A Proactive Wearout Recovery Approach for Exploiting Microarchitectural Redundancy to Extend Cache SRAM Lifetime

Jeonghee Shin†, Victor Zyuban‡, Pradip Bose‡ and Timothy M. Pinkston†

† EE-Systems, University of Southern California, Los Angeles, CA
‡ IBM T. J. Watson Research Center, Yorktown Heights, NY

ISCA 2008, Beijing, China
Chip Lifetime Reliability

Technology scaling

- Device feature sizes are decreasing, but variability is increasing
- Supply voltage not scaling with device sizes
- Increased current density, power density, and temperature
- Increased number of “weaker” devices in vulnerable structures
Chip Lifetime Reliability

chip failure rate

Infant mortality period
(manufacturing defects)

Useful lifetime period

Aging period
(wearout failures)

time

Lifetime reliability-aware designs

◆ Mitigate wearout effects by reducing the stress conditions of various failure mechanisms: to exploit recovery effects of NBTI such as supply voltage, frequency, threshold voltage, duty cycle, ...

◆ Tolerate wearout failures using microarchitectural redundancy proactively!! sparing, graceful performance degradation techniques
Conventional “Reactive” Use of Redundancy

- Isolate and replace microarchitectural component with a spare on *first-come, first-served* basis *after* they fail

8 arrays per way

Way 1

Way 2

Way 7

Way 8

Cache line interleaving across the arrays composing the same associative way

e.g., implemented in Intel Itanium 2, …
Chip Lifetime Reliability

Lifetime reliability-aware designs

- Mitigate wearout effects by reducing the stress conditions of various failure mechanisms: to exploit recovery effects of NBTI such as supply voltage, frequency, threshold voltage, duty cycle, ...

- Tolerate wearout failures using microarchitectural redundancy proactively!! sparing, graceful performance degradation techniques

chip failure rate

Infant mortality period (manufacturing defects)

Useful lifetime period

Aging period (wearout failures)

Chip lifetime reliability

time
“Proactive” Wearout Recovery in a Nutshell

- Redundancy is used proactively to suspend and “recover” from component wearout well before failure occurs by temporarily deactivating (recovery mode) and later reactivated (active mode) all non-faulty components periodically, on a rotating basis.

**Advantages of Proactive Approach**

- Better use of spare components to better balance wearout across the chip to stave off chip-kill owing to heavily worn-out components.
- The wearout of components due to certain failure mechanisms (e.g., NBTI) can be reversed during recovery mode.
- A limited amount of redundancy used proactively provides benefits.
Outline

- Introduction, motivation, basic idea
- Exploitation of NBTI wearout recovery properties
- Proactive wearout recovery approach
  - Circuit-level techniques for wearout recovery of SRAM arrays
  - Cache architectures with proactive use of redundancy
- Evaluation methodology and results
- Related work
- Wrap-up
- **Vulnerable device type**: PFET devices
- **Stress condition**: Negative bias ($V_{gs} < 0$), high temp’s
- **Cause**: Increased positive charges created by dissociated silicon-hydrogen bonds at silicon & gate oxide interface
- **Effect**: Increases threshold voltage, $V_T$, which increases device delay and degrades SRAM cell stability
- **Recovery condition**: No or positive bias ($V_{gs} \geq 0$)

**Exploit during recovery mode**

\[
\Delta V_T \propto t_{ox} \cdot \sqrt{C_{ox}} \cdot \left| V_{gs} - V_T \right|
\]

\[
E_{ox} / E_0 \cdot e^{-E_{a,NBTI} / kT} \cdot t^n
\]

**Stress**

**Recovery**

\[
\Delta V_T = \Delta V_{ur} + (\Delta V_{T0} - \Delta V_{ur}) \cdot e^{\frac{t-t_0}{\tau}}
\]

---

R. Vattikonda, et al., 2006

Exploit during recovery mode
Implementing SRAM Arrays with Intense Recovery

- Low area and design complexity overhead
  - Virtual power rails already exist for most array designs
  - The power line running vertically across array cells can be divided into two, one for each side

- Discharge the virtual $V_{dd}$ of the PFETs entering into recovery mode
  - Write the proper cell values to charge the gate of the PFETs
Proactive Wearout Recovery of Cache SRAMs

- Proactive use of spare array(s) allow the other SRAM arrays to enter into recovery mode on a rotating basis to recover from NBTI wearout.

- Must maintain correctness amid dynamic transitions between active and recovery modes of array operation.
  - Retain valid cache contents despite losing cell values ⇒ *drain process*
  - Supply requested data only from active arrays ⇒ *array selection logic*

Drain Process

- *Invalidation mechanism*
  - Write back dirty/shared lines; invalidate lines cached in upper levels
  - May cause extra cache misses (due to requests to invalidated lines)

- *Migration mechanism*
  - Migrate array contents to the newly allocated array (e.g., spare) via dedicated “migration” links or via the normal read/write cache bus
  - May add wiring overhead and/or additional resource conflicts
Invalidation Drain Mechanism

8 arrays per way

Array entering recovery mode

Replace a_{11}
Migration Drain Mechanism

Array entering into recovery mode

Replace $a_{12}$

Replace $a_{11}$

8 associative ways

8 arrays per way
Evaluation Methodology

 ► Performance
  ◆ IBM Mambo (cycle-accurate, standalone/full-system mode)
  ◆ POWER5-like processor configurations (256KB L2, 4MB L3)
  ◆ Proposed (and other) techniques are faithfully implemented in Mambo
  ◆ Memory-intensive microbenchmarks (working data set size = L2 cache size) as well as SPLASH applications

 ► Lifetime reliability
  ◆ Baseline configuration: 256KB cache consisting of 64 arrays, each with 128 columns and 256 rows, and no redundancy
  ◆ Reactive use of redundant components: cell bits, columns, rows or arrays added to the baseline configuration, depending on the type of redundancy evaluated
  ◆ The lifetime reliability of each evaluated configuration is quantified using a Matlab toolbox and derived SRAM cell lifetime distributions
<table>
<thead>
<tr>
<th>Redundancy Type</th>
<th>SystemSurvives As Long As</th>
<th>Lifetime Distribution</th>
<th>Application, Technology, Circuit Design, and Process Variation Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold redundancy</td>
<td>at least K components survive.</td>
<td>( N ) components survive as long as at least one component survives.</td>
<td>Microarchitecture parameters</td>
</tr>
<tr>
<td>Warm redundancy</td>
<td>( F(t) = F(t) \otimes F(t) \otimes \cdots \otimes F(t) )</td>
<td>( R(t) = \sum_{i=0}^{N-K} \binom{N}{i} R(t)(1-R(t))^i )</td>
<td>Matlab Reliability Analysis Toolbox</td>
</tr>
</tbody>
</table>

**Matlab Reliability Analysis Toolbox**

- **fseries.m**
- **fecc.m**
- **fspare.m**
- **fmttf.m**
- **fstandby.m**
- **fparallel.m**
- **fstandby.m**
- **fparallel.m**

*Developed Matlab reliability toolbox for any type of system customized for arrays.*
Impact of Drain Process on Performance

- Some performance loss due to cache locking, misses, bus contention
- Performance loss is negligible even for the memory-intensive workloads (i.e., < 0.1% in IPC)
- Performance loss becomes even more insignificant as the interval between successive drains increases
Lifetime Reliability Analysis

- balanced: up to 3x improvement
- reactive + balanced: nearly same
- proactive: 5x to 10x improvement
- proactive + balanced: 6x to 16x (3x to 5x over balanced alone)

• balanced (50%) duty cycle
• reactive + balanced: nearly same
• proactive: 5x to 10x improvement
• proactive + balanced: 6x to 16x (3x to 5x over balanced alone)
Proactive vs. Reactive Approaches w.r.t. NBTI

- **Lifetime vs. Area overhead**
- **Lifetime vs. Performance loss**

- **ECC:** < 3x improvement but area overhead increases as granularity decreases
- **Reactive Sparing:** < 3x improvement, also with increasing area overhead
- **Graceful Performance Degradation (GPD):** < 2x improvement with IPC loss
- **Proactive Wearout Recovery (recover an array once every million cycles):**
  - ~7x lifetime improvement with only 2% area overhead and < 0.001% IPC loss

---

SMART Superior Multiprocessor ARchiTecture - http://ceng.usc.edu/smart/

IBM
Related Work

- Conventional reactive approaches for using microarchitectural redundancy to tolerate wearout effects to extend chip lifetime
  - Sparing and/or graceful performance degradation of register files, branch history table, load/store queues, reorder buffer, L1 cache and functional units (M. A. Lucente, et al., 1991; F. A. Bower, et al., 2004; J. Srinivasan, et al., 2005; many others)
  - Redundancy used reactively which prevents components from being able to suspend or recover from wearout

- Other techniques for mitigating or tolerating wearout effects
  - Power gating and voltage scaling techniques: aim to reduce the stress condition of wearout failure mechanisms (J. Srinivasan, et al., 2004; X. Yang, et al., 2007; others)
  - Cell value flipping techniques: attempt to balance the duty cycle of paired PFET devices (S. V. Kumar, et al., 2006, J. Abella, et al., 2007)
  - Delay margin and circuit delay tuning of parameters: transistor size, threshold voltage, supply voltage, etc. (B. C. Paul, et al., 2005; R. Vattikonda, et al., 2006; X. Yang and K. Saluja, 2007; others)
  - Lifetime reliability is prolonged by proactive measures, but redundancy is not exploited and recovery effects are not fully exploited (intensified)
Wrap-up

◆ Conclusion:
  ◇ Proactive wearout recovery can significantly extend cache SRAM lifetime by fully exploiting NBTI recovery properties:
    ❖ Lifetime extended by ~ 7x at 2% area overhead with < 0.001% performance loss on SPLASH applications
    ❖ Lifetime extended by ~ 5x to 10x with < 0.1% performance loss on memory-intensive workloads

◆ Future Work:
  ◇ Recovery scheduling techniques to better balance wearout
  ◇ Extension of the proposed ideas to other microarchitectural structures and other failure mechanisms
  ◇ Evaluation of the impact of using reactive and proactive approaches in combination on various sets of workloads to find best tradeoffs in area, power, performance, and reliability
Thank You!
Back-up Slides
Modeling SRAM Cell Lifetime Distribution w.r.t. NBTI

- $V_T$ shift over time due to NBTI
  - Technology dependent

- Duty cycle distribution of applied workloads
  - Application dependent

- Cell stability distribution after manufacturing
  - Technology & circuit design dependent

- Convert $V_T$ shift to cell stability loss in $\sigma$
  - Technology & circuit design dependent

- Cell stability loss distribution over time

- Cell lifetime distribution, $F(t)$

As the $V_T$ of PFETs shifts over time, the stability of the cell degrades & eventually fails.
Lifetime Reliability Modeling for Redundant (Cache Memory) Systems

Limitations of the sum-of-failure-rates model

- The first component failure causes the microarchitecture to fail
- Microarchitectures employing redundancy continue to be operative in the presence of failures
- The failure rate of components is constant over time (i.e., exponential lifetime distribution)
- Failure rate during aging period tends to increase over time

*K-out-of-N models*

- *K*-out-of-*N* redundant systems consist of *N* components and survive as long as *K* components are non-faulty, i.e., can tolerate *N*-*K* failures by any means

Redundancy types

- Warm: powered on at system deployment (e.g., GPD, ECC and warm sparing)
- Cold: powered off until put into use (e.g., cold sparing)

We derive lifetime distribution of SRAM cells w.r.t. NBTI, by taking into account architectural effects as well as technology, circuit design and process variation effects.
Exploitation of NBTI Recovery Effects

- **Power Gating**
  - $1/3$-$2/3 \text{V}_{\text{dd}}$ across gates
  - Not sufficient to stimulate the NBTI recovery effect

- **Wearout Recovery**
  - No electric field across gates for NBTI recovery
  - Higher leakage power than power gating

- **Intense Wearout Recovery**
  - Reverse bias to accelerate and intensify NBTI recovery effect
  - Applicable to SRAM cells
# Matlab Reliability Analysis Toolbox

Developed Matlab reliability toolbox for any types of system

<table>
<thead>
<tr>
<th>Number of system components</th>
<th>Number of redundant components</th>
<th>Redundancy type</th>
<th>Lifetime distr. function of components</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N )</td>
<td>( N-K )</td>
<td>warm/cold</td>
<td>( F(t) )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Warm redundancy</th>
<th>1-out-of-( N )</th>
<th>The system consisting of ( N ) components survives as long as at least one component survives.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F_s(t) = \prod_{i=1}^{N} F(t) )</td>
<td>( R_s(t) = \sum_{i=K}^{N} \binom{N}{i} (R(t))^i (1 - R(t))^{N-i} )</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cold redundancy</th>
<th>K-out-of-( N )</th>
<th>The system consisting of ( N ) components survives as long as at least ( K ) components survive.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F_s(t) = f(t) \otimes \ldots \otimes f(t) \otimes F(t) ) ( \text{N-1} )</td>
<td>( R_s(t) = \sum_{r=0}^{N-K} W(K, r) )</td>
<td></td>
</tr>
<tr>
<td>( W(i, j) = \sum_{r=0}^{j} W(i-1, r) \cdot N(j-r) )</td>
<td>( R(t) = 1 - F(t) )</td>
<td></td>
</tr>
</tbody>
</table>

- \( fparallel.m \)
- \( fstandby.m \)
- \( fmttf.m \)
- \( fecc.m \)
- \( fspare.m \)
- \( fseries.m \)
- \( \ldots \)
SRAM Arrays Are Most Susceptible to NBTI

► 6T SRAM cell

As the $V_T$ of the PFETs increases over time, the stability of the cell degrades & eventually the cell value flips

SRAM arrays more vulnerable to NBTI

- SRAM generally takes up a large portion of the chip, having a large number of devices vulnerable to NBTI
- Array cells hold the same value over longer time periods, causing a higher duty cycle for one of the two PFETs of the cells
- Cell stability degradation is more complicated than that of logic circuit speed

D. W. Plass, et al., 2007
H. Q. Le, et al., 2007

► IBM POWER6

≈ 20% of chip size
(≈ 0.2B PFETs)

SMART Superior Multiprocessor ARchiTecture - http://ceng.usc.edu/smart/

IBM