A Software Technique to Improve Yield of Processor Chips in Presence of Ultra-Leaky SRAM Cells Caused by Process Variation

Maziar Goudarzi, Tohru Ishihara, Hiroto Yasuura

System LSI Research Center
Kyushu University, Fukuoka, Japan
Outline

- Background
  - Process variation
  - Power in nanometer embedded processor-based systems

- Our work
  - Motivation
  - Approach
  - Experiments

- Summary and Future work

---

1 This is part of the CREST “Ultra Low Power Design Projects” sponsored by Japan Science and Technology Corporation (JST), http://www.slrc.kyushu-u.ac.jp/~ishihara/CREST/e_kenkyu.html
Feature Size Scale Down

We focus on Threshold Voltage ($V_{th}$) variation

Both inter-die and intra-die variations become increasingly important!

Our Focus:  
Intra-die (Within-die) $V_{th}$ Variation

Large Intra-Die Variation

Current  3-sigma = 13%  
$V_{th}$  3-sigma = 67mV

Variation is huge in small transistors

\[
\sigma_{Vth} = \frac{q}{C_{ox}} \sqrt{\frac{N_a \cdot W_{dm}}{3 \cdot L \cdot W}}
\]

$L, W$: Effective channel length and width  
$q$: electron charge  
$C_{ox}$: oxide capacitance  
$N_a$: substrate doping concentration  
$W_{dm}$: maximum depletion width

Unavoidable Cause of $V_{th}$ Variation: Random Dopant Fluctuation (RDF)

Nature of variations
- Systematic
- Random

ITRS-2005 roadmap forecast

Source: S. Borkar
Our Focus: Leakage Power

Power consumption
- Dynamic
  - activity-based
- Static (leakage)
  - activity-independent

Trend
- Traditionally:
  - Dynamic >> Static
- Nanometer technologies
  - Static >> Dynamic

Source: P.K. Huang, S. Ghiasi (DAC’06)
Our Focus: Caches Memories

- Largest portion of chips => biggest leakage
- Minimum-area transistors => most susceptible to process variation

\[ \sigma_{Vth} = \frac{q}{C_{ox}} \sqrt[3]{\frac{N_a \cdot W_{dm}}{3 \cdot L \cdot W}} \]
Process Variation at 90nm

\[ I_{\text{Subthreshold}} \propto \frac{W \cdot V_T^2}{T_{\text{ox}} \cdot L} \cdot \exp\left(\frac{-V_{\text{th}}}{\alpha \cdot V_T}\right) \]

\( V_T \): Thermal voltage (25mV@room temperature)
\( \alpha \): Sub-threshold factor (1.40~1.65)
\( T_{\text{ox}} \): Oxide thickness

<table>
<thead>
<tr>
<th>Year</th>
<th>min. ( L ) [nm]</th>
<th>(^1V_{TH} ) [V]</th>
<th>(^2V_{TH} ) [V]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td>37 (90)</td>
<td>0.32</td>
<td>0.12</td>
</tr>
<tr>
<td>2005</td>
<td>32 (80)</td>
<td>0.33</td>
<td>0.09</td>
</tr>
<tr>
<td>2006</td>
<td>28 (70)</td>
<td>0.34</td>
<td>0.06</td>
</tr>
</tbody>
</table>

1: Low Operating Power Process  2: MPU process

Ultra-Leaky Transistor (ULT): Transistors that leak beyond a given constraint

1 transistor out of 64K-Byte SRAM

\( 5\sigma_{V_{\text{th}}} = 0.3V \)

Leakage is 1,400x higher than nominal!

100 tr.

Threshold Voltage

\( \pm \sigma \): 68.3%
\( \pm 2\sigma \): 95.4%
\( \pm 3\sigma \): 99.7%
\( \pm 4\sigma \): 99.9936%
\( \pm 5\sigma \): 99.99994%

Large Leak

Large Delay

330x
Ultra-Leaky SRAM Cells Problem

Problem
- Ultra-leaky cache cells dissipate lots of power
- Especially for long-standby applications, cause rapid discharge of battery
Ultra-Leaky SRAM Cells Problem (cont’d)

Naïve solution

– Mark as faulty, replace with spare row/column

– Disadvantages

- Spares may be leaky themselves
- Spares should replace slow/faulty cells as well
- Fuse-blowing expensive and slow
- Aging may introduce ULTs over time
- Temperature may also introduce ULTs
Our Fundamental Observation:
Cell Leakage is Value-Dependant

- If M2, M3, or M5 is leaky, the SRAM cell is 1-leaky
- If M1, M4, or M6 is leaky, the SRAM cell is 0-leaky

Charged to $V_{dd}$ at inactive mode

Our Approach:
Store the Leakage-Safe Value when entering standby mode
Flow of Operations

1. Offline Testing/Booting Phase
2. Detect Leaky Cache Lines
3. RTOS Schedules Apps.
4. Decision to go standby
5. Actually go standby
6. Suppress Leaky Cache Lines
7. Leakage is saved here. Suitable for long-standby low-power applications
8. Wakeup
Offline Testing Phase

Goal:
- Detect location of ULTs
- Location accuracy: cache line or cache cell

Idea
- $\Delta I_{\text{DDQ}}$ Testing:
  - If the leaky cell is sensitized, the quiescent current reflects an abnormal change.

General outline
- Write all 0’s, then all 1’s to every cache line and measure the leakage current
Improvement in Leakage Yield

Leakage Yield = % of chips meeting a given leakage constraint

Experiments:
- Monte Carlo simulation
- 1000 chips
- 32 Kb data + 22 Kb tag
- 60mv within-die $V_{th}$ variation
- Nominal values from a 90nm process
  $V_{th}=320mv$

Nominal transistor leakage = 0.345 nA
Maximum Leakage Power Saving vs. Within-die Variation

Nominal transistor leakage = 0.345 nA
# Associated Costs

<table>
<thead>
<tr>
<th>Costs</th>
<th>Why to pay</th>
<th>When to pay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>Run instructions to store leakage-safe values in leaky cache lines</td>
<td>When going to standby mode</td>
</tr>
<tr>
<td></td>
<td>Invalidated, but later-referenced, cache contents</td>
<td>After returning from standby mode</td>
</tr>
<tr>
<td>Performance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>Leakage-measurement on-chip circuitry</td>
<td>Chip design &amp; manufacturing</td>
</tr>
</tbody>
</table>
Energy benefit & Performance cost linearly depend on the number of leaky cells cured ($N$)

$$\text{Energy Saving}(t) = N \times (P_{\text{leak}} \times t - E_{\text{lock}} - E_{\text{fetch}})$$

$$\text{Perf. Penalty} \leq N \times (T_M - T_c)$$

$N$: Number of leaky cells cured
$t$: Time duration spent in standby
$P_{\text{leak}}$: Avg. power saved per cured cache line
$E_{\text{lock}}$: Energy for locking leakage-safe value in the cache
$E_{\text{fetch}}$: Energy for fetching invalidated data if needed
$T_M$: Memory access time
$T_c$: Cache access time

Results for M32R processor:
- 0.18μ process, 200mW @ 50MHz
- Memory latency: 10 ns
- Cache latency: 1 ns
Effect of the Processor Used

M32R: 0.18u, 200mW @ 50 MHz
ARM920: 0.18u, 0.8mW / MHz

1 http://www.arm.com
Presented a *software* technique to suppress, during standby mode, leakage of ultra-leaky transistors

- No major hardware/circuit change required
- Only uses already-popular cache-control instructions
- Useful even for dynamic effects such as aging and temperature

**Results**

- Reduced leakage power in standby mode
- Salvage chips containing ULTs => higher yield for long-standby low-power applications

**Future work**

- Reduce leakage power, even in *active* mode, by matching cache contents with the less-leaky state of cache cells