FabScalar RISC-V

Rangeen Basu Roy Chowdhury
Anil Kumar Kannepalli
Eric Rotenberg
FabScalar

• Generates synthesizable RTL (Verilog) for arbitrary superscalar cores within a canonical superscalar template

• Vision
  
  o Accelerate development of single-ISA heterogeneous multi-core processors comprised of many microarchitecturally-diverse core types
  
  o Superscalar technology accessible to everyone (not just few elite teams at Goliath processor companies)
  
  o Research framework
    • High-fidelity cycle time, power, and area estimation of whole cores
    • Proof-of-concept of new microarchitectures
    • Technology-driven computer architecture research
    • FPGA and ASIC prototyping

Outline

• FabScalar Toolset
  o Approach
  o Other Tools
• FabScalar Outreach
  o User data
• FabScalar Based Chips
• FabScalar Evolution
• FabScalar RISC-V
  o Microarchitecture
  o Performance
FabScalar Approach

- **Canonical Superscalar Template**
  - Defines canonical pipeline stages and their interfaces

- **Canonical Pipeline Stage Library (CPSL)**
  - Provides many different designs for each canonical pipeline stage
  - Diversity is focused along three key dimensions:
    - *Superscalar Complexity*: Superscalar width, Sizes of stage-specific structures for extracting instruction-level parallelism (ILP)
    - *Sub-pipelining*: Pipeline depth of a canonical stage
    - *Stage-specific design choices*: e.g., different speculation alternatives, recovery alternatives, etc.

- **Core Generator**
  - References CPSL and Template to compose a core of desired configuration
CPSL

Fetch

Rename

Issue

Canonical Superscalar Template

Fetch

Decode

Rename

Issue

Dispatch

Execute

Writeback

Retire

App. 1

core configuration

Core Generator

synthesizeable RTL of customized core
CPSL

Fetch

Rename

Issue

Canonical Superscalar Template

Fetch

Decode

Rename

App. 2

core configuration

Core

Generator

Issue

Register Read

Execute

Writeback

Retire

synthesizable RTL

of customized core

6/30/2015
Tools Offered by FabScalar

- **FabScalar**
  - Template, CPSL, and Core Generator (just described)

- **FabMem**
  - Support for highly-ported RAMs and CAMs
    - Estimation tool
    - Memory compiler (auto-generate layouts that pass LVS and DRC)
  - Targets FreePDK 45nm

- **FabFPGA**
  - A version of FabScalar for FPGA prototyping
FabScalar Outreach

U.S. Universities | Int'l Universities | Industry Labs | Countries
---|---|---|---
UC Santa Cruz (CA) | Ghent University (Belgium) | Global Foundries | Australia
UC San Diego (CA) | Simon Fraser University (Canada) | Intel Labs (2 sites) | Belgium
Northwestern University (IL) | Tsinghua University (China) | Synopsys | Brazil
UIUC (IL) | TU-Darmstadt (Germany) | Calvista | Canada
Harvard University (MA) | Alexander Tech. Educ. Institute of Thessaloniki (Greece) | IBM | China
SLSU (NC) | IIT Delhi (India) | | Denmark
Cornell University (NY) | IIT Madras (India) | | France
Univ. of Rochester (NY) | Politecnico di Milano (Italy) | | Germany
Drexel University (PA) | Min University (Japan) | | Greece
UT Austin (TX) | National University of Singapore (Singapore) | | India
UT Dallas (TX) | KAIST (South Korea) | | Iran
Univ. of Virginia (VA) | Barcelona Supercomputing Center (Spain) | | Israel
Virginia Tech (VA) | Cambridge University (UK) | | Italy
UW Madison (WI) | ABV-JITM (India) | | Japan
SUNY Binghamton (NY) | Bilkent University (Turkey) | | Korea
Utah State University (UT) | DA-IICT (India) | | Norway
Columbia University (NY) | Karlsruhe Institute of Technology (Germany) | | South Korea
Stanford University (CA) | Wuhan University (China) | | Spain
Univ. of Maine (ME) | Chalmers University (Sweden) | | Sweden
USC (CA) | SouthEast University (China) | | Turkey
UC Riverside (CA) | Univ. of Tehran (Iran) | | USA
CMU (PA) | Tel Aviv University (Israel) | | USA
Georgia Tech (GA) | Chinese Academy of Sciences (China) | | USA
UC Irvine (CA) | Yonsei University (South Korea) | | USA
Univ. of Michigan (MI) | University of Augsburg (Germany) | | USA
Duke University (NC) | Federal University of Mato Grosso do Sul (Brazil) | | USA
Arizona State University (AZ) | Henan University (China) | | USA
NYU Polytechnic (NY) | State Key Laboratory of High Perf. Computing (China) | | USA
Univ. of Central Florida (FL) | Zhejiang University (China) | | USA
Univ. of Chicago (IL) | Univ. of British Columbia (Canada) | | USA
Penn State University (PA) | IIT Bombay (India) | | USA
Univ. of Minnesota (MN) | IIT (India) | | USA
Stony Brook University (NY) | Univ. of Waterloo (Canada) | | USA
Univ. of Victoria (Canada) | Univ. of Campinas (Brazil) | | USA
NTNU - Norwegian Univ. of Science & Technology (Norway) | Federal University of Santa Catarina (Brazil) | | USA
University of Tokyo (Japan) | ENS Rennes / IRISA (France) | | USA
Osaka University (Japan) | Politecnico di Torino (Italy) | | USA
Islamic Azad University (Iran) | Technical University of Denmark (Denmark) | | USA
The University of New South Wales (Australia) | Pontificia Universidade Catolica do Rio grande do Sul / PUCRS (Brazil) | | USA

(a) Affiliations.

(b) New members over time.

(c) Google group activity.

User data through October 2014.

6/30/2015
FabScalar Based Chips at NC State

• H3 (“Heterogeneity in 3D”)
  o Two cores with different microarchitectures
  o Hardware support for fast thread migration

FabScalar Based Chips at NC State

- AnyCore
  - One core with reconfigurable microarchitecture
  - Adapts to workload to improve efficiency

**AnyCore Zoomed-in**

<table>
<thead>
<tr>
<th>Adaptive microarchitecture feature</th>
<th>Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>fetch/dispatch width (instructions/cycle)</td>
<td>1, 2, 3, 4</td>
</tr>
<tr>
<td>issue width (instructions/cycle)</td>
<td>3, 4, 5</td>
</tr>
<tr>
<td>physical register file &amp; active list</td>
<td>64, 96, 128</td>
</tr>
<tr>
<td>load and store queues (each)</td>
<td>16, 32</td>
</tr>
<tr>
<td>issue queue</td>
<td>16, 32, 48, 64</td>
</tr>
</tbody>
</table>
Non-NCSU FabScalar Based Chips

- Mei University, Japan fabricated a FabScalar MIPS32 based chip
  - Coprocessor 0
  - L1 Caches
  - AMBA based system bus
# FabScalar Evolution

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
</table>
| CPSL approach requires making changes in each stage variant, or modifying scripts that generate CPSL. | **Superset Core**: A single parameterized System Verilog description.  
- Structure sizes already parameterized  
- Parameterized widths and sub-pipelining |
| No multi-core / SoC support | **FabCache, FabBus**: Prof. T. Sasaki @ Mei Univ.  
- Generate diverse cache hierarchies [7]  
- Generate buses for multi-core and accelerator support [8] (AMBA protocol) |
| PISA (SimpleScalar) ISA:  
- No privileged ISA.  
- No software ecosystem (old gcc, no linux) | **FabScalar-MIPS ports**:  
- FabScalar-MIPS32 + Co-processor 0 (MMU) + Linux (Prof. T. Sasaki @ Mei Univ.)  
- FabScalar-MIPS64 + Co-processor 1 (FPU) |
| MIPS ISA:  
- Proprietary ISA: Concerned about releasing FabScalar-MIPS Superset Core  
- OOO compatibility: Has frustrating ISA features (delay slots, conditional moves) | **FabScalar-RISC-V**:  
- Open ISA  
- No frustrating features w.r.t. OOO implementation  
- Privileged ISA  
- Software ecosystem |
FabScalar Superset Core

```
define FETCH_FOUR_WIDE
define ISSUE_THREE_WIDE
define ISSUE_TWO_DEEP
define RR_TWO_DEEP
```
`define FETCH_TWO_WIDE
`define ISSUE_TWO_WIDE
`define SIZE_PRF 128

`define SIZE_BTB  2048
`define SIZE_ACTIVE_LIST  128
`define SIZE_IQ  64
Changes for RISC-V port

• Starting point was PISA Superset Core (64-bit instructions, 32-bit address and data)
  o RISC-V 64-bit has 32-bit instructions and 64-bit data
Changes for RISC-V port

- Starting point was PISA Superset Core (64-bit instructions, 32-bit address and data)
  - RISC-V 64-bit has 32-bit instructions and 64-bit data
Changes for RISC-V port

- RISC-V very similar to PISA (no delay slots, no conditional moves, etc.)
  - RISC-V specific changes mostly in Fetch, Decode, and Execute
Changes for RISC-V port

- 64-bit for both INT and FP makes adding FP straightforward
  - Unified Physical Register File
  - Unified Issue Queue – FP ALU is just another function unit

- Additional committed state
- FP ALU just another function unit

32 additional logical registers
FabScalar RISC-V Test Harness

- MMU and CSRs are currently implemented in C++
  - Accessed through System Verilog DPI
  - Will be replaced with RTL implementations
- The C++ part communicates with the Front End Server through HTIF
Basic Performance Evaluation, 4-wide Superscalar Configuration

Array Reduction

for(i=0;i<20000;i++){  
    temp = a[i];  
    sum = sum + 3;  
    sum = sum + 4;  
    sum = sum + 5;  
    sum1 = sum1 + temp;  
    sum2 = sum2 + temp;  
}

Assembly

1016c: lw  a5,0(a6)  
10170: addi a2,a2,3  
10174: addi a2,a2,4  
10178: addi a2,a2,5  
1017c: addw a3,a3,a5  
10180: addw a4,a4,a5  
10184: addi a6,a6,4  
10188: bne a6,a1,1016c <main+0x34>

IPC = 3.7
FabScalar RISC-V Offerings

• FabScalar RISC-V: An open-source tool
  o Parameterized OOO superscalar implementation of RV64G
  o Complete with uncore components
  o Verification infrastructure
• CAD flow for easy synthesis and place-and-route
• A C++ timing simulator for performance studies
• FabScalar RISC-V will be available on GitHub in Fall
  o Users can commit improvements
  o Users can “cherry-pick” specific changes and bug fixes
Future Work

- Implement privileged ISA to boot Linux on FabScalar cores
- Untether FabScalar cores (Do not use HTIF)
- Add testcases to stress different design features
- Port FabFPGA to RISC-V
References


