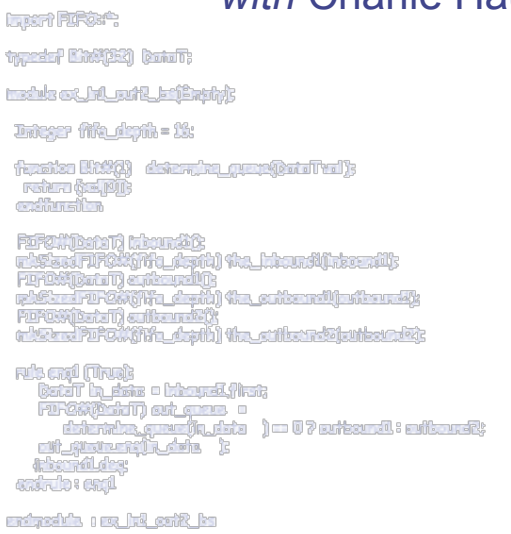
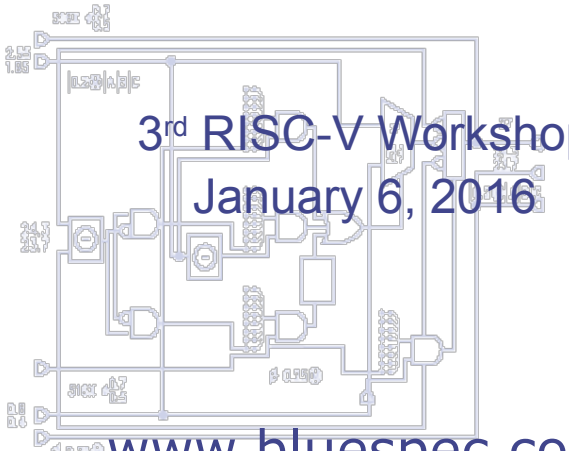
Bluespec's "RISC-V Factory"

(components and development/verification environment)

Rishiyur S. Nikhil

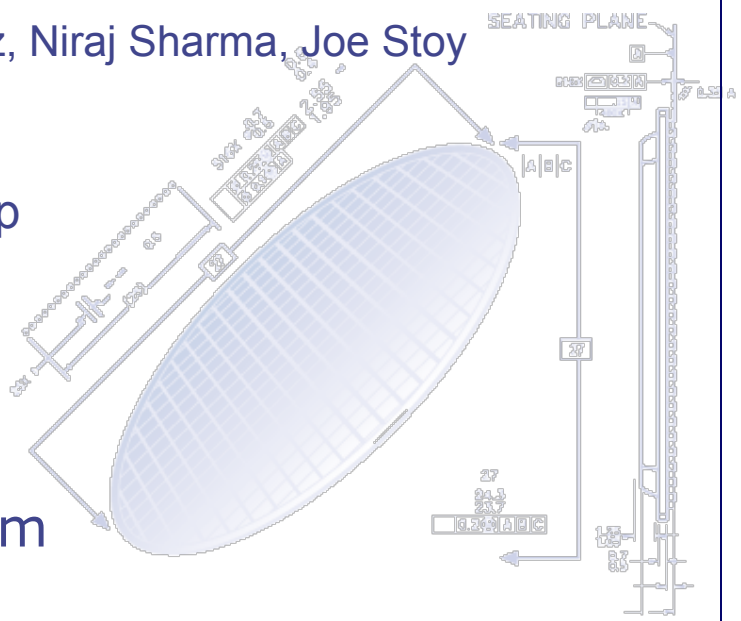
with Charlie Hauck, Darius Rad, Julie Schwartz, Niraj Sharma, Joe Stoy





3rd RISC-V Workshop

January 6, 2016



www.bluespec.com

Introduction

Bluespec's "RISC-V Factory" is aimed at organizations that wish to create and own their own custom RISC-V-based CPUs or SoCs without the usual learning curve, startup costs and ownership costs.

- Focus on your own "delta" (whether CPU, SoC, add-on, extension, software, ...)
- Starting with a "complete" Linux-ready baseline that is easily customizable
- With strong hooks for debugging and verification

A suite of substitutable components:

- RISC-V CPUs ("Flute", a basic in-order pipeline now; more variants later in 2016)
  - RV32/64I, M, F, D; Privileged Spec v1.7 (May 9, 2015)
  - Customizable for your extensions
- "Uncore" components: MMUs, L1 caches, L2 caches
- SoC components: Interconnect fabric, memory controller, DMA engine, a few devices

with easy expandability and substitutability to incorporate your own components.

A complete development/verification environment:

- Direct GDB control
- Continuous Tandem Verification™
  - Against host executable spec (e.g., Spike, Bluespec Cissr, or your own)
  - Against synthesizable executable spec (e.g., Bluespec BluROCS)
- With Bluespec components, runs Linux and apps out of the box
  - Tethered (via Bluespec's PCIe connectivity) and untethered (UART, Ethernet)

(already in use with selected partners; general availability end 2016 Q1)

Example engagement

DRAPER

DOVER project with Draper Inherently Secure Processing Hive
cf. talk by André Dehon at this workshop

- Baseline: Bluespec Flute RISC-V CPU, MMUs, Caches, SoC
- DOVER Extensions:
  - Tags on all architectural registers, all memory data
    - Need support in caches, interconnect fabric, mem,...
  - "PUMP" tag-processing unit integrated in CPU pipeline
  - Extra CSRs for managing PUMP
  - Interconnect fabric → Trusted and untrusted fabrics
  - System SW support for tag traps, secure tag management

By starting with Bluespec's baseline and development environment, Draper is going from start-of-project in Sept 2015 to first synthesized, untethered, demo in Feb 2016.

bluespec

Component: Flute RISC-V CPU

- RV32/64 I, M, F, D
- Privileged Spec v1.7 (May 9, 2015)
- Boots Linux

- 6-stage in-order pipeline
  - Branch prediction
  - Concurrent integer, floating point, memory pipes

- Clean, well-defined interface including:
  - Interface for memory system (IMem and DMem)
  - Interface for direct GDB control
    - Standard GDB, including ELF load, run, single-step, breakpoint, examine, ...
    - "Standard" GDB extensions for MMU and cache manipulation, etc.
  - Interface for Tandem Verification

- Hooks for optional tagged data

- "Elastic" (latency-insensitive) pipeline for easy modification
  - Custom functionality
  - Pipeline depth
  - etc.

Substitute your own CPU with this interface to take advantage of Bluespec Factory's GDB and tandem verification infrastructure

bluespec

Components: MMUs, Caches

- Separate IMem and DMem components

- MMUs:
  - Support Sv32 (RV32) and Sv39 (RV64)
  - Support 'R' (referenced) and 'D' (dirty) bits on PTEs
  - Support superpages
  - In-order now; will support out-of-order (e.g., TLB hit under miss) soon

- Caches:
  - Support out-of-order accesses (hits under misses)
  - Support speculative ops + later commit/discard on retirement

- FENCE.I and SFENCE.VM support in caches and MMUs

- Support for memory-mapped I/O (uncached, possibly side-effecting LOADs, ...)
- Optional support for tagged memory

"Ideal Memory" system (in simulation only)
- "Ideal": single-cycle response
  - But with full support for speculation, retire commit/discard

- For analyzing, debugging CPU pipeline issues in isolation from memory system limits

- Identical interface (plug-compatible) with the actual memory system
- Optional tagged memory

Components: SoC

Interconnect Fabric

- Crossbar, parameterized by # initiators, # targets, routing function
- Fully pipelined for requests and responses
- Supports out-of-order responses from targets
- Supports burst reads and writes

Memory controller

- Mediates between fabric and low-level memory interface
  - Ready for certain FPGA boards; customizable for other interfaces
- Pipelined
- Supports fabric's burst reads and writes

DMA engine

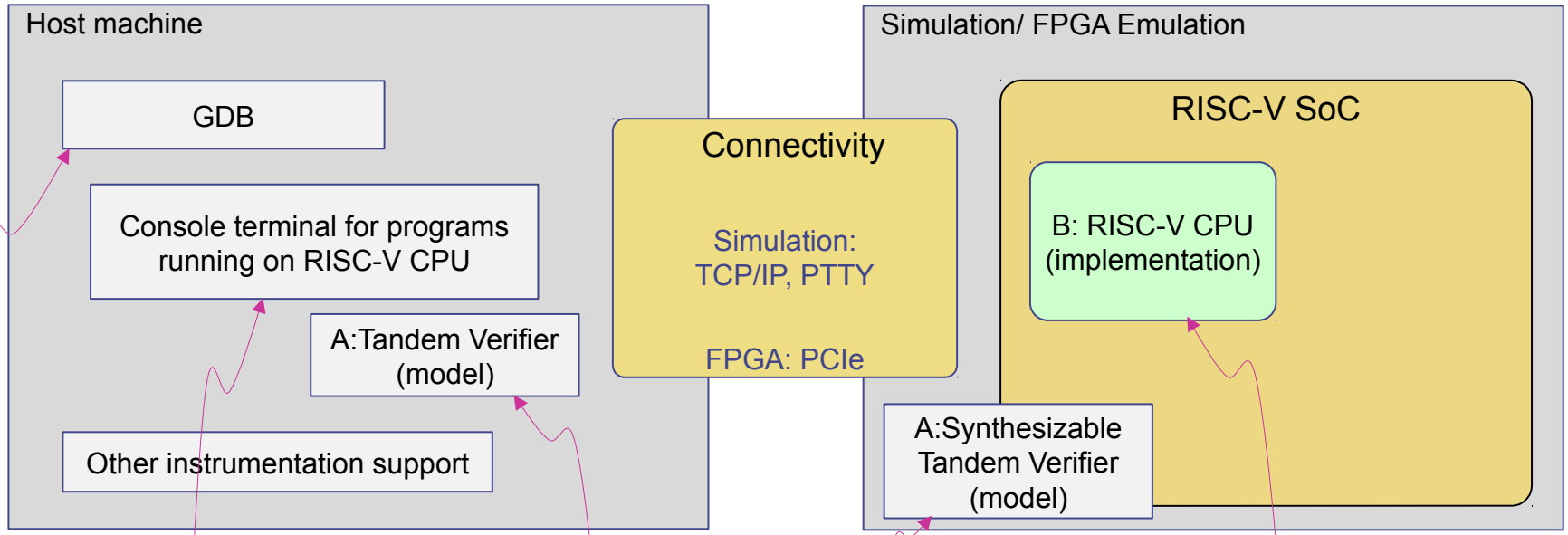
- Multi-channel, concurrent, pipelined copy between targets on fabric

Devices

- UART (for console, etc.)
- (Expected Q1): Flash for booting, Ethernet

Development/Verification Environment

Identical environment whether simulation or FPGA execution



Standard RISC-V GDB in "remote debug" mode

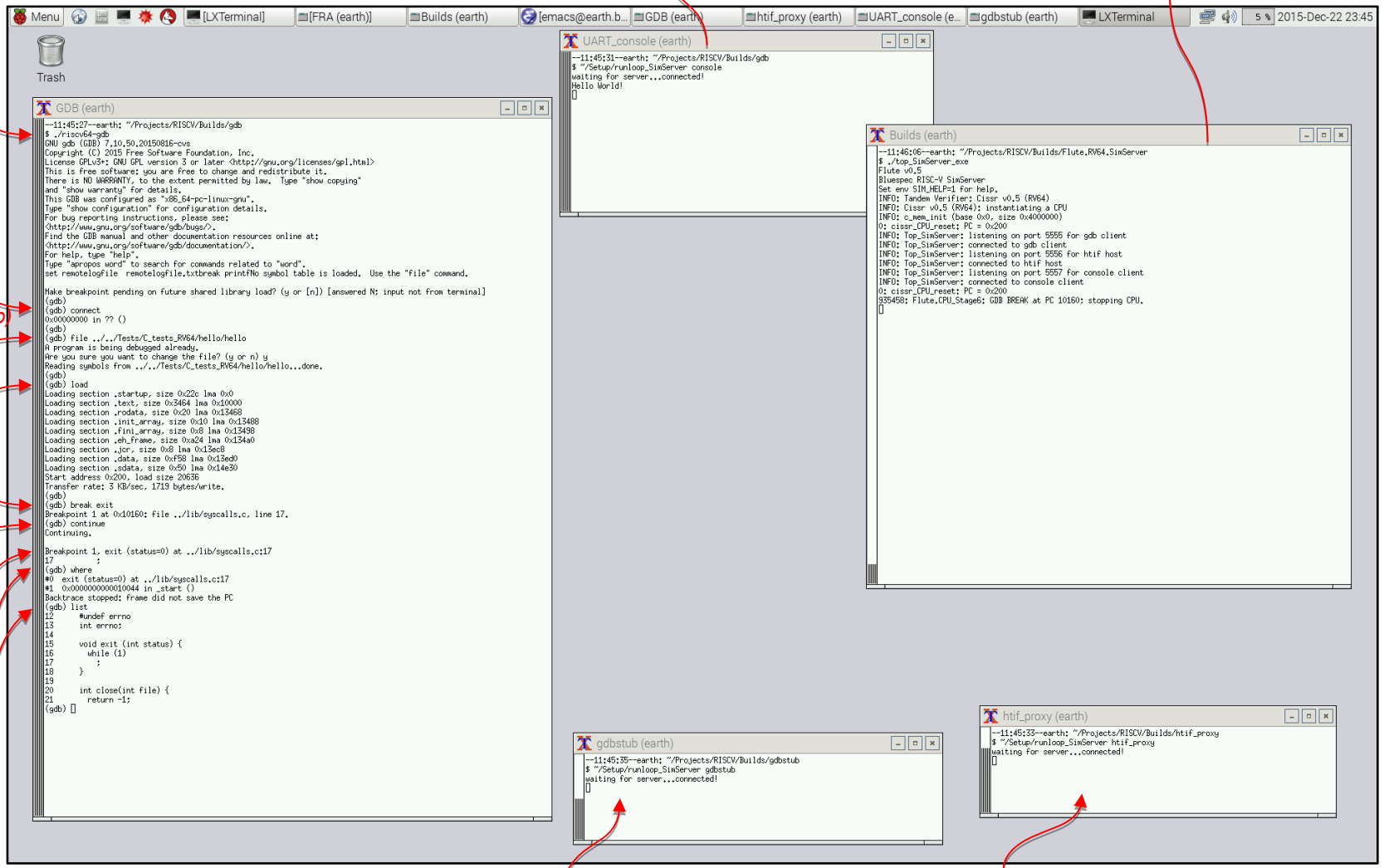
HTIF or UART

A: Verification Model
Spike
Bluespec Cissr
Bluespec BluROCS (synthesizable)
Your own RISC-V CPU model (e.g., SRI L3 Formal model [Mundkur])

B: Implementation	*Sim*	*FPGA*
Spike	✓	✗
Bluespec Cissr	✓	✗
Flute	✓	✓
Your own RISC-V CPU	✓	?

Note: can verify implementations against models, but also new models against old models, etc.

Screenshot: Development/Verification Environment



Summary

Bluespec's "RISC-V Factory" is aimed at organizations that wish to create and own their own custom RISC-V-based CPUs or SoCs without the usual learning curve, startup costs and ownership costs.

- Focus on your own "delta" (whether CPU, SoC, add-on, extension, software, ...)
- Starting with a "complete" Linux-ready baseline that is easily customizable
- With strong hooks for debugging and verification

A suite of substitutable components:

- RISC-V CPUs ("Flute", a basic in-order pipeline now; more variants later in 2016)
  - RV32/64I, M, F, D; Privileged Spec v1.7 (May 9, 2015)
  - Customizable for your extensions
- "Uncore" components: MMUs, L1 caches, L2 caches
- SoC components: Interconnect fabric, memory controller, DMA engine, a few devices

with easy expandability and substitutability to incorporate your own components.

A complete development/verification environment:

- Direct GDB control
- Continuous Tandem Verification™
  - Against host executable spec (e.g., Spike, Bluespec Cissr, or your own)
  - Against synthesizable executable spec (e.g., Bluespec BluROCS)
- With Bluespec components, runs Linux and apps out of the box
  - Tethered (via Bluespec's PCIe connectivity) and untethered (UART, Ethernet)

(already in use with selected partners; general availability end 2016 Q1)

bluespec

End