stages. In the IVM processor, the fetch stage, for instance, is divided into three stages. We report only the propagation delay, T_{PD} for the slowest among the three fetch stages. Similarly, we report the contamination delay, T_{CD}, for the shortest path across the three fetch stages. The timing values, reported in ns, are obtained from static timing analysis reports. However, the percentage of registers falling in the critical path and leakage power (P_{leak}) are reported for all three stages combined. In the table, we report the percentage of registers that have path terminating at them with delay values greater than or equal to $3.5ns$.

As explained in Section II-A, it is necessary to augment critical registers with error detection and recovery circuit, and also increase contamination delay of paths terminating at critical registers to a value greater than the desired extent of overclocking. Our simulator overclocks up to 40% of the worst case clock period. This requires the increase of contamination delay to over 40% the clock period. Using set minimum delay constraints in the Synopsys compiler, we increased the contamination delay to the required value. The overall increase in area for reliable overclocking was 3.5%.

Once we got the synthesized blocks for a particular stage, we replaced the RTL model for that block with the synthesized model. We also annotated timing information, extracted in standard delay format (SDF), on the blocks, so that we can run timing accurate simulations. We ran the instruction profile of various benchmarks obtained from the SimpleScalar simulator through the various stages. We used random data values for other inputs, filled the memory with random data. We measured error rate for various benchmark profiles.

Figure 7 shows the cumulative error rate for the IVM processor. We ran the experiment for 100000 cycles, and repeated the experiment with different sequences of 100,000 instructions for each benchmark. Average values are reported in the chart. We fixed the worst-case delay at $7ns$ to allow the maximum propagation delay of $6.5ns$ in the execute stage. We split 32 equal intervals from 7 to $3.5ns$ and measured error rate at each interval. We noticed around 89.17% of the paths fail in the issue stage at $3.5ns$, which causes a sudden rise in error rate, as observed in the Figure 7.

The error rate values we obtained from our hardware

simulation are incorporated in our functional simulator. While operating the simulator at higher frequencies, we use the error rates relatively. The leakage power, estimated at $105°C$ and $1V$ for IVM alpha processor at $45nm$, is used in Wattch power model to estimate leakage power at various temperatures.

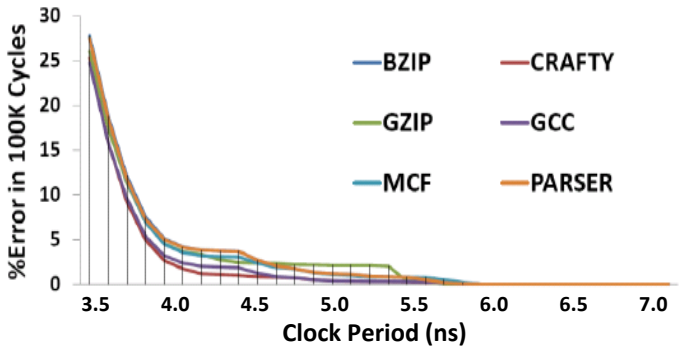

Fig. 7. Cumulative error rate at different clock periods for the IVM processor executing instructions from SPEC INT 2000 benchmarks

Parameter	Value
Fetch width	4 inst/cycle
Decode width	4 inst/cycle
Issue width	4 inst/cycle (ooo)
Commit width	4 inst/cycle
Functional units	4 INT ALUs 1 INT MUL/DIV 4 FP ALUs 1 FP MUL/DIV
L1 D-cache L1 I-cache L2 Unified	128K 512K 1024K
Technology node	$45nm$
Voltage	$1.25V$
Minimum frequency	$1024MHz$
Maximum frequency	$1862MHz$
No. of freq levels per voltage	32
Area	$10mm^2$
Temperature sampling	$1ms$
Freq sampling	$10\mu s$
Freq penalty	Single PLL: $10\mu s$ Dual PLL: $0\mu s$

TABLE II
SIMULATOR PARAMETERS

D. Area and Power Model for ROP

As explained in Section II-A, it is necessary to augment critical registers with error detection and recovery circuit, and also increase contamination delay of paths terminating at critical registers to a value greater than the desired extent of overclocking. Our simulator overclocks up to 45% of the worst-case clock period. This requires increasing contamination delay to over 45% the clock period. Using set minimum delay constraints in the Synopsys compiler, we increased the contamination delay to the required value. We adopt an overall increase in area of 3.5% for the additional circuitry from [7].

The power dissipation for the ROP was modeled based on our hardware implementation. We accounted for the additional area incurred for the local timing error detection and recovery circuits. For our model, we assume the increase in power dissipation to be directly proportional to the area increase we obtained. For each functional block in the processor the total power dissipation was increased by the fractional increase in overall area, obtained from our hardware experiments. The leakage power, estimated at $105°C$ and $1V$ for IVM alpha processor at $45nm$, is used in Wattch power model to estimate leakage power at various temperatures.

E. Simulation Parameters

Table II presents the simulation parameters. We evaluate the system temperature while running at $1.25V$. From Figure 2, we can see that clock period can be scaled only up to 50% of the original cycle time. We assume up to 45% overclocking. Table II provides the worst-case frequency and the maximum overclocked frequency we considered for our simulations. We perform a binary search between 32 frequency levels within the allowed range, based on error rate and also temperature, when employing thermal throttle. We assume the presence of two PLLs, so that there is no performance penalty involved, while switching between frequencies. If there is only one PLL, it takes up to $10\mu s$ to change from one frequency to another.

IV. On-chip temperature trends in reliably overclocked processors

We simulated six SPEC INT 2000 benchmarks to analyze the on-chip temperature trends in a reliably overclocked processor. The benchmarks reflect a broad spectrum of compute intensive workloads: gcc is a C compiler, $crafty$ is a chess program, mcf solves the minimum network flow problem, $parser$ involves natural language processing, $bzip2$ and $gzip$ are data compression utility applications.

We evaluate ROP performance with and without thermal throttling. Figures 8, 9, 10 and 11 compare the transient temperature trends and mean time to failure of a reliably overclocked processor with a non-overclocked processor for four of the benchmarks. Other two benchmarks have similar thermal characteristics. From the plots, we can clearly see that there is up to 15K difference between a reliably overclocked processor and a non-overclocked processor. Also, we see that the reliably overclocked processor reaches and exceeds 360K on executing around 3 million instructions. Based on the cooling solution used, the system will reach a steady state temperature and remain there. In our experiments, a non-overclocked processor settles at 347K for the same cooling solution. We start our experiments at a steady state temperature of 340K. This initial temperature is based on the assumption that the system has already performed certain operations, before it executes the benchmark of interest.

A. Frequency Based Thermal Throttling

When incorporating thermal throttle, we find that the temperature gets clamped at the desired choice of operating temperature. Since thermal sensor outputs are available once

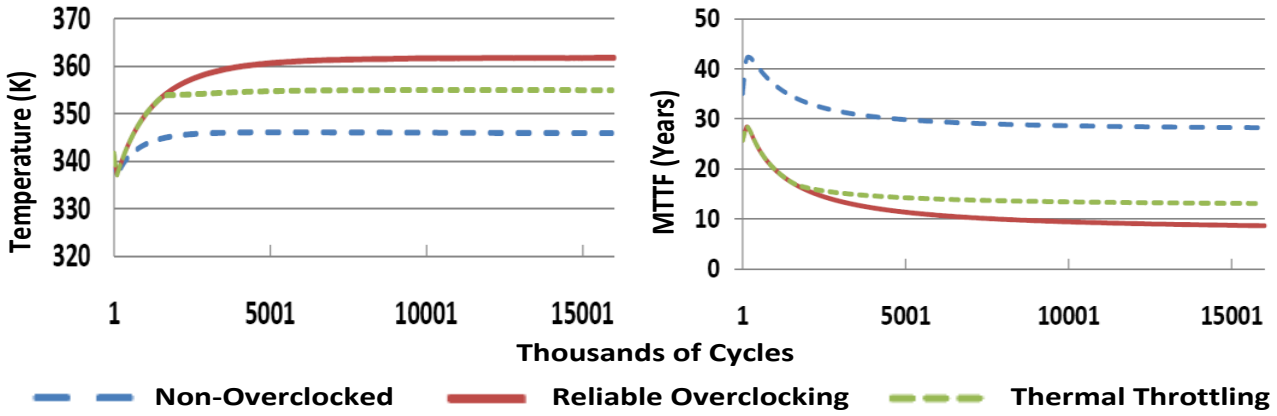

Fig. 8. $bzip2$ (a) On-chip Temperature Trend (b) MTTF Chart

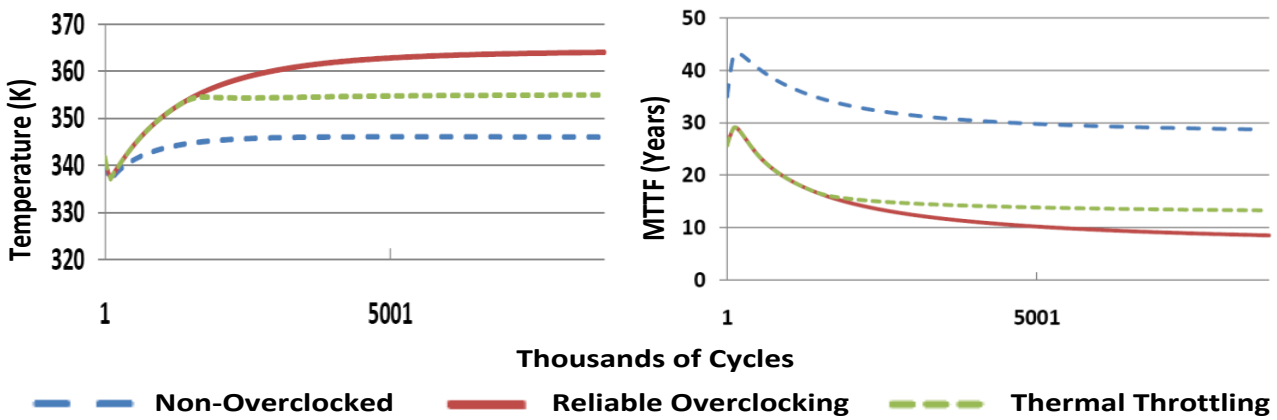

Fig. 9. $crafty$ (a) On-chip Temperature Trend (b) MTTF Chart

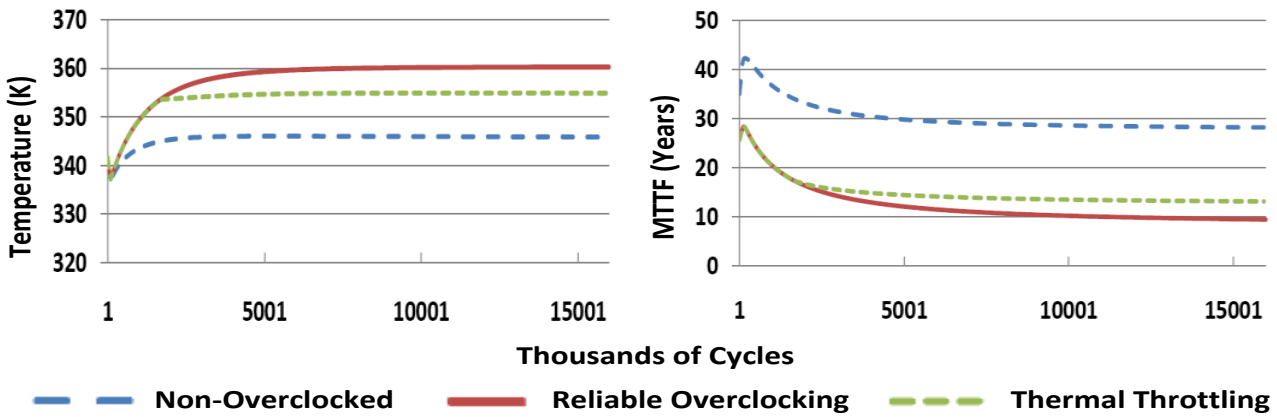

Fig. 10. $gzip$ (a) On-chip Temperature Trend (b) MTTF Chart

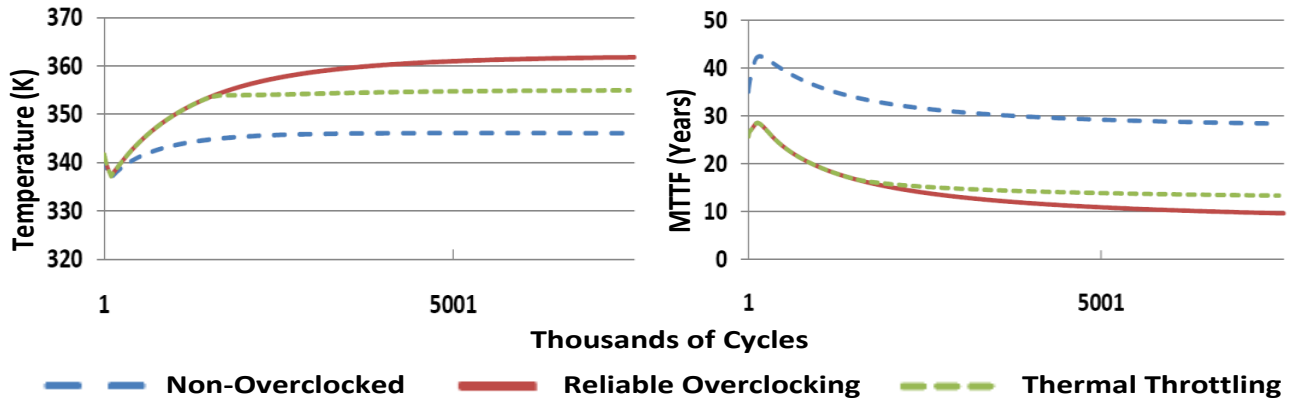

Fig. 11. mcf (a) On-chip Temperature Trend (b) MTTF Chart

every ms, it is good to choose a temperature 3K below the critical temperature, so that even if the system temperature overshoots before getting a thermal measurement, it will not exceed the critical temperature.

The relative speed-up for the six benchmarks, running 10^7 instructions is illustrated in Figure 12. Reliable overclocking, on an average, achieves 35% increase in performance over a non-overclocked system. When a thermal throttle is applied, the performance gain drops to 25%.

The MTTF values are obtained from the formulas mentioned in Section II-D. We calculate MTTF based on the on-chip temperature at that given instant of time. We obtain the proportionality constant for our calculations from the baseline MTTF at 337K [29]. We observe that a non-overclocked system has a longer lifetime, of about 30 years, as its on-chip temperature does not exceed 347K. However, a reliably overclocked system has a much shorter lifetime of about 9 years. Applying thermal throttling at about 355K increased the system lifetime to about 14 years. We understand from the figure that running a benchmark at lower temperatures over a long period of time improves the MTTF significantly. This motivates the need for having efficient dynamic thermal management techniques, alongside reliable overclocking, to achieve performance gain and reliability. Also, thermal management techniques alleviate the need for having an expensive cooling solution, making it cost effective to have high performance systems.

V. Conclusions

We have presented an initial study of the effects of reliably overclocked systems on on-chip temperatures. In addition, we also analyze the consequent effects on lifetime reliability of these systems. We considered a reliable overclocking framework and studied its thermal behavior compared to worst-case design. Our work in this paper is an initial exploration of dynamic thermal management in reliably overclocked systems. We are continuing this work by developing a powerful thermal management scheme that enhances performance as much as possible while operating well within the thermal limits, guaranteeing an extended system lifetime. The results we have

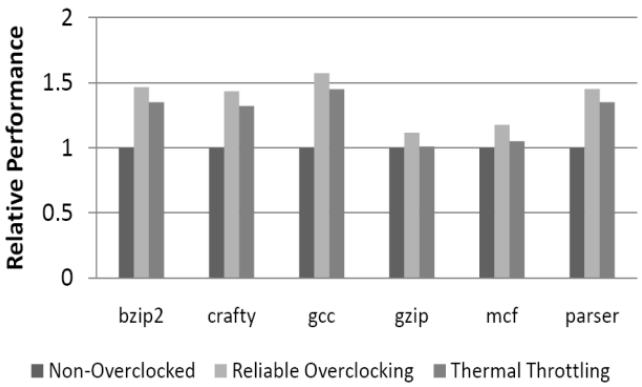

Fig. 12. Relative performance for SPEC INT 2000 benchmarks

obtained at this juncture are very promising, opening up many different directions for the near future.

We would like to further this work by implementing it on a hardware platform such as FPGA, by tracking temperature online through thermal sensors. We also plan to test our scheme with an ASIC model. As an extension to this work, we are planning to combine this technique with the existing dynamic voltage and frequency scaling (DVFS) and unify them within a common framework for better power saving. Finally, this work also opens up a new direction of managing power in chip multiprocessors, where our technique has the potential to allow fine grained and more accurate power management across the cores.

Acknowledgment

The research reported in this paper is partially supported by NSF grant number 0311061 and the Jerry R. Junkins Endowment at Iowa State University.

References

[1] V. Subramanian, P. K. Ramesh, and A. K. Somani, "Managing the Impact of On-Chip Temperature on the Lifetime Reliability of Reliably Overclocked Systems," in *Second International Conference on Dependability-DEPEND'09, June 18-23*, 2009.

[2] *http://www.crn.com/hardware/212101254*.

[3] R. McGowen, C. Poirier, C. Bostak, J. Ignowski, M. Millican, W. Parks, and S. Naffziger, "Power and temperature control on a 90-nm Itanium family processor," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 229–237, 2006.

[4] E. Rotem, A. Naveh, M. Moffie, and A. Mendelson, "Analysis of thermal monitor features of the intel pentium m processor," in *TACS Workshop at ISCA-31*, 2004.

[5] "AMD PowerNow! Technology," http://www.amd.com/epd/processors/6.32bitproc/8.amdk6fami/x24267/24267a.pdf, Date Accessed: March 31, 2009.

[6] D. Ernst, N. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, *et al.*, "Razor: A low-power pipeline based on circuit-level timing speculation," in *MICRO-36, 36th Annual IEEE/ACM International Symposium on Microarchitecture*, 2003, pp. 7–18.

[7] V. Subramanian, M. Bezdek, N. Avirneni, and A. Somani, "Superscalar processor performance enhancement through reliable dynamic clock frequency tuning," in *IEEE/IFIP International conference on Dependable Systems and Networks*, 2007, pp. 196–205.

[8] D. Burger and T. Austin, "The SimpleScalar tool set, version 2.0," *ACM SIGARCH Computer Architecture News*, vol. 25, no. 3, pp. 13–25, 1997.

[9] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *Proceedings of the 27th Annual International Symposium on Computer architecture*. ACM New York, NY, USA, 2000, pp. 83–94.

[10] W. Huang, K. Sankaranarayanan, R. Ribando, M. Stan, and K. Skadron, "An improved block-based thermal model in HotSpot 4.0 with granularity considerations," in *Proceedings of the Workshop on Duplicating, Deconstructing, and Debunking*, 2007.

[11] S. . B. Suite, http://www.spec.org/cpu2000/.

[12] A. Uht, "Uniprocessor performance enhancement through adaptive clock frequency control," *IEEE Transactions on Computers*, vol. 54, no. 2, pp. 132–140, 2005.

[13] A.K.Uht, "Achieving typical delays in synchronous systems via timing error toleration," in *Technical report 032000-0100*, Dept. of Electrical and Computer Eng., Univ. of Rhode Island, Kingston, 2000.

[14] T. Sato and I. Arita, "Constructive timing violation for improving energy efficiency," in *Compilers and operating systems for low power*, Kluwer Academic Publishers, 2003.

[15] X-Vera, O.Unsal, and A. Gonzalez, "X-pipe: An adaptive resilient microarchitecture for parameter variations," In Workshop on Architectural Support for GigascaleIntegration, 2006.

[16] T. Austin, V. Bertacco, D. Blaauw, and T. Mudge, "Opportunities and challenges for better than worst-case design," in *ASP-DAC '05: Proceedings of the 2005 Asia and South Pacific Design Automation Conference*. New York, NY, USA: ACM, 2005, pp. 2–7.

[17] G. Memik, M. Chowdhury, A. Mallik, and Y. Ismail, "Engineering over-clocking: reliability-performance trade-offs for high-performance register files," in *Dependable Systems and Networks, 2005. DSN 2005. Proceedings. International Conference on*, June-1 July 2005, pp. 770–779.

[18] V. Subramanian and A. K. Somani, "Conjoined Pipeline: A Fault-Tolerant High Performance Microarchitecture," in *Pacific Rim International Symposium on Dependable Computing, , Taipei, Taiwan, Dec*, 2008.

[19] N. D. Avirneni, V. Subramanian, and A. K. Somani, "Low Overhead Soft Error Mitigation Techniques for High-Performance and Aggressive Systems," in *IEEE/IFIP Dependable Systems and Networks*, 2009.

[20] F. M. J. Renau, "Effective optimistic checker tandem core design through architectural pruning," In International Symposium on Microarchitecture, 2007.

[21] T. M. Austin, "Diva: a reliable substrate for deep submicron microarchitecture design," in *MICRO 32: Proceedings of the 32nd annual ACM/IEEE international symposium on Microarchitecture*. Washington, DC, USA: IEEE Computer Society, 1999, pp. 196–207.

[22] S. Kim and A. K. Somani, "Ssd: An affordable fault tolerant architecture for superscalar processors," in *Pacific Rim Dependable Computing Conference*, December 2001, pp. 27–34.

[23] A. Tiwari, S. R. Sarangi, and J. Torrellas, "Recycle: Pipeline adaptation to tolerate process variation," 34th International Symposium on Computer Architecture (ISCA), 2007.

[24] S. Sarangi, B. Greskamp, A. Tiwari, and J. Torrellas, "Eval: Utilizing processors with variation-induced timing errors," In International Symposium on Microarchitecture, 2008, pp. 423–434.

[25] S. Sarangi, B. Greskamp, and J. Torrellas, "A model for timing errors in processors with parameter variation," in *Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on*, March 2007, pp. 647–654.

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

[27] *PowerPC 750GX Dynamic Power-Performance Scaling*. Version 1.0, January 29, 2009: IBM Systems and Technology Group. [Online]. Available: http://www.ibm.com

[28] J. H. et al, *A Robust Physical and Predictive Model for Deep-Submicrometer MOS Circuit Simulation*. Proceedings of the IEEE Custom Integrated Circuits Conference, pp.14.2.1-4, May 1993.

[29] J. Srinivasan, S. Adve, P. Bose, and J. Rivers, "Lifetime reliability: Toward an architectural solution," *IEEE Micro*, vol. 25, no. 3, pp. 70–80, 2005.

[30] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proceedings of the International Symposium on ComputerArchitecture*, 2003, pp. 2–13.

[31] S. Heo, K. Barr, and K. Asanovic, "Reducing power density through activity migration," in *ISLPED'03, Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, 2003, pp. 217–222.

[32] P. K. Ramesh, V. Subramanian, and A. K. Somani, "Thermal Management in Reliably Overclocked Systems," in *IEEE Workshop on Silicon Errors in Logic - System Effects, Stanford, CA, Mar*, 2009.

[33] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, *The Case for Lifetime Reliability-Aware Microprocessors*. The 31st International Symposium on Computer Architecture (ISCA-04), June 2004.

[34] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital integrated circuits*. Prentice Hall Englewood Cliffs, New Jersey, 2002.

[35] N. Wang, J. Quek, T. Rafacz, and S. Patel, "Characterizing the effects of transient faults on a high-performance processor pipeline," in *International Conference on Dependable Systems and Networks*, 2004, pp. 61–70.

[36] J. Stine, I. Castellanos, M. Wood, J. Henson, F. Love, W. Davis, P. Franzon, M. Bucher, S. Basavarajaiah, J. Oh, *et al.*, "FreePDK: An Open-Source Variation-Aware Design Kit," in *Proceedings of the 2007 IEEE International Conference on Microelectronic Systems Education*. IEEE Computer Society Washington, DC, USA, 2007, pp. 173–174.