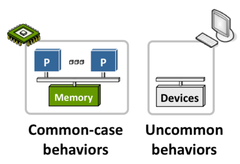
The Open-Source ProtoFlex Simulator

Eric S. Chung, Michael K. Papamichael*, James C. Hoe, Ken Mai, Babak Falsafi
{echung, jhoe, kenmai, babak}@ece.cmu.edu, *papamix@cs.cmu.edu
http://www.ece.cmu.edu/~protoflex

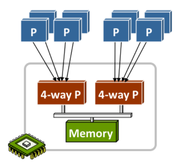
Computer Architecture Lab at Carnegie Mellon

ProtoFlex: FPGA-Accelerated Full-System Multiprocessor Simulation

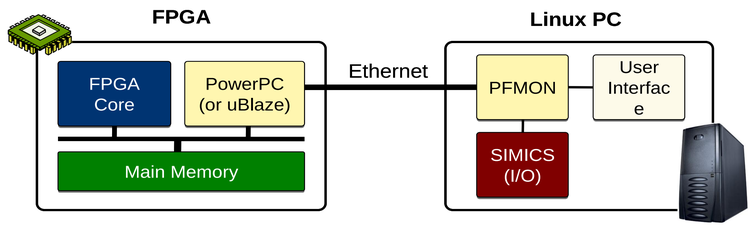
- **History**
  - Project started (circa 2007) to build scalable, full-system multiprocessor simulators using FPGAs
- **Key Features**
  - Functional simulator for N-way UltraSPARC III server (~50-90 MIPS)
  - Using hybrid simulation, runs real server apps + Solaris OS
  - Employs multithreading to virtualize # CPUs per FPGA core



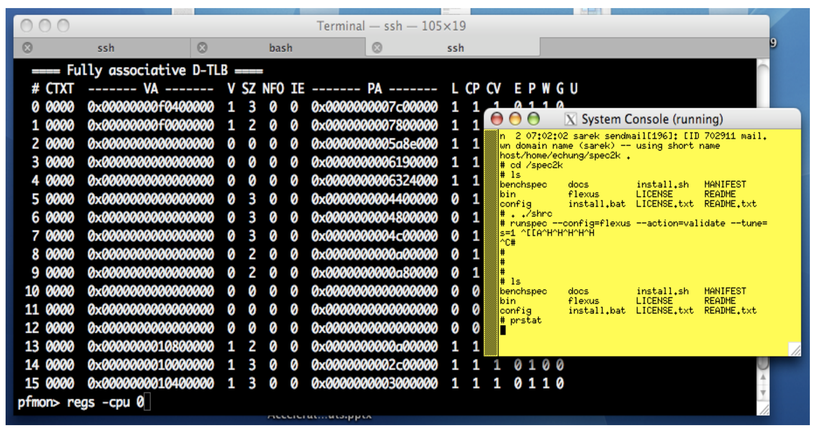
Hybrid Simulation



Virtualization



- **User perceives familiar SW-like UltraSPARC III simulator**
  - Software user interface similar to Simics
  - Applications load directly from Simics checkpoints
  - Standard simulation features: state viewing, scripting, single-stepping,[1] checkpointing, terminal, profiling/monitoring

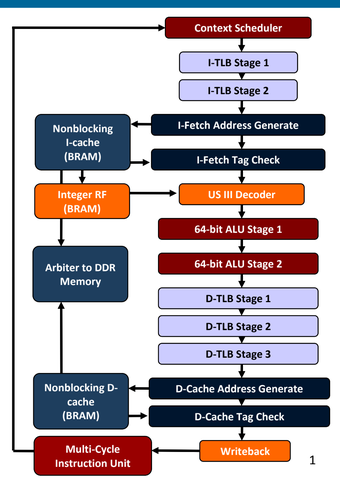


ProtoFlex (GNU GPL Release)

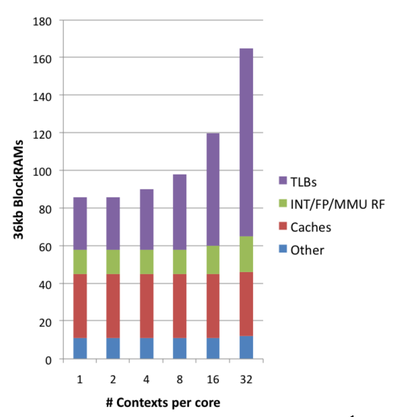
- **Why open source?**
  - Demonstration of FPGAs as viable architecture research vehicle
  - Facilitate adoption of **hybrid simulation** & **host multithreading**
  - Encourage building on top of our work
- **What are we releasing?**
  - Bluespec source HDL, Verilog and pre-generated netlists for SPARCV9 CPU model + interfaces
  - XUPV5 Reference Design for EDK 10.1
  - Virtutech Simics plug-ins for hybrid simulation
  - Top-level SW controller, user command-line interface
  - Documentation through online wiki

BlueSPARC Core and Statistics

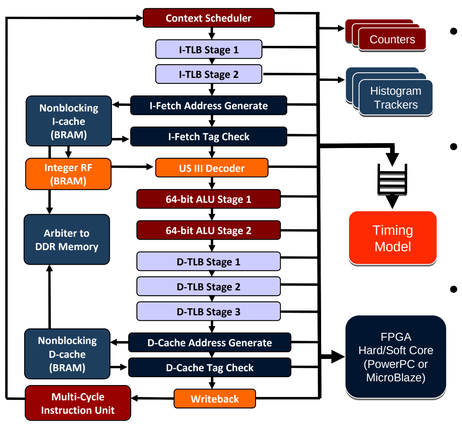
- **ISA Specifications**
  - 64-bit SPARCV9 ISA + US III extensions
  - 8 register windows, 4 global register files
  - 512-entry D-TLB, 128 I-TLB
- **Implementation**
  - **14-stage, multi-threaded pipeline, switch context on each cycle**
  - On Virtex-5, XST~148MHz, Placed & routed @ 100MHz
  - Parameterized non-blocking caches
  - FP + rare MMU instructions are SW-emulated by nearby uBlaze
  - 100% mirrors Virtutech Simics model



- **Runs 100MHz on V5**
  - Synthesizes up to 148MHz using standard tools (ISE XST)
- **Logic usage**
  - 23.5 KLUTs **(11.3% LX330T)**
- **BRAM usage**
  - 120 BRAMs for 16-context configuration **(37% LX330T)**
- **Future optimizations**
  - Paging structures to SRAM or DRAM can reduce BRAM by significant amount
  - Will release in future updates

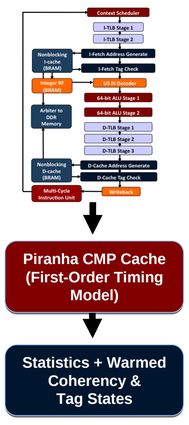


ProtoFlex Instrumentation and Applications

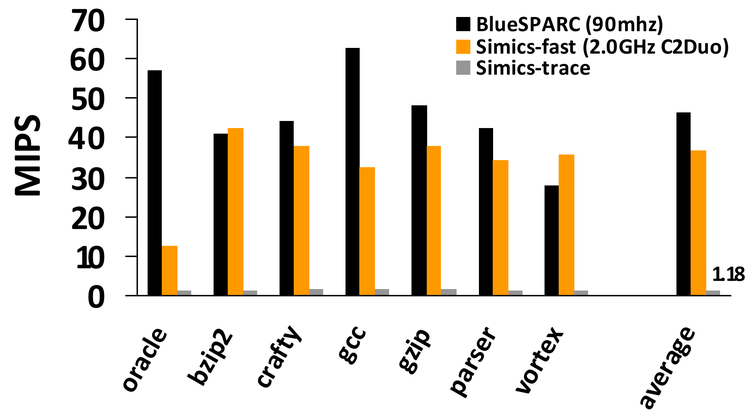


- **Add passive monitors**
  - Counters, histograms
  - Roll your own
- **Trace-based simulation**
  - Collect dynamic traces
  - Feed traces to functional-first timing model
- **Sampled Program Monitoring**
  - Use micro-blaze (or PPC) to monitor core/memory state
  - Unintrusive profiling w/o changes to target SW

- **Examples**
  - Functional-first CMP cache coherency model for first-order timing models and functional warming [TRETS'09]
  - Real-time stack trace profiling
  - CMP interconnect model (in progress)
  - Realistic CPU traffic generators (in progress)
- **… running real 16-CPU server workloads**
  - Oracle TPC-C, IBM DB/2 TPC-C, TPC-H, SPEC2K



Performance on BEE2 Platform



Hardware Requirements

Linux PC

Ethernet

FPGA Board

PFMON + Simics ⟷ BlueSPARC

- **Why XUPv5?**
  - Inexpensive (~$750), easily accessible
  - Standard tool flows (EDK, ISE)
  - Reference design portable to other platforms – just drop in our 'pcores'

- **Supporting other platforms**
  - Future ports to BEE3 & Xilinx Accelerated Computing Platform (ACP)
  - Plan to release with future updates

Bluespec Language & Flow

- **We use Bluespec System Verilog (high-level, synthesizable HDL)**
  - 4-8 weeks learning curve for normal HDL users
  - Once learned, easier to read/modify than conventional RTL
  - Requires BSV compiler (free for academics)
  - Paper in *MEMOCODE'09* describes BSV coding/validation of core
- **Sample code:**

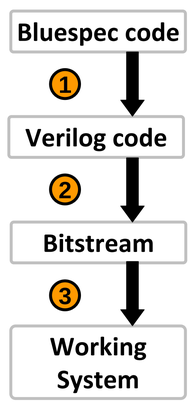
```
rule split_ALU_pipeline (True);
  ...
  p1 = piperegs[DECODE];
  piperegs[ALU1] <= doALUStage1(p1, alu_ifc);
  p2 = piperegs[ALU1];
  piperegs[ALU2] <= doALUStage2(p2, alu_ifc);
  ...
endrule
```

2-stage ALU

```
rule merged_ALU_pipeline (True);
  ...
  p1 = piperegs[DECODE];
  p_tmp1 = doALUStage1(p1, alu_ifc);
  p_tmp2 = doALUStage2(p_tmp1, alu_ifc);
  piperegs[ALU] <= p_tmp2;
  ...
endrule
```

1-stage ALU

1. **Bluespec → Verilog**
   - Bluespec compiler
   - **~30 minutes**
2. **Verilog → Bitstream**
   - Xilinx EDK
   - **~ 3 hours**
3. **Bitstream → Working System**
   - Stream mem. image over ethernet
   - **~ 5 minutes** (for 512MB image)



Other simulator features

- **Changing Core Parameters**
  - Number of CPU contexts
  - Cache sizes
  - Merge/split pipeline stages
  - Enable/disable modules for profiling & debugging
  - Clock frequency (tested @ 10 MHz – 100 MHz)
  - Set optimal LUTRAM size (16 = V2P, 64 = V5)
  - Choose LUTRAMs or BRAMs for any CPU state
- **System Parameters**
  - UDP or TCP/IP (for PFMON-to-FPGA communication)
  - XUPv5, BEE2

Thank you!

- **Topics & References**
  - **Hybrid Simulation and FPGA Core Multithreading**
    A Complexity-Effective Architecture for Accelerating Full-System Multiprocessor Simulations Using FPGAs [FPGA'08]
  - **Functional-First, First-Order Timing Model for a CMP cache**
    ProtoFlex: Towards Scalable, Full-System Multiprocessor Simulations Using FPGAs [TRETS'09]
  - **FPGA Core Design & Validation Using Flight Data Recorder**
    Implementing a High-performance Multithreaded Microprocessor: A Case Study in High-level Design and Validation [MEMOCODE'09]

Acknowledgements

We would like to thank our colleagues in the RAMP and TRUSS projects. Our work in this area has been supported in part by NSF, IBM, Intel, Sun, Xilinx, and Microsoft.