

Online Timing Analysis for Wearout Detection

Jason A. Blome, Shuguang Feng, Shantanu Gupta, Scott Mahlke

Advanced Computer Architecture Laboratory
University of Michigan - Ann Arbor, MI
{jblome, shoe, shangupt, mahlke}@umich.edu

ABSTRACT

CMOS feature size scaling has long been the source of dramatic performance gains. However, because voltage levels have not scaled in step, feature size scaling has come at the cost of increased operating temperatures and current densities. Further, since most common wearout mechanisms are highly dependent upon both temperature and current density, reliability issues, and in particular microprocessor lifetime, have come into question. In this work, we explore the effects of wearout upon a fully synthesized, placed and routed implementation of an embedded microprocessor core and present a generic wearout detection unit. Since most common failure mechanisms may be characterized by a period of increased latency through ailing transistors and interconnects before breakdown, this wearout detection unit serves to predict imminent failure by conducting online timing analysis. In addition to measuring signal propagation latency, it also includes a unique two-level sampling unit which is used to smooth out timing anomalies that may be caused by phenomenon such as temperature spikes, electrical noise, and clock jitter.

1. INTRODUCTION

Future microprocessors will be composed of billions of transistors, many of which will be dead-on-arrival at manufacture time, and many more of which are likely to degrade in performance or fail over the expected lifetime of the processor [11]. Traditionally, microprocessors have been designed with the worst case operating conditions in mind, and manufacturers have employed guard bands and speed binning to ensure that processors will meet a predefined lifetime qualification, or mean time to failure (MTTF). However, shrinking feature sizes magnify the effects of process parameter variation, making process variation increasingly difficult to mitigate simply through provisioning for the worst case. Further, as CMOS feature size scales to smaller dimensions, voltage is expected to scale at a much slower rate, increasing on chip power densities. Areas of high power density increase local temperatures leading to “hot spots” on the die [20]. Since many important wearout mechanisms, such as electromigration (EM), time dependent dielectric breakdown (TDDB), negative bias temperature instability (NBTI), and hot carrier injection (HCI) are highly dependent on temperature, power density, or a combination of the two, the occurrence of wearout-related phenomenon will become increasingly common in future technology generations. These trends in VLSI design imply that the traditional approach of designing for the worst case in order to meet reliability goals will soon become infeasible, and that

the problem of meeting reliability goals must be addressed dynamically at the architectural-level throughout the lifetime of the processor.

High-end server systems designed with reliability as a first-order design constraint have typically relied on coarse grained replication in order to provide a high degree of reliability [9, 22]. However, dual and triple modular redundant systems are problematic for a number of reasons. First, they incur a significant overhead in terms of area and power. Second, without additional cold-spare devices, these techniques cannot significantly extend the lifetime of the processor because the redundant devices are constantly in operation and will all experience similar wearout effects. While the addition of cold-spare structures on the chip will help to prolong the processor lifetime, it requires additional on-chip area, beyond the redundant structures that are necessary to perform failure detection and diagnosis.

Another traditional approach to addressing wearout related failures is the use of canary circuits which fail faster than the circuits that they are meant to monitor, and thereby indicate that important processor structures may also fail soon. Canary circuits are an efficient and generic means to predict failure, however, there are a number of sensitive issues that must be addressed to effectively deploy canary circuits. First, the placement of these circuits is extremely important, because in order for the canary circuit to provide an accurate prediction, it must be subjected to the same environmental conditions as the circuit being monitored. At the same time, it is important that these circuits are not placed such that they disturb the actual structures that they are meant to protect by actually increasing the local power density or temperature.

Recent work by Srinivasan, et al. [25] proposes a dynamic monitoring system for quantifying the MTTF for different structures within a microprocessor as a function of the observed temperatures and utilization patterns. This system can then be used to swap in cold spares based on an online calculation of the expected time to failure for a given structure. While this technique may dynamically identify structures that are likely to fail in the near future, it requires extremely accurate device wearout models in order to effectively make a diagnosis. If the models are too conservative, structures will be retired before their time, and if the models are too lenient, failures in the field may potentially go undetected.

Rather than aggressively deploying duplicate fault-checking structures or attempting to predict failure with the aid of complex analytical wearout models, we propose an early warning system which leverages the *symptoms* of wearout in order to predict that failure is imminent. Wearout caused by phe-

nomenon, such as TDDB, EM, NBTI, and HCI each express themselves as an increasing signal propagation latency prior to inducing device failure. In this work, we propose a generic online latency monitoring and sampling circuit for detecting the onset of wearout within a microprocessor core. In order to justify and validate our failure prediction circuit design, we conduct accelerated wearout experiments on a fully synthesized, placed and routed Verilog implementation of an embedded microprocessor core.

The remainder of this work is organized as follows, Section 2 provides a detailed discussion of existing wearout models and the impact of wearout on VLSI designs. In Section 3 we introduce the experimental setup and the methodology used to conduct accelerated wearout experiment on a microprocessor core. Results from these experiments are presented in Section 3.3. Analysis of these results forms the basis for the design of our wearout detection unit (WDU) presented in Section 4. Next, we evaluate the design of WDU in Section 5 and illustrate the gains in processor lifetime achievable by selectively deploying WDU units and cold spares across the chip. Section 6 discusses related work, and is followed by the conclusion.

2. BACKGROUND

While feature size scaling continues to yield considerable gains in performance and area, in future generations these gains will come at the cost of reliability. Given the increasing susceptibility to wearout, reliability budgeting can no longer be simply an afterthought and must join the ranks of power and area as a first order design constraint.

With each new technology generation, non-ideal voltage scaling has led to increases in both current and power densities each contributing to rapidly rising temperatures. Research characterizing various failure mechanisms has shown that many of these mechanisms are heavily dependent on these factors [4]. In fact, the most extensively studied causes of wearout, TDDB and EM, experience exponential acceleration due to non-ideal scaling [15] [13] [24]. Though the remainder of this paper focuses on the study of the wearout effects of these two mechanisms, phenomenon such as negative-bias temperature inversion, hot carrier injection, and many others have been shown to be similarly dependent on temperature and current density. The following sections provide a survey of the relevant research and detail the models used for our studies.

2.1 Electromigration

Electromigration is a well-known physical phenomena that causes the mass transport of metal within semiconductor interconnects. As electrons flow through the interconnect, momentum is exchanged as the moving electrons collide with the metal ions. This “electron wind” tends to push the ions in the direction of electron flow and at high current densities results in the formation of voids (regions of metal depletion) and hillocks (regions of metal deposition) in the conductor metal [13]. In 1967, Black published work on electromigration in aluminum interconnects [10], describing the basic mechanism and developing a quantitative model.

The model of electromigration that we employ in this work

is based on a version of Black’s equation found in [4] and is consistent with recent literature [23] [19].

$$\begin{aligned} MTTF_{EM} &\propto (J - J_{crit})^{-n} e^{(E_a/kT)} \\ &\propto J^{-n} e^{(E_a/kT)} \\ &\propto (V_{dd} \cdot f \cdot p)^{-n} e^{(E_a/kT)} \end{aligned} \quad (1)$$

where

- J = current density ($J \gg J_{crit}$)
- J_{crit} , threshold current density at which EM begins
- $n = 1.1$, material dependent constant
- $E_a = 0.9eV$ (activation energy)
- k = Boltzmann’s constant
- T = temperature

Furthermore, current density J is modeled as in [14] with

$$\begin{aligned} J &= \frac{CV_{dd}}{WH} \cdot f \cdot p \\ &\propto V_{dd} \cdot f \cdot p \end{aligned} \quad (2)$$

where the line capacitance and geometries (C, W, H) are folded into the proportionality constant.

Studies have shown that the progression of EM over time can be separated into two distinct phases. During the first phase, sometimes referred to as the incubation period, interconnect characteristics remains relatively unchanged as void formations slowly increase in size. Once a critical void size is achieved, the second phase, the catastrophic failure phase is entered, characterized by a sharp linear increase in interconnect resistance [17] [13].

2.2 Time Dependent Dielectric Breakdown

Since the late 1960’s and early 1970’s, researchers have recognized the susceptibility of the silicon oxide insulator to radiation and high voltage damage. Time-dependent dielectric breakdown, also known as gate oxide breakdown, is caused by the formation of a conductive path through the gate oxide. TDDB exhibits two distinct failure modes, namely soft and hard breakdown [15] [7] [21]. The widely accepted “Klein/Solomon” model [21] characterizes oxide wearout as a multistage event with a prolonged wearout period (“trap generation”) during which charge traps are formed within the oxide. This is followed by a partial discharge event (“soft breakdown”) triggered by locally high current densities due to the accumulation of charge traps. Typically in thinner oxides, a series of multiple soft breakdowns eventually leads to a catastrophic thermal breakdown of the dielectric (“hard breakdown”).

The rate of failure due to TDDB is dependent on many factors, the most significant being oxide thickness (process dependent), operating voltage, and temperature. The work presented in this paper uses the empirical model (Equation 3) described by Srinivasan [23] which is based on experimental data collected at IBM [26].

$$MTTF_{TDDB} \propto \left(\frac{1}{V}\right)^{(a-bT)} e^{\frac{(X+Y+ZT)}{kT}} \quad (3)$$

where

- V = operating voltage
- T = temperature
- k = Boltzmann's constant
- $a, b, X, Y,$ and Z are all fitting parameters based on [26]

2.3 Effects of Wearout

In this section, the characterizations for TDDB and EM given in the previous sections are leveraged to demonstrate how the effects of wearout due to these mechanisms can be represented at the microarchitectural level as increasing signal delays. This abstraction allows for simple yet sufficiently accurate modeling of wearout and is supported by the literature. Further, a number of other important wearout mechanisms such as NBTI and HCI have shown similar effects on signal delays, however, due to time and space constraints, we have omitted them from our experiments. While this does not affect the applicability of the proposed wearout detection unit, it may affect the MTTF estimates presented in later sections. In order to clearly demonstrate the effect of wearout on signal propagation latency, we further discuss analytical models relating the effects of wearout mechanisms to timing.

The Fitted Elmore Delay model [3], shown in Equation 4, is derived from the Elmore delay equation, arguably the most widely used interconnect delay model [16]. This model describes how delay through interconnects are related to various parameters including: driver impedance, load capacitance, geometry, etc.

$$\begin{aligned} \text{Delay}(r_d, c_l, l, w) = & A \cdot r_d c_a l w + B \cdot r_d c_f l \\ & + C \cdot r_d c_l + D \cdot \frac{r c_a l^2}{2} \\ & + E \cdot \frac{r c_f l^2}{2w} + F \cdot \frac{r l c_l}{w} \end{aligned} \quad (4)$$

After collecting terms, (4) can be rewritten with appropriate substitutions as

$$\begin{aligned} \text{Delay}(r) &= \kappa + r \cdot \gamma \\ \Delta \text{Delay}(\Delta r) &\propto \Delta r \cdot \gamma \end{aligned} \quad (5)$$

where

- r is the resistance (sheet resistance) of the interconnect wire
- κ incorporates all terms in (4) that are independent of r
- γ incorporates all terms in (4) that are dependent on r

As mentioned in Section 2.1 EM studies focusing on interconnects, spanning a wide range of process technologies, have shown sharp rises in resistance as the mass transport of metal begins to inhibit the movement of charge. Couple this understanding with Equation 5, and it follows that EM can reasonably be modeled as an increasing interconnect delay.

Similarly, the literature has shown that TDDB has a detrimental impact on circuit performance. As the gate oxide wears down the combined effects of increased leakage current and

shifting I-V curves results in devices with slower response times [7].

It is a common practice to combine the effects of individual mechanisms in order to characterize their cumulative impact on reliability. Figure 1 depicts the widely accepted bathtub curve, which is used to describe the failure rate for semiconductor devices. The failure rate is represented in terms of failures in time, or FITs, which corresponds to one failure per billion hours of operation. The life of a semiconductor device can be partitioned into three distinct regions: an infant mortality period, a grace period, and a breakdown period. The infant mortality region exhibits a high failure rate, as failures caused by manufacturing defects are expressed early in the device's lifetime. The high failure rate from the infant mortality period rapidly decreases over time and settles to a small, nearly constant, failure rate for the expected lifetime of the device. As the device moves into the breakdown period, the failure rate increases dramatically.

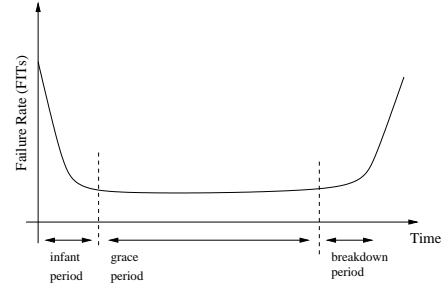


Figure 1: Bathtub curve

Our objective in this work is to develop a wearout detection scheme that accurately identifies when a processor is entering the breakdown period shown in Figure 1. Since wearout mechanisms may be characterized by an increasing latency through semiconductor devices and the breakdown period is caused by the rapid progression of wearout mechanisms, the remainder of this paper describes and evaluates a detection unit suitable for detecting increases in circuit delays over time.

3. WEAROUT ANALYSIS

This section describes the simulation system used to measure the effects of wearout, as well as the simulation results. It includes a discussion of the microprocessor core that was used to conduct experiments, the simulation infrastructure, and an analysis of the effects of wearout on the core.

3.1 CPU Core

In this work, we use a Verilog model of the OpenRISC 1200 (OR1200) CPU core [1]. The OR1200 is an open-source, embedded-style, 32-bit, Harvard microarchitecture that implements the ORBIS32 instruction set. The microprocessor contains a single-issue 5-stage pipeline, with direct mapped 8KB instruction and data caches and virtual memory support. This microprocessor core has been used in a number of commercial products and is capable of running the μ Clinux operating system.

The OR1200 core was synthesized using the Synopsys Phys-

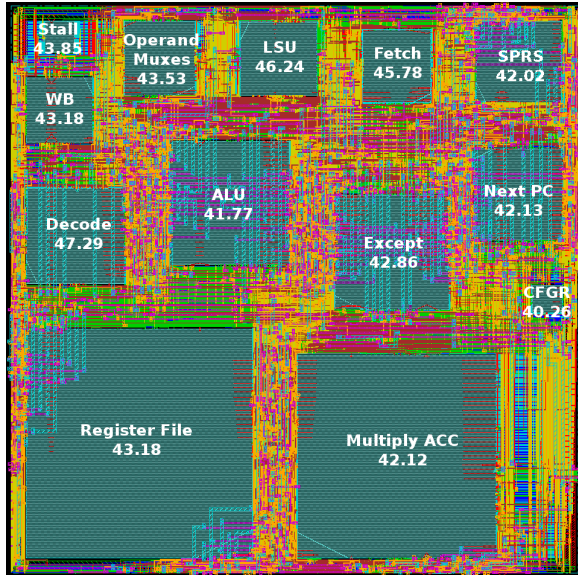


Figure 2: Open Risc 1200 CPU datapath. The figure is annotated with the structure names, and their steady state temperatures in °C.

ical Compiler with an Artisan cell library characterized for a 130 nm process and a clock frequency of 5ns (200 MHz). The synthesized OR1200 netlist was then imported into Cadence First Encounter to conduct floorplanning, cell placement, routing, and clock tree synthesis. From this design flow we obtained very accurate timing information (cell and interconnect delays), circuit parasitics (resistance and capacitance values), and a floorplan for the OR1200 core.

After placement and routing, the total die area of the OR1200 core was $16mm^2$ with an aspect ratio of 1.0. A major portion of this die area is covered by caches, embedded memory and the associated memory management units (MMUs). The OR1200 CPU, which is responsible for the highest power dissipation, occupies a total area of $0.5625mm^2$ with an aspect ratio of 1.0. A die picture, from First Encounter, of the OR1200 CPU with all the metal layers and the interconnects is shown in Figure 2.

3.2 Simulation Infrastructure

In order to analyze the effects of wearout on the microprocessor core, a framework for simulating accelerated wearout as a function of the per-design element MTTF due to EM and TDDB was developed. In this framework, wearout is modeled as an increased delay through logic cells and across interconnects within the OR1200 core. The relationship between wearout and increased delay is discussed in Section 2.3.

As demonstrated in Section 2, the rate at which wearout occurs within the design is heavily dependent upon power, temperature, and activity. In order to determine the relative amount of wearout on each design element (both logic cells and wires), the MTTF for each design element is derived. The MTTF for interconnect elements depends primarily on the rate at which EM occurs, where the MTTF is determined using

Equation 1 described in Section 2. The MTTF for logic cells is derived as a function of TDDB using Equation 3 also from Section 2. These equations require information about activity, temperature, and operating conditions, which are obtained as follows:

Operating conditions: Voltage and frequency values are uniform for all the devices across the chip. The core operating voltage is 1.2 V and the chip is run at a frequency of 200 MHz.

Activity: A set of libraries was developed to monitor the activity of each design element within the OR1200 core as the netlist was being simulated using Synopsys VCS.

Temperature: The steady state temperature of the processor is calculated using HotSpot [20], a tool from the University of Virginia that requires a chip floorplan and a per-structure dynamic power trace. The floorplan was generated using Cadence First Encounter as discussed earlier in Section 3.1. The dynamic power trace was generated using Synopsys PrimePower. PrimePower takes in a synthesized design netlist, cell parasitic data and a value change dump from netlist simulation as input.

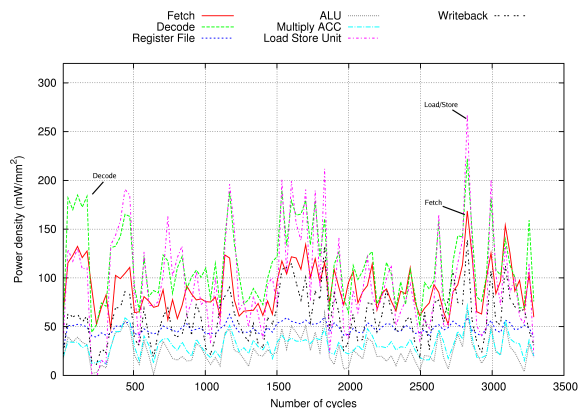
After generating this data, an MTTF value is computed for each logic cell and wire within the design. We then normalize the MTTF to the worst-case (smallest) MTTF across all devices within the design. The relative amount of wearout for each design element is taken to be the reciprocal of this normalized MTTF.

After determining the relative amount of wearout for each device in the design, we simulate the effects of wearout on the design as a whole. This is done by first selecting a worst-case delay to be induced on design elements as a result of wearout. We vary this delay on the interval of zero to a single inverter delay. The worst-case delay is then multiplied by the relative amount of wearout for each device to determine how much delay the device will experience as a result of wearout. We then simulate the synthesized netlist with these delay values annotated to each logic cell and wire within the design. While conducting these simulations, we monitor the latency observed for signals arriving at the output of a number of architectural structures within the core.

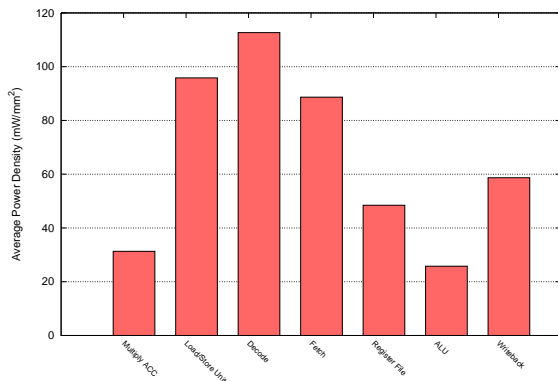
3.3 Simulation Results

Here we present an analysis of the simulation results from the infrastructure described in Section 3.2. Power density and temperature data are presented along with the calculated MTTF for each structure in the design. Further, we demonstrate the impact of wearout on the observed signal propagation latency through the ALU.

We begin by simulating the netlist of the OR1200 core executing the Sobel edge detection benchmark. This simulation was used to generate an activity trace for all the design elements. PrimePower then derives a per-structure power trace for the design given the characteristics of our 130nm cell library. The power consumption trace for every structure is divided by the respective structure area in order to obtain a per-



9(a) Power density trace for OpenRisc 1200 embedded core running Sobel image edge detection benchmark.



9(b) Average power density

Figure 3: Power density results

structure power density trace. These structure areas were obtained from Cadence First Encounter, our placement and routing tool.

The power density trace results are shown in Figure 3. The trace information shown here is limited to the coarse-grained architectural structures within the CPU datapath. Other smaller structures, that have negligible power densities, were left out in order to keep the graph from getting congested. Figure 3(a) shows the power density trace over time, and the average power density per-structure is shown in Figure 3(b). It can be seen from these plots that the Decode unit, Load/Store unit, and Fetch unit, in that order, are the top three structures in terms of power density. Therefore, the likelihood of these structures to have higher temperatures is also greater.

We used the power trace generated by PrimePower, coupled with the floorplan information from Cadence First Encounter, as input to the HotSpot temperature analysis tool. The steady-state temperature results for the execution of the Sobel benchmark are shown in Figure 2. The major structures have been annotated with their name and steady state temperature (in °C). As expected, the temperature more or less follows the trends seen in the power density trace of Figure 3. In Table 1, we demonstrate the MTTF for each module as-

Module	MTTF (years)
Decode	30.0
Fetch	34.9
Freeze	42.3
Writeback	42.7
Operand Muxes	44.9
Register File	45.8
Except	47.4
Next PC	51.1
Multiply ACC	51.2
LSU	87.2
SPRS	111.9
ALU	181.6

Table 1: Calculated MTTF values for each module within the design

suming a 30 year MTTF qualification for the microprocessor. From the MTTF calculations, we can observe that the decode and fetch units are the most likely units to fail. Further, we see that the LSU, despite its relatively high temperature, lies in the middle of the pack. This is due to a relatively low amount of activity over the course of benchmark execution. Also, the ALU winds up being last on the list of structures to fail. This is because the benchmark that was executing has a large fraction of multiplies which were handled by the multiply accumulate unit rather than the ALU.

All the activity, power and temperature data is next used to calculate the relative amount of wearout for each of the design elements. The calculations are done as explained per Section 3.2. Using this information we conduct a series of simulations on a per-structure basis. For each experiment, we increase the delay through each design element of a particular structure by an amount proportional to its relative wearout. The maximum possible delay injected on any device element is the worst-case delay explained previously in Section 3.2. We conduct these experiments using worst-case delay increments of 1ps over the interval of a single inverter delay (20ps) and measure the latency for all the stable output signals (of the given structure being analyzed).

Though we present results in Section 5 for a larger set of architectural structures, here we present some analysis of the ALU as a representative module. Figure 4 demonstrates the increase in average latency observed on the result bus from the ALU versus the average amount of delay injected per design element. The second label for the x-axis on the top of the graph gives the worst case delay injected per logic cell. This figure illustrates that very small timing effects of wearout occurring at the device level are greatly magnified at the architectural level. This happens because any given output signal travels through many design elements within the structure - each of which is undergoing wearout. Therefore, the cumulative shift in the mean latency value for an output signal is considerably large.

There are a few exceptions to this magnification rule. For example, `result[29]` has a very slow growth with increasing latency injection. This is largely because this particular line experienced very little activity during benchmark execution. Since the amount of activity directly affects the rate of

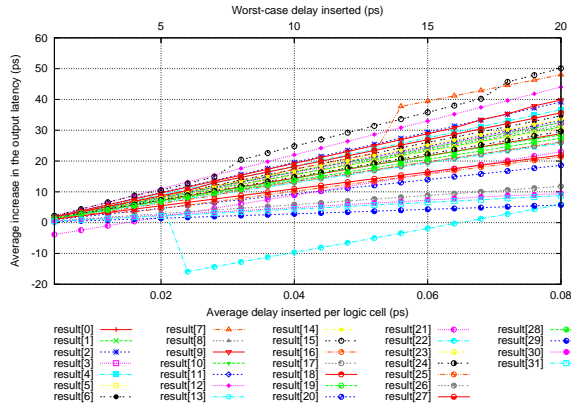


Figure 4: Increase in average output latency for ALU result bus. The lower labels on the x-axis indicate the average latency added per logic cell inside the ALU and the upper labels on the x-axis indicate the maximum latency injected into a logic cell.

wearout, this particular output demonstrated a slow rate of delay growth. Another interesting data point is `result[22]` which has a negative increase in the mean latency value. We found that this effect is caused by a reduction in the number of intermittent values on this wire before the final value stabilizes. Consider an AND gate. Depending upon the relative signal arrival times at the inputs of this gate, the output may or may not experience multiple transitions. Hence, many transitions that happen before an output stabilizes can potentially be masked by a change in timing through the circuit. Many such occurrences could lead to a reduction in the average latency value, as seen for `result[22]` signal.

In the following sections, we describe how we leverage these trends in increasing output latency throughout the breakdown period in order to detect the onset of wearout and predict the failure of structures within the design.

4. WEAROUT DETECTION UNIT

The bathtub curve presented in Section 2.3 demonstrates how failure rates increase throughout the breakdown period of semiconductor devices. This failure rate increases as a result of wearout mechanisms such as TDDB and EM, which have been shown to progress rapidly after a prolonged incubation period. Since these wearout mechanisms are known to increase the delay through logic cells and across interconnects, we identify the breakdown period by an observed increase in delay through architectural structures within the microprocessor.

In the next subsection, we describe the design and operation of the WDU. We then examine how the WDU would be integrated into a microprocessor core. In Section 5, we discuss the area and power overhead of the WDU, its accuracy in detecting wearout, and the increase in processor lifetime that can be achieved by augmenting a design with WDU units and cold spare structures.

4.1 Wearout Detection Unit Design

Figure 4 from Section 3.3 shows that an average delay increase of only 0.04 ps per logic cell within the ALU can lead to more than 20 ps, or about one inverter delay, of additional latency, as observed on the ALU result bus. This illustrates how seemingly negligible wearout effects at the device level are amplified at the architectural level. The wearout detection unit presented in this work leverages this rapidly increasing latency that manifests during the breakdown period to accurately detect wearout and diagnose that the failure of a given structure is imminent. The wearout detection unit (WDU) employs a small, efficient multi-tiered sampling circuit to determine that the propagation latency of a signal at the output of a module is increasing over time, indicating that catastrophic device failure is near.

A circuit-level diagram of the proposed WDU is shown in Figure 5. The WDU consists of three distinct stages. The first stage serves to measure the amount of extra slack time that exists between when a module’s output signal stabilizes and the positive edge of the clock. The second stage of the WDU is used to conduct fine-grained sampling of the latency measurements from Stage 1. The sampled values are then used to calculate an exponential moving average (EMA) [8] which smooths out the sample stream and favors more recently sampled data over older data, highlighting local trends in the observed signal latency. In the final stage, the EMA is used to calculate a coarse-grained running average of the observed latency. The purpose of the final stage is to maintain a long-running history of the observed signal latency, which can be easily differentiated from the local trends that are observed in the second stage. Due to the long incubation periods for EM and TDDB, the coarse-grained running average is required to routinely save state back to a nonvolatile storage, such as flash memory. Also, because the wearout process is fairly gradual (during the grace period), the WDU is not required to be constantly operating.

In the following subsections we detail the design of each stage of our wearout detection unit.

4.1.1 Stage 1: Signal Latency Detection

The input to the first stage of the WDU is an output wire from the module being monitored. The purpose of the first stage is to determine the amount of time between when the input signal arrives and the positive edge of the clock. To this end, the input signal is subjected to a series of delay buffers, where each buffer has a delay of five logic inverters. The signals at the outputs of each of these delay buffers serve to feed a vector of registers. At the positive edge of the clock, some portion of these registers will store the correct output value from the module being monitored, while other registers will store the previous value on the input line. This happens because the increased propagation delay causes the signal to arrive later than the clock edge for a subset of these registers. The value stored at each of these registers is then compared to a stored copy of the original input. This pair-wise comparison gives a bit vector which is used to estimate the amount of slack between the signal arrival time and the positive edge of the clock. This slack value is equal to the number of zero values read from the computed bit vector.

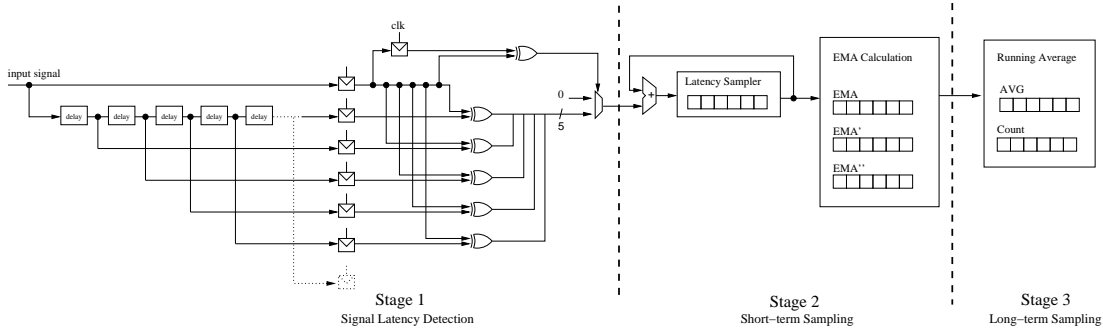


Figure 5: Wearout detection unit

The bit vector representing the signal arrival time is then encoded and passed to the second stage of the WDU. It is important to note that this latency value is only passed on if the value on the input line changed from the previous cycle. If there was no transition on the input line, then the signal latency information is not valid, and would skew the data sample if it were used.

One important parameter in designing Stage 1 of the WDU is the length of the chain of buffers used to determine the amount of slack time. The number of buffers in this chain is proportional to the difference between the clock cycle time and the minimum signal arrival time, divided by five inverter delays. The division by five inverter delays results in a coarser resolution, which does not significantly impact functionality, but helps in maintaining a reasonable bit vector width. Note that in a high-performance design with tighter timing constraints, there would be less variance in output delays. Therefore, to maintain the same bit vector width the delay in the buffers would be less than five inverters in order to provide the same timing resolution. In our experiments, measuring signal latency for the ALU we used thirty-eight delay buffers spanning the signal latency range of 190 inverter delays. As we demonstrate later, the length of this delay chain does not significantly impact the overall area of the WDU.

4.1.2 Stage 2: Short-term Sampling

Since the propagation latency for a signal is highly dependent upon the input vector, the first portion of the sampling unit in Stage 2 is meant to simply capture an accurate local sample of the latency for a given signal. In our experiments, we sampled the input signal at every other transition and accumulated 1024 of these samples before calculating a moving average for the latency data.

In order to quickly identify a trend of increasing latency on an input, we employ a circuit that calculates a triple-smoothed exponential moving average (EMA) or TRIX value [8]. The TRIX value calculation is used to both smooth out any outlying data points that may be caused by program phase behavior, as well as to aid in trend analysis for the signal latency by putting emphasis on recent data points. This trend analysis is necessary to provide an early indication that the signal latency is increasing and that the device may have entered the breakdown region.

In order to simplify the TRIX calculation and maintain a

high degree of accuracy, each accumulated sample sum is taken to be a snapshot of the signal latency. To compute the TRIX value, the EMA value is calculated three times, where each EMA value is calculated as follows:

$$EMA = \alpha * sample + (1 - \alpha)EMA_{previous}$$

Here, α is a fitting parameter that determines how much emphasis is placed on new sample values versus the previously computed EMA. In our experiments, we used an α value of 2^{-6} so that multiplications by α may be conveniently calculated by shift operations in the hardware. The TRIX value is then computed by calculating three EMA values as follows:

$$\begin{aligned} EMA_1 &= \alpha(sample - EMA_{1previous}) + EMA_{1previous} \\ EMA_2 &= \alpha(EMA_1 - EMA_{2previous}) + EMA_{2previous} \\ TRIX &= \alpha(EMA_2 - TRIX_{previous}) + TRIX_{previous} \end{aligned}$$

Since the three EMA calculations are identical, the impact of Stage 2 on both area and power can be minimized by spanning the calculation of the TRIX value over three cycles and only synthesizing a single piece of the EMA calculation hardware. The TRIX value calculated here represents a snapshot of local signal latency trends and is routinely sampled by the Stage 3 long-term sampling unit as described in the next section.

4.1.3 Stage 3: Long-term Sampling

The final stage, Stage 3, conducts a coarse-grained sampling of the TRIX values from Stage 2 and calculates a running average. Since the grace period, which precedes the breakdown period, is exceptionally long, this running average will be biased towards early-life operating conditions. While this circuit cannot maintain an infinite running average of samples, this unit is rarely updated, and is parameterized such that it can maintain sufficient history to bias old values from the grace period of the bathtub curve appropriately.

Stage 3 of the WDU consists of a register containing the running average and a counter that holds the count for the number of TRIX samples that were used to compute this running average. Since, the TRIX value calculated in the second stage is inherently biased towards recent trends in the circuit's latency, it will quickly register any increasing latency

on the signal under observation. Further, since the coarse-grained running average of this final stage is biased towards much older latency information, a steadily growing distance between these two values serves to characterize the onset of the breakdown period. To do this in hardware, every new incoming TRIX value from second stage is compared against the running average. If this difference consistently surpasses a predefined threshold, the signal is flagged and the unit is marked as failing. Otherwise the running average is updated using the incoming TRIX value.

In this work, we assume that an increase in circuit latency of 20% indicates total failure. We justify this assumption by citing the fact that many chip manufacturers impose a guard-band of 20% of the cycle time to account for process variation. This means that in the absence of process variation, a critical path in the design may be delayed by up to 20% of the clock period before timing violations occur. When evaluating the predictive accuracy of the WDU, we require that it predicts the failure before the circuit delay increase surpasses the 20% value.

5. WEAROUT DETECTION UNIT EVALUATION

In this section, we present area and power consumption results for a synthesized WDU. We also evaluate the efficacy of the WDU in detecting the onset of wearout and predicting failure. It is envisioned that this information would be used to reconfigure the system, potentially swapping in cold spare devices or deallocating structures, in response to the threat of failure.

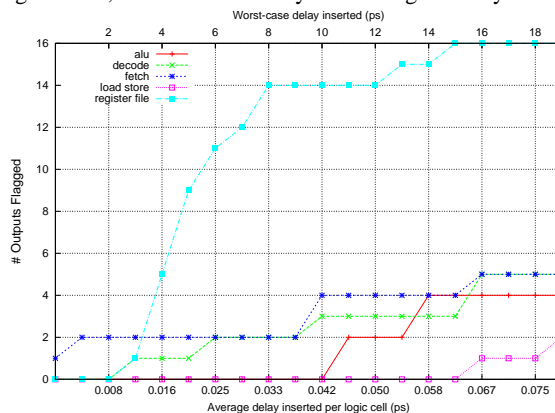
In Table 2, we demonstrate the area and power consumption results from synthesis and place and route of the WDU design. This table shows that a single WDU unit consumes only about 0.0075mm^2 (excluding the non-volatile storage) and that adding a single WDU increases the overall CPU datapath area by only about 1.3%. To put this into perspective, the die area of a typical ARM 926E implemented in a 130 nm process is 5.0mm^2 , and so a single WDU would only consume 0.15% of the overall chip area. The power consumption for the WDU is estimated by Synopsys Physical Compiler to be 0.93mW , as compared to an estimate of about 78mW for the OR1200 datapath by Physical Compiler.

	WDU	CPU Datapath
Area (mm^2)	0.007514	0.5625
Power (mW)	0.927	78.0438

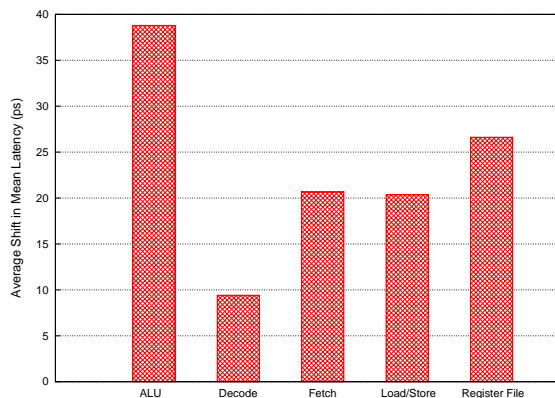
Table 2: Area and power results for a single wearout detection unit

The next set of experiments were conducted by simulating the WDU given the increase in delay from our accelerated wearout experiments. In these experiments, we track the minimum amount of delay that the system must experience before a module output is flagged as increasing in latency, marking the module as failing. Figure 6(a) illustrates the number of module outputs that are flagged versus the average and maximum amount of delay that is injected at the device level. From this figure, we can see that some modules require significantly

more wear than others before the WDU can detect a shift in the mean latency. For example, it is quickly determined that the register file is experiencing wear on multiple different outputs, whereas, it requires much more wear before the WDU flags the LSU as likely to fail. This is typically because the amount of delay added to the LSU is relatively small due to its high MTTF, the result of a fairly low average activity factor.



9(a) Diagnostic for different CPU structures.



9(b) Shift in mean detected

Figure 6: WDU results for the OR1200 running Sobel

In Figure 6(b), we correlate the time a latency increase is first detected with the mean latency increase observed. From this data, we can see that even though the average amount of injected latency is quite small (e.g. 2 fs for the ALU), the shift in the mean signal arrival time increases dramatically (almost 40 ps for the ALU). This phenomenon allows the WDU to very accurately detect the onset of wearout. Further, even though the WDU did not detect the increase in latency on *all* output signals for any module, it was able to detect an increase in latency on *some* signals, which is sufficient to determine that a module is undergoing wearout and likely to fail.

6. RELATED WORK

Issues in technology scaling and process variation have raised concerns for reliability in future microprocessor generations. Recent research work has attempted to diagnose and, in some cases, reconfigure the processing core to increase operational

lifetime. In this section, we briefly discuss this related work and how it has influenced our own.

DIVA [5] targeted soft error detection and online correction using a low cost and power alternative to full scale redundancy. It uses a simplified second core to monitor any differences in execution from the main core, and flushes the pipeline in the event of a mismatch. Though this technique will prove useful in certain contexts, it requires the design and verification overhead of building a second microprocessor core, as well as the on-chip area overhead of this redundant structure.

Bower et al. [12] present a method for detecting and diagnosing failures using a DIVA checker. The proposed technique relies on maintaining counters for each architectural unit in the microprocessor and incrementing the counter for each unit that was used when incorrect execution is detected by the DIVA checker. When a unit is found to have count over a predefined threshold it is deemed faulty. After detection of the fault, re-configuration is done using the cold spares.

Srinivasan et al. have also been very active in promoting the need for robust designs that can mitigate the inevitable onslaught of reliability challenges on the horizon [25]. Their work attempts to accurately model the MTTF of a device over its operating lifetime which allows intelligent application of techniques like dynamic voltage and/or frequency scaling to meet reliability goals. Although we may share some common physical models, the focus of this paper is not to ensure that designs can achieve any particular reliability goals but rather to enable a design to recognize behavior that is symptomatic of wearout and react accordingly.

The notion of "prognostic cells" championed by the Ridgetop Group [2] is also an attempt at symptom based detection of wearout. Their proprietary library of Sentinel Silicon cells claim to provide a defense against an assortment of failure mechanisms (TDDB, EM, NBTI, etc.) through the use of in-situ canary circuits. Again, although similar in nature we believe our proposal reaps a significant advantage over their scan-chain based technique because the WDU is an online approach that is transparent to the end user.

Analyzing circuit timing in order to self-tune processor clock frequencies and voltages is a well studied area. Kehl [18] discusses a technique for retiming circuits based on how close to the positive clock edge worst-case latency signals arrive. The presented technique requires offline testing involving a set of stored test vectors in order to tune the clock frequency. Though the proposed circuit design is similar in nature, it only examines a small period before the clock edge and is only concerned with worst case timing estimation, whereas the WDU employs sampling over a larger time span in order to conduct average case timing analysis. Similarly Razor [6] is a technique for detecting timing violations using time-delayed redundant latches to determine if operating voltages may be lowered. Again, this work studies only worst-case latencies for signals arriving very close to the clock edge.

7. CONCLUSION

In this work, we propose an online wearout detection unit to predict the failure of architectural structures within microprocessor cores. This unit leverages the symptoms of wearout

to predict imminent failure. To the best of our knowledge, this is the first solution of its kind which seeks to utilize signal latency information for wearout detection and failure prediction. To justify the design of the WDU, we conducted accelerated wearout experiments on the OpenRISC 1200 embedded microprocessor core that was synthesized and routed using industry standard CAD tools. Further, accurate models for TDDB and EM were used to model the wearout related failures and determine the MTTFs for devices within the design. The results of these accelerated wearout experiments showed that even small increases in latency at the device level could cause significant delay at the structural level. This result formed the basis for the design of our WDU.

The WDU presented in this work accurately detects and diagnoses wearout with only a small area footprint. The WDU is shown to successfully detect the trends of increasing latency across multiple output signals for each module of the OpenRISC 1200 that was examined.

9

8. REFERENCES

- [1] Openrisc 1200, 2006. http://www.opencores.org/projects.cgi/web/or1k/openrisc_1200.
- [2] Ridgetop group, 2006. <http://www.ridgetop-group.com/>.
- [3] A. I. Abou-Seido, B. Nowak, and C. Chu. Fitted elmore delay: A simple and accurate interconnect delay model. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 12(7):691–696, July 2004.
- [4] J. S. S. T. Association. Failure mechanisms and models for semiconductor devices. Technical Report JEP122C, JEDEC Solid State Technology Association, Mar. 2006.
- [5] T. Austin. Diva: a reliable substrate for deep submicron microarchitecture design. In *Proc. of the 32nd Annual International Symposium on Microarchitecture*, pages 196–207, 1999.
- [6] T. Austin, D. Blaauw, T. Mudge, and K. Flautner. Making typical silicon matter with razor. *IEEE Computer*, 37(3):57–65, Mar. 2004.
- [7] A. Avellan and W. H. Krautschneider. Impact of soft and hard breakdown on analog and digital circuits. *IEEE Transactions on Device and Materials Reliability*, 4(4):676–680, Dec. 2004.
- [8] M. Batty. Monitoring an exponential smoothing forecasting system. *Operational Research Quarterly*, 20(3):319–325, 1969.
- [9] D. Bernick, B. Bruckert, P. D. Vigna, D. Garcia, R. Jardine, J. Klecka, and J. Smullen. Nonstop Advanced Architecture. In *International Conference on Dependable Systems and Networks*, pages 12–21, June 2005.
- [10] J. R. Black. Mass transport of aluminum by momentum exchange with conducting electrons. In *Proc. of the 1967 International Reliability Physics Symposium*, Nov. 1967.
- [11] S. Borkar. Designing reliable systems from unreliable components: The challenges of transistor variability and degradation. *IEEE Micro*, 25(6):10–16, 2005.
- [12] F. A. Bower, D. J. Sorin, and S. Ozev. A mechanism for online diagnosis of hard faults in microprocessors. In *Proc. of the 38th Annual International Symposium on Microarchitecture*, pages 197–208, 2005.
- [13] A. Christou. *Electromigration and Electronic Device*

- Degradation*. John Wiley and Sons, Inc., 1994.
- [14] A. Dasgupta and R. Karri. Electromigration reliability enhancement via bus activity distribution. In *Proc. of the 33rd Design Automation Conference*, pages 353–356, ACMpress-a, 1996. ACMpress.
 - [15] D. Dumin. *Oxide Reliability: A Summary of Silicon Oxide Wearout, Breakdown, and Reliability*. World Scientific Publishing Co. Pte. Ltd., 2002.
 - [16] W. C. Elmore. The transient response of damped linear network with particular regard to wideband amplifiers. *Journal of Applied Physics*, 19(1):55–63, Jan. 1948.
 - [17] C.-K. Hu et al. Effects of overlayers on electromigration reliability improvement for cu/low k interconnects. In *Proc. of the 2004 International Reliability Physics Symposium*, pages 222–228, Apr. 2004.
 - [18] T. Kehl. Hardware self-tuning and circuit performance monitoring. In *Proc. of the 1993 International Conference on Computer Design*, pages 188–192, Oct. 1993.
 - [19] E. Ogawa. Electromigration reliability issues in dual-damascene cu interconnections. *IEEE Transactions on Reliability*, 51(4):403–419, Dec. 2002.
 - [20] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. Temperature-aware microarchitecture: Modeling and implementation. *ACM Transactions on Architecture and Code Optimization*, 1(1):94–125, 2004.
 - [21] P. Solomon. Breakdown in silicon oxide - a review. *Journal of Vacuum Science and Technology*, 14(5):1122–1130, Sept. 1977.
 - [22] L. Spainhower and T. Gregg. IBM S/390 Parallel Enterprise Server G5 Fault Tolerance: A Historical Perspective. *IBM Journal of Research and Development*, 43(6):863–873, 1999.
 - [23] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. The case for lifetime reliability-aware microprocessors. In *Proc. of the 31st Annual International Symposium on Computer Architecture*, pages 276–287, June 2004.
 - [24] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. The impact of technology scaling on lifetime reliability. In *Proc. of the 2004 International Conference on Dependable Systems and Networks*, pages 177–186, June 2004.
 - [25] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. Exploiting structural duplication for lifetime reliability enhancement. In *Proc. of the 32nd Annual International Symposium on Computer Architecture*, pages 520–531, June 2005.
 - [26] E. Wu et al. Interplay of voltage and temperature acceleration of oxide breakdown for ultra-thin gate oxides. *Solid-State Electronics*, 46:1787–1798, 2002.