EECS 598: Integrating Emerging Technologies with Computer Architecture

Lecture 9: Near-Threshold Computing

Instructor: Ron Dreslinski

Winter 2016
Near-Threshold Complications

- To maintain same robustness at low voltages SRAM cell sizes needs to be increased to compensate effects of process variation
- Increased size leads to higher energy consumption, and longer interconnects
System Design

Goal:
- Design a machine that can operate at multiple operating points, including:
  - a) Low Power Mode: Low frequencies (10MHz) while consuming only hundreds of microwatts
  - b) High Performance Mode: Scaled to 400MHz operation with as little impact on energy/runtime overhead as possible

Possible Choices:
- Traditional Filter Cache
  - Shield high dynamic access of the L1 with a small L0

- Near Threshold Filter Cache
  - Allow small filter caches to scale below 600mV for more energy efficiency
  - Cache cells need to be increased by ~2x to maintain robustness
  - Increased cell size reduces cache capacity significantly, leading to increased off-chip accesses
  - Consuming more energy
  - Increasing runtime

Alternative Designs to mitigate runtime/energy impacts at 400MHz
- Bypass Filter Cache
- Reconfigurable Energy-efficient Near Threshold (RENT) Cache
Outline

- Motivation
- *Baseline System*
- *Filter Caches*
  - Design
  - Comparison Study
  - Runtime Impacts
- Alternative Designs
  - Bypass Filter Cache
  - RENT Cache
  - Comparison Study
- Spec2000 Analysis
- Conclusions
Baseline Architecture

- Core Options
  - Assumes SubVt support
  - Voltage/Freq. Pairs
    - 1.2V -> 400MHz
    - 450mV -> 10MHz

- L1 Cache Options
  - Operating Voltages
    - 800mV to 1.2V

![Diagram of Core with L1 Cache Options]
Filter Cache

- Insert a small, low access energy cache (L0) in between the L1 and core
  - Smaller cache, leading to lower dynamic access energy
  - Low access energy shields access to L1, similar to low latency L1’s shielding high latency memory

- Positives
  - Provides lower energy via shielding
  - Only need to use near threshold cells on a small cache, retaining L1 cache capacity

- Negatives
  - Runtime increase, every miss in the L0 results in a 2-cycle access
Filter Cache Architecture

- Core Options
  - Same as Baseline

- L0 Options
  - SRAM Cells:
    - Traditional SRAM
      - 800mV – 1.2V
      - Normal Cell Size (1kB L0)
    - Near Threshold SRAM
      - 450mV – 1.2V
      - Increased Cell Size (512B L0)
  - Size of I/D same

- L1 Cache Changes
  - Size decreased to make room for L0
Simulation Environment

- **M5 full-system simulator**

- **Benchmark Suites:**
  - MIBench: Embedded Benchmarks
    - Each benchmark run to completion
  - Spec2000
    - SimPoints used to reduce simulation time

- **Energy Models:**
  - **Core**
    - Energy consumption based from ARM core
    - Scaled using Spice analysis for near-threshold energy consumption
  - **Cache**
    - Energy consumption computed from Spice Analysis of SRAM cells
    - Importance Sampling techniques used to determine cell sizes to retain yield
BitCount Comparisons
BitCount Comparisons (zoomed in)

- Filter Cache: Near Threshold SRAM
- Traditional Filter Cache: Standard SRAM
- Baseline

- Core
- Off Chip
- DL1 Dynamic
- DL1 Leakage
- DL0 Dynamic
- DL0 Leakage
- IL1 Dynamic
- IL1 Leakage
- IL0 Dynamic
- IL0 Leakage

Energy (pJ/Inst)

- 400 MHz
- 10 MHz
Filter Cache – Near Threshold SRAM
MiBench Benchmarks @ 10MHz

Average Bit Count
Basic Math
CJPEG
DJPEG
FFT
Patricia
QSort
Sha
String Search
Susan

Energy (pJ/Inst)

Core
Off Chip
DL1 Dynamic
DL1 Leakage
DL0 Dynamic
DL0 Leakage
IL1 Dynamic
IL1 Leakage
IL0 Dynamic
IL0 Leakage

Average Reduction In Energy:
Compared to 10MHz Traditional Filter: 36.7%
Compared to 10MHz Baseline: 83.5%

Benchmarks Differ:
Some more data cache critical – FFT, DJPEG, Dijkstra
Some more instruction cache critical – BitCount, Sha
MiBench Runtime Increase @ 400MHz

- Average
- Susan
- String Search
- Sha
- QSort
- Patricia
- FFT
- DJPEG
- Dijkstra
- CJPEG
- Basic Math
- Bit Count

Normalized Run Time @ 400MHz to Baseline

Filter Cache - 6kB L1, 1kB Filter
Outline

- Motivation
- Baseline System
- Filter Caches
  - Design
  - Comparison Study
  - Runtime Impacts
- Alternative Designs
  - Bypass Filter Cache
  - RENT Cache
  - Comparison Study
- Spec2000 Analysis
- Conclusions
Bypass Filter Cache

- **Bypass Mode**, skip the L0

- **Positives:**
  - Can skip latency/energy of L0

- **Negatives:**
  - Less total cache size available…
  - More off-chip accesses
    - Longer latencies
    - Energy hungry
Reconfigurable Energy-efficient Near Threshold (RENT) Cache

- **In L0 Mode:**
  - Accesses are serial: First En1 then En2
  - Misses swap data between ways using swap buffer

- **In Normal mode:**
  - En1 and En2 are accessed in parallel

- **Positives:**
  - Keep full cache capacity

- **Negatives:**
  - Complicated control logic (just different than WB Buffers)
MiBench Runtime Increase @ 400MHz

Normalized Run Time @ 400MHz to Baseline

- Filter Cache - 6kB L1, 1kB Filter
- Filter Cache - 6kB L1, Bypass Filter
- RENT Cache - 6kB Conventional, 1kB Nearthreshold
Outline

- Motivation
- Baseline System
- Filter Caches
  - Design
  - Comparison Study
  - Runtime Impacts
- Alternative Designs
  - Bypass Filter Cache
  - RENT Cache
  - Comparison Study
- Spec2000 Analysis
- Conclusions
GZIP (Spec2000) Benchmark

- Filter Cache (Near Threshold SRAM)
  - Core
  - Off Chip
  - DL1 Dynamic
  - DL1 Leakage
  - DL0 Dynamic
  - DL0 Leakage
  - IL1 Dynamic
  - IL1 Leakage
  - IL0 Dynamic
  - IL0 Leakage

- Traditional Filter Cache (Standard SRAM)

- Baseline

Energy (pJ/Inst) vs. Frequency (MHz)
Spec2000 Comparisons @ 10MHz

- Tend to have larger number of off-chip accesses
- Data Cache Accesses are more than Instruction Cache Accesses
Spec2000 Summary

- 10MHz Rent Cache Compared to:
  - 10MHz Traditional Filter - 57% Reduction in energy
  - 10MHz Baseline - 82% Reduction in energy
  - 400MHz Rent Cache – 77% Reduction in energy

- Rent Cache sees 4% increase in runtime at 400MHz
Conclusions

- **Goal:** Design a system to run in a high performance mode, and a low power mode. The low power mode should provide as much energy savings as possible with minimal impact on high performance mode.

- **Results:**
  - 10MHz **Low Power mode** consuming on average **77% less energy** to complete the same task than the 400MHz High performance mode
  - Only a **4% increase** on average in **runtime** in high performance mode for the Benchmark