ScalCore: Designing a Core for Voltage Scalability

Bhargava Gopireddy, Choungki Song, Josep Torrellas, Nam Sung Kim, Aditya Agrawal, Asit Mishra
University of Illinois, University of Wisconsin, Nvidia, Intel
Motivation

- Application characteristics vary dynamically

- Goal: **Single core design** that attains
  - High performance at nominal voltage (~0.9V)
  - High energy efficiency at low voltage (~0.5V)

A Voltage-Scalable Core
Observations

- SRAM vs Logic delay scaling
- Small increase in $V \rightarrow$ large improvement in delay

![Graph showing normalized delay vs $V_{dd}(V)$]

- SRAMDelay
- LogicDelay

$V_{min}$

~2x
ScalCore Idea

- Design a Voltage-Scalable core based on the two observations

![Graph showing frequency and voltage relationship]
ScalCore: A Voltage-Scalable Core

- Decouple $V_{dd}$ of logic and storage structures in the pipeline
  - Improve energy efficiency

- Raise $V_{dd}$ of storage structures a little
  - Reconfigure the pipeline to take advantage of faster storage structures
  - Further improve performance and reduce leakage energy !!

- Evaluated the proposed design under various scenarios
  - Reduce execution time by 31%, energy by 48% and ED by 60% over the state of the art

Truly Voltage Scalable Core !!
ScalCore Design

- **Goal**: Maximize the energy-efficiency at low-voltage (EEMode)
- **Constraint**: No impact on performance or energy at nominal voltage (HPMode)
- **Approach**: In EEMode,
  - Provide different $V_{dd}$s for storage and logic stages
    - Storage stage $\sim 2x$ faster than logic stage
  - Reconfigure pipeline in one of the two ways
    - Fuse storage stages in the pipeline (e.g. Accessing Register File)
    - Increase storage structure sizes
Fusing Two Pipeline Stages into One

HPMode

CLK

Logic Stage 1

Storage Stage 2a

Storage Stage 2b

Logic Stage 3

V_{nom}
Fusing Two Pipeline Stages into One

HPMode

CLK

Latch

Logic Stage 1

Latch

Storage Stage 2a

Latch

Storage Stage 2b

Latch

Logic Stage 3

Enable Flow-through

V_{nom}

V_{nom}

V_{nom}

V_{nom}

---

EEMode

CLK

Latch

Logic Stage 1

Latch

2a

Latch

2b

Latch

Logic Stage 3

Enable Flow-through

V_{logic}

V_{op}

V_{op}

V_{logic}
Increasing Size of Structures
ScalCore Pipeline

Main storage structures
## Pipeline Structure Changes

<table>
<thead>
<tr>
<th>Fuse Stages</th>
<th>Increase Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register File</td>
<td>Array access + Source drive</td>
</tr>
<tr>
<td></td>
<td>1.5X PRF</td>
</tr>
<tr>
<td>Allocation</td>
<td>Rename + Dispatch</td>
</tr>
<tr>
<td></td>
<td>---</td>
</tr>
<tr>
<td>Load Store Unit</td>
<td>Addr. generation + Memory disambiguation</td>
</tr>
<tr>
<td></td>
<td>1.5X Load/Store Queue and Store buffer</td>
</tr>
<tr>
<td>Reorder Buffer</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>1.5X ROB</td>
</tr>
<tr>
<td>Branch Prediction</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>---</td>
</tr>
</tbody>
</table>
Controlpath Changes

- Programmable counters for variable latencies
  - Execution schedules
  - Wakeup logic

- Programmable counters for variable sizes
  - List of free registers
  - ROB, Load/Store queue sizes
Circuit Issues

HPMode

EEMode

V_{op}

V_{nom}

V_{logic}

Logic Stage 1

Latch w/ Level Conv.

Storage Stage 2a

Latch

Storage Stage 2b

Latch

Logic Stage 3

Circuit Issues

Evaluation Methodology

- 16 OOO cores
  - 64K I-L1/D-L1, 1MB L2

- Voltages and frequencies
  - HPMode
    - $V_{\text{nom}} = 0.90 \, \text{V, } f=3.5\, \text{GHz}$
  - EEMode
    - $V_{\text{logic}} = 0.50 \, \text{V, } f=600\, \text{MHz}$
    - $V_{\text{op}} = 0.65 \, \text{V, } f=600\, \text{MHz}$
    - $V_{\text{min}} = 0.6 \, \text{V, } f=900\, \text{MHz}$
Configurations

- **Baseline:**
  - HPRef: Conventional HP (3.5GHz, 0.9V)
  - DVFS: Most aggressive (900MHz, 0.6V)

- **ScalCore:**
  - HPMode: HPRef with penalty (3.3GHz, 0.9V)
  - EEMode (600MHz, 0.5V, 0.65V)
    - Pipe2Vdd: Separate voltages only
    - SC: Fuse the two stages of RF, Allocate;
      Increase size ROB, LSQ structures
Iso-Power Comparison for Parallel Programs

Reduced execution time by 31%;
ED by 60% compared to DVFS
Dyn-SC reduces execution time by 31%, energy by 22% and ED by 46% compared to conventional DVFS
Also in the Paper

- ScalCore complexities and overheads
- Comparison with Intel Claremont
- Impact on different classes of applications
- More results
  - Unconstrained power budget
  - Intermediate design points of ScalCore
Conclusion

- Presented ScalCore, a core designed for voltage scalability
- Designed a voltage-scalable core by
  - Decoupling $V_{dd}$ of logic and storage structures in the pipeline
  - Raising $V_{dd}$ of storage structures a little
  - Reconfiguring the pipeline to take advantage of faster storage structures and improved performance, energy
ScalCore: Designing a Core for Voltage Scalability

Bhargava Gopireddy, Choungki Song, Josep Torrellas, Nam Sung Kim, Aditya Agrawal, Asit Mishra
University of Illinois, University of Wisconsin, Nvidia, Intel
Why not Big/Little?

- Heterogeneous cores on a chip ideally suited for different tasks 😊
- Fixed partitioning of cores 😞
- A fraction of chip unused 😞
- Migration overhead 😞
Configurations

- **Baseline:**
  - HPRef: Conventional HP (3.5GHz, **0.9V**)
  - DVFS: Most aggressive (900MHz, **0.6V**)

- **ScalCore:**
  - HPMode: HPRef with penalty (3.3GHz, **0.9V**)
  - EEMode (600MHz, **0.5V**, **0.65V**)
    - Pipe2Vdd: Separate voltages only
    - SCspeed: Fuse the two stages of RF, LSU, Allocate
    - SC: Fuse the two stages of RF, Allocate
      - Increase size ROB, LSQ structures