High-Performance Arithmetic Challenges: From Architectures to Circuits

Ram K. Krishnamurthy, Sanu K. Mathew, Shekhar Borkar
Microprocessor Research, Intel Labs
Intel Corporation, Hillsboro, OR, USA
ramk@ichips.intel.com

Prof. Vojin Oklobdzija
ACSEL Lab, Dept. of ECE
University of California, Davis, CA, USA
vojin@ece.ucdavis.edu
Outline

- Motivation
- Design choices for high-performance circuits
- SOI vs. Bulk devices: ALU design test-case
  - 64-bit ALUs in PD-SOI and Bulk CMOS
- Energy-efficient high-performance AGU/ALUs
  - 4GHz Sparse-tree AGU Design
  - 6.5-10GHz Integer ALU Design
- Summary
High-performance trends

- Frequency doubles every generation
- Performance-critical units
  - ALUs & AGUs
  - Register files, L0 caches

Single-cycle latency & throughput
64-bit ALUs in 0.18μm PD-SOI/Bulk CMOS: Design & Scaling Trends

[S. Mathew et al, ISSCC 2001]
[S. Mathew et al, JSSC, Nov 2001]
Design choices

- High performance devices:
  - Partially depleted Silicon-on-Insulator
  - Pros & Cons vs. bulk CMOS
  - Scaling trends

- High performance circuit design:
  - Sparse-tree semi-dynamic AGU
  - Single-rail dynamic ALU
PD-SOI Devices

- Body of devices not tied to $V_{cc}/V_{ss}$
- Body is isolated by buried oxide
- Floating Body!
History Effect in PD-SOI

- Delay = Function of switching history
  - Capacitive coupling from S/G/D
  - Impact Ionization, Diode conduction
  - Transient $V_{bs} \neq DC V_{bs}$

Complicates timing analysis
64-bit ALU architecture

Ideal test-bed for evaluating process technologies
High-performance Adders: Kogge Stone

Generate all carries:

- Full-blown binary tree $\Rightarrow$ energy-inefficient

$\#$ Carry-merge stages $= \log_2(N)$
64-bit Han-Carlson adder core

- Carry-merge done on even bitslices
- 50% fewer carry-merge gates vs. Kogge-Stone
- Extra logic stage generates odd carries
## Energy-efficient adder core

<table>
<thead>
<tr>
<th>Adder architecture</th>
<th>Energy/transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kogge-Stone</td>
<td>120pJ</td>
</tr>
<tr>
<td>Han-Carlson</td>
<td>68pJ</td>
</tr>
</tbody>
</table>

43% less energy/transition at iso-performance
Han Carlson carry-merge tree

- Single rail adder core
- CSG circuit generates dual-rail carry
Complementary signal gen.

- Domino-compatible Carry/Carry
- Permits a single-rail carry-merge tree design
- Not time-borrowable – Penalty absorbed by placing gate at $\Phi_2$ boundary
Partial sum generator

- Generates domino-compatible partial sum
- Placing the gate at $\Phi_1$ boundary mitigates output noise-glitches
ALU performance in bulk CMOS

64b Han-Carlson ALU simulation results

| ALU delay | 482ps |

0.18μm bulk CMOS, $V_{cc} = 1.5V$
Porting from bulk to PD-SOI

Direct port

SOI design

Design issues:
- Noise tolerance due to lowered $V_t$
- Min-delay timing-analysis

Bulk design

SOI favored redesign

SOI-optimal design

Motivation for redesign:
- Reduced SOI stack penalty
- Deeper stack design
- Stage reduction

Design choices:
- Architecture should favor deep stack design
- Avoid increase in fanouts
### 0.18μm Bulk & PD-SOI technologies

<table>
<thead>
<tr>
<th></th>
<th>$I_{off}(nA/\mu m)$</th>
<th>$I_{dsat}(\mu A/\mu m)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMOS-Bulk</td>
<td>3.3</td>
<td>1070</td>
</tr>
<tr>
<td>NMOS-SOI</td>
<td>3.3</td>
<td>1050</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$I_{off}(nA/\mu m)$</th>
<th>$I_{dsat}(\mu A/\mu m)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMOS-Bulk</td>
<td>0.7</td>
<td>445</td>
</tr>
<tr>
<td>PMOS-SOI</td>
<td>0.7</td>
<td>441</td>
</tr>
</tbody>
</table>

- Equal $I_{off}$ at DC $V_{bs}$
- SOI $I_{dsat}$ is 1-2% lower
History effect measurements in 0.18μm PD-SOI

These gates are used in the ALU design.

5-11% delay variation

Measurements agree with simulation results.
Direct port of Han-Carlson ALU to PD-SOI

0.18μm technology, $V_{cc}=1.5V$

<table>
<thead>
<tr>
<th>64b Han-Carlson ALU delay simulations</th>
<th>% Delay improvement over bulk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulk</td>
<td>482ps</td>
</tr>
<tr>
<td>Direct-port to SOI</td>
<td>403ps</td>
</tr>
</tbody>
</table>

- Adder core speedup = 14%
  - [Stasiak et al., ISSCC 2000] 21% speedup
### Speedup analysis

<table>
<thead>
<tr>
<th>Stage type</th>
<th>Speedup over bulk from direct port to 0.18μm PD-SOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static gates</td>
<td>12-15%</td>
</tr>
<tr>
<td>Dynamic gates</td>
<td></td>
</tr>
<tr>
<td>3:1 TG Mux</td>
<td>2-9%</td>
</tr>
<tr>
<td>5:1 TG Mux</td>
<td>20%</td>
</tr>
<tr>
<td>9:1 TG Mux</td>
<td>23%</td>
</tr>
</tbody>
</table>

- Diffusion dominated muxes ➔ Max. speedup
- Load dominated gates ➔ Speedup decreases
Motivation for PDSOI-optimal redesign

- Reduced stack penalty in SOI
- Deeper stack design \(\rightarrow\) Stage reduction
- ALU is amenable to such a redesign
  - Not true for all CPU critical paths
- SOI-optimal ALU architecture
  - Increasing stack depth must not increase fanouts
- A novel deep-stack sparse-tree ALU was developed
Sparse-tree adder core

- 50% reduced fanouts compared to Han-Carlson
- 7 gate stages (Two less than Han-Carlson)
Intermediate Carry Generator

- Generates 1 in 4 carries (C_3, C_7, C_{19} \ldots, C_{59})
- Non-critical path (ripple carry-select scheme)
- Fast carry selects between the conditional carries
Non-critical Sum Generator

- Non-critical path: ripple carry chain
- Reduced area, energy consumption, leakage
- Generate conditional sums for each bit
- 1 in 4 carry selects appropriate sum

\[ P_{i+3}, G_{i+3} \]
\[ P_{i+2}, G_{i+2} \]
\[ P_{i+1}, G_{i+1} \]
\[ P_i \]
Sparse-tree adder critical path

- Fast carry-merge path → Critical path
- Non-critical side-paths → Ripple-carry

Intermediate carry generator

Input: 2N → 2P → 4N → 2P → 3N → Fast carry-merge path

Sum generator: 3N → 2P → 2N → Sumout
PD-SOI optimal redesign in 0.18μm

<table>
<thead>
<tr>
<th>64b ALU delay simulations</th>
<th>Speedup over bulk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulk</td>
<td>482ps</td>
</tr>
<tr>
<td>Direct-port SOI</td>
<td>403ps</td>
</tr>
<tr>
<td>SOI-optimal redesign</td>
<td>380ps</td>
</tr>
</tbody>
</table>

Deeper stack redesign → additional 5% speedup
Margining for reverse-body bias in PD-SOI

- 400mV rvs. bias increases rise-delay by 10%
- Difficult to detect for large circuits
- 10% Margin required for all max-delay paths

Overall PD-SOI speedup reduces to 11%
Reducing reverse-bias penalty in dynamic SOI gates

- Point solution for dynamic designs
- Pre-charging stack node decreases penalty to 2%

Cost
5% increase in clock energy

Max-delay margin reduced to 2%
### 0.18μm ALU performance after margining

<table>
<thead>
<tr>
<th>64b ALU delay simulations</th>
<th>Speedup over bulk</th>
<th>Speedup after margining</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulk</td>
<td>482ps</td>
<td>-</td>
</tr>
<tr>
<td>Direct-port SOI</td>
<td>403ps</td>
<td>16%</td>
</tr>
<tr>
<td>SOI-Optimal redesign</td>
<td>380ps</td>
<td>21%</td>
</tr>
</tbody>
</table>

**Maximum PD-SOI speedup reduces to 19%**

0.18μm technology, $V_{cc}=1.5V$
Scaling to 0.13μm technologies

- Equal SOI & bulk $I_{OFF-DC}$

- MOSFET & impact ionization data obtained from 0.13μm bulk measurements

- SOI parasitic BJT/diode characteristics unchanged from 0.18μm fitting
Scaling ALU designs to 0.13µm technology

0.13µm technology, $V_{cc}=1.2V$

<table>
<thead>
<tr>
<th>64b ALU delay simulations</th>
<th>Speedup over bulk</th>
<th>Speedup after margining</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulk</td>
<td>351ps</td>
<td>-</td>
</tr>
<tr>
<td>Direct-port SOI</td>
<td>312ps</td>
<td>11%</td>
</tr>
<tr>
<td>SOI-Optimal redesign</td>
<td>286ps</td>
<td>18%</td>
</tr>
</tbody>
</table>

Maximum PD-SOI speedup reduces to 16%
SOI vs. bulk Summary

- 482ps energy-efficient dynamic 64b ALU in 0.18\(\mu\)m bulk
  - 310ps adder core

- Direct port to 0.18\(\mu\)m SOI \(\rightarrow\) 14% speedup
- SOI optimal redesign \(\rightarrow\) 19% speedup

- Floating body can get reverse-biased
  - Preconditioning reduces margin from 10% to 2%

- Scaling to 0.13\(\mu\)m decreases PD-SOI speedup
- Maximum PD-SOI speedup in 0.13\(\mu\)m falls to 16%
High-Performance Low Power Datapath design

Goal: Shift the E-D curve
A 4GHz 130nm Address Generation Unit with 32-bit Sparse-tree Adder Core

[S. Mathew et al, VLSI Symp. 2002],
[S. Mathew et al, JSSC May 2003]
- AGUs: performance and peak-current limiters
- High activity $\Rightarrow$ thermal hotspot
- Goal: high-performance energy-efficient design
AGU Architecture

- Single-cycle latency and throughput
- Effective Address = Base + Index*Scale + (Segment + Displacement)
- 2-phase address computation
- Index pre-scaled via 3-bit barrel shifter
- 3:2 compressor renders partial address:
  - Carry-save format
- Adder in pre-charge state
**AGU Operation: Phase 2**

- Carry-save to binary format conversion:
  - 2’s complement parallel 32-bit adder

```
Base → 32
Index → 32
Segment + Displacement → 32
clk

3:2 Compressor

3b shift

32 b adder
clk2
clk3

Effective Address → 32
```

38
Kogge-Stone Adder

- Critical path = PG+5+XOR = 7 gate stages
- Generate, Propagate fanout of 2,3
- Maximum interconnect spans 16b
Sparse-tree Adder Architecture

- Generate every 4th carry in parallel
- Side-path: 4-bit conditional sum generator
- 73% fewer carry-merge gates $\Rightarrow$ energy-efficient
Non-critical Sum Generator

- Non-critical path: ripple carry chain
- Reduced area, energy consumption, leakage
- Generate conditional sums for each bit
- Sparse-tree carry selects appropriate sum
Conditional Carry for Cin=0

- Carry-merge stage reduces to inverter
- Conditional carry \( C_i^0 = G_i \)
Optimized First-level Carry-merge

Conditional carry for Cin=1

- \( P_i \) & \( G_i \) correlated
- Conditional carry \( C^{#}_1 = P_i# \)

<table>
<thead>
<tr>
<th>( A_i )</th>
<th>( B_i )</th>
<th>( P_i )</th>
<th>( G_i )</th>
<th>( C^{#}_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Optimized Sum Generator

- Optimized non-critical path: 4 stages
Adder Core Critical Path

- Critical path: 7 gate stages ⇒ same as KS
- Sparse-tree: single-rail dynamic
- Exploit non-criticality of sum generator
- Convert to static logic ⇒ Semi-dynamic design
1st-level Carry-merge: Static Latch

- Holds state in pre-charge phase
- Prevents pre-charging of static stages
- Sum = Sum0 during pre-charge
- Mux output resolves during evaluation
Sparse-tree Architecture

- **Performance impact:** (20% speedup)
  - 33-50% reduced G/P fanouts
  - 80% reduced wiring complexity
  - 30% reduction in maximum interconnect

- **Power impact:** (56% reduction)
  - 73% fewer carry-merge gates
  - 50% reduction in average transistor size
- 20% speedup over Kogge-Stone
- 56% worst-case energy reduction
- Scales with activity factor
Semi-dynamic Design

- Static sum generators: low switching activity
- 71% lower average energy at 10% activity
Dual-$V_t$ Allocation

**130nm CMOS, 1.2V, 110°C Simulation**

<table>
<thead>
<tr>
<th></th>
<th>Low-$V_t$</th>
<th>Dual-$V_t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay</td>
<td>152ps</td>
<td>152ps</td>
</tr>
<tr>
<td>Switching Energy</td>
<td>36pJ</td>
<td>34pJ (-6%)</td>
</tr>
<tr>
<td>Leakage Energy</td>
<td>0.9pJ</td>
<td>0.4pJ (-56%)</td>
</tr>
</tbody>
</table>

- Exploit non-criticality of sidepaths
  - Use high-$V_t$ devices
- 0% performance penalty
- 56% reduction in active leakage energy
Scaling Performance

- Average transistor size = 3.5\(\mu m\)
- Reduces impact of increasing leakage
- 80% reduction in wiring complexity
- Reduces impact of wire resistance
- 33% delay scaling, 50% energy reduction

<table>
<thead>
<tr>
<th></th>
<th>130nm</th>
<th>100nm (-33%)</th>
<th>100nm (-50%)</th>
<th>100nm (-23%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay</td>
<td>152ps</td>
<td>102ps</td>
<td>18pJ</td>
<td>0.7pJ</td>
</tr>
<tr>
<td>Switching Energy</td>
<td>36pJ</td>
<td>102ps (-33%)</td>
<td>18pJ (-50%)</td>
<td>0.7pJ (-23%)</td>
</tr>
<tr>
<td>Leakage Energy</td>
<td>0.9pJ</td>
<td>102ps (-33%)</td>
<td>18pJ (-50%)</td>
<td>0.7pJ (-23%)</td>
</tr>
</tbody>
</table>
A 6.5GHz, 130nm Single-ended Dynamic ALU

[M. Anders et al, ISSCC 2002],
[S. Vangal et al, JSSC November 2002]
32-bit ALU/Scheduler Loop

- Performance-critical execution core loop
Han-Carlson ALU Organization

- Single-rail dynamic 9-stage low-Vt design

5:1 Mux Control

Propagate/Generate/Partial Sum (dynamic)

Carry merge 0 (static)
Carry merge 1 (dynamic)
Carry merge 2 (static)
Carry merge 3 (dynamic)
Carry merge 4 (static)
Carry merge 5 (CSG) / Sum

84um loopback bus

Sum

Sum#
Odd-bits CSG Sum Generation

- Final carry-merge CSG (dual-rail carry output)
  → pass-transistor sum XOR
Even-bits CSG Sum Generation

- Domino-compatible sum
- Dual-rail sum from single-ended g inputs
Die Micro-photograph

• 130nm 6-metal dual-Vt CMOS
• Scheduler:
  • 210μm x 210μm
• ALU:
  • 84μm x 336μm
Delay and Power Measurements

- 6.5GHz at 1.1V, 25°C
- Power: 120mW total, 15mW leakage
- Scalable to 10GHz at 1.7V, 25°C
Improvements Over Dual-rail Domino

- Leakage reduced by eliminating dual-rail logic
- Robustness not compromised
- CSG improves both area and performance

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>50%</td>
</tr>
<tr>
<td>Performance (Delay)</td>
<td>10%</td>
</tr>
<tr>
<td>Active Leakage</td>
<td>40%</td>
</tr>
<tr>
<td>Robustness</td>
<td>equal</td>
</tr>
</tbody>
</table>
Summary

- **4GHz AGU in 1.2V, 130nm technology**
- Sparse-tree adder architecture described
- 20% speedup and 56% energy reduction
- Semi-dynamic design:
  - Energy scales with switching activity
- Dual-$V_t$ non-critical paths:
  - Low active leakage energy
- **6.5GHz ALU and scheduler loop at 1.1V, 25ºC**
  - Scalable to 10GHz at 1.7V, 25ºC