RISC-V Software and Firmware Development in the Cloud Using OpenPiton+Ariane on Amazon F1

Jonathan Balkind and David Wentzlaff
Princeton University

openpiton.org
pulp-platform.org
The DECADES Project

• Software Defined Hardware (SDH)
  • Runtime-reconfigurable hardware to accelerate data-intensive applications
    • Machine learning and data science
    • Graph analytics and sparse linear algebra

• DECADES: heterogeneous tiled architecture
  • Combination of core, accelerator, and intelligent storage tiles
  • Princeton/Columbia collaboration led by PIs Margaret Martonosi, David Wentzlaff, Luca Carloni

• Our tools are open-source!
  • https://decades.cs.princeton.edu/
OpenPiton+Ariane Overview

• Collaboration between Princeton University and ETH Zürich

• Goal is to build a permissively licensed (BSD + Apache/Solderpad), Linux capable manycore research platform based on RISC-V
  • Based on mature, extensible designs
  • Booted SMP Linux in <6 months
  • The world's first open-source, SMP Linux-booting, RISC-V manycore

• Ariane
  • RV64GC Core (with extensions)
  • Linux capable

• OpenPiton
  • Manycore research platform
  • Distributed cache coherence and NoC
Ariane RV64GC Core

- Application class processor
  - Written in SystemVerilog
- Linux Capable
  - Tightly integrated D$ and I$
  - M, S and U privilege modes
  - TLB, SV39
  - Hardware PTW
- Optimized for performance
  - Frequency: 1.5 GHz (22 FDX)
  - Area: ~ 175 kGE
  - Critical path: ~ 25 logic levels
- Designed for extensibility
OpenPiton

• Open source (SPARC/RISC-V/...) manycore
• Written in Verilog RTL
• P-Mesh coherence scales to ½ billion cores
• Configurable core, uncore
• Simulation in VCS, ModelSim, Incisive, Verilator, Icarus, Riviera*
• Includes synthesis and back-end flow
• ASIC & FPGA verified
• ASIC power and energy fully characterized [HPCA 2018]
• Runs full stack multi-user Debian Linux
• Used for Architecture, Programming Language, Compilers, Operating Systems, Security, EDA research
OpenPiton+Ariane Tile
OpenPiton+Ariane Cache Modifications

- New write-through cache subsystem with invalidations and the TRI interface
- LR/SC in L1.5 cache
- Fetch-and-op in L2 cache
OpenPiton+Ariane Platform Support

- Bootrom auto-generated with device tree from configuration
- RISC-V Debug
  - OpenOCD + GDB
  - Bootloading
- CLINT
- PLIC
  - lowRISC rv_plic
## Configurability Options

<table>
<thead>
<tr>
<th>Component</th>
<th>Configurability Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores (per chip)</td>
<td>Up to 65,536</td>
</tr>
<tr>
<td>Cores (per system)</td>
<td>Up to 500 million</td>
</tr>
<tr>
<td>Core Type</td>
<td>OpenSPARC T1</td>
</tr>
<tr>
<td></td>
<td>Ariane 64 bit RISC-V</td>
</tr>
<tr>
<td>Threads per Core</td>
<td>1/2/4</td>
</tr>
<tr>
<td>Floating-Point Unit</td>
<td>FP64, FP32</td>
</tr>
<tr>
<td></td>
<td>FP64, FP32, FP16, FP8, BFLOAT16</td>
</tr>
<tr>
<td>TLBs</td>
<td>8/16/32/64 entries</td>
</tr>
<tr>
<td></td>
<td>Number of entries (16 entries)</td>
</tr>
<tr>
<td>L1 I-Cache</td>
<td>Number of Sets, Ways (16kB, 4-way)</td>
</tr>
<tr>
<td>L1 D-Cache</td>
<td>Number of Sets, Ways (8kB, 4-way)</td>
</tr>
<tr>
<td>L1.5 Cache</td>
<td>Number of Sets, Ways (8kB, 4-way)</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>Number of Sets, Ways (64kB, 4-way)</td>
</tr>
<tr>
<td>Intra-chip Topologies</td>
<td>2D Mesh, Crossbar</td>
</tr>
<tr>
<td>Inter-chip Topologies</td>
<td>2D Mesh, 3D Mesh, Crossbar, Butterfly Network</td>
</tr>
<tr>
<td>Bootloading</td>
<td>SD/SDHC Card, UART, RISC-V JTAG Debug</td>
</tr>
</tbody>
</table>
FPGA Prototyping Platforms

- **Digilent Genesys2**
  - $999 ($600 academic)
  - 1-2 cores at 66MHz

- **Xilinx VC707**
  - $3500
  - 1-4 cores at 60MHz

- **Digilent Nexys Video**
  - $500 ($250 academic)
  - 1 core at 30MHz

- **Xilinx VCU118, BittWare XUPP3R**
  - $7000-8000
  - >100MHz

- **Amazon AWS F1**
  - Rent by the hour
  - 1-15 cores
OpenPiton+Ariane on F1

• We want to make OpenPiton available for everyone to use
  • With or without FPGAs
  • With or without recompiling hardware
• Amazon F1 enables cloud rental of FPGAs
  • Can rebuild in the cloud
• AWS repository of F1 bitfiles as "Amazon Global FPGA Images" (AGFIs)
  • Pre-built bitfiles great "real hardware" for software development/testing
• Instructions for OpenPiton AGFI in GitHub README.md
  • agfi-0d87a634f93fe7c83
OpenPiton on F1 - Synthesis

• **Command:** `protosyn -b f1 --uart-dmw ddr -c ariane`
• ~2-3 hours
• Generates design checkpoint (.dcp)
• Postprocessing on Amazon ~1-1.5 hours

• 1 core runs 125MHz
• 12 cores runs 62.5MHz, consume ~70% LUTs
• More information in GitHub readme
OpenPiton on F1 - Design

Generated bitfile

- `piton_aws`
  - `uart`
  - `piton_aws_uart`
  - AXI-Lite
  - AXI4 DMA interface
  - AXI4 memory interface

- `piton_aws_xbar`
  - LEDs, switches, clocks, reset

System

OpenPiton

AWS-provided
Boot Process

1. Zero Stage Bootloader (ZSBL)
   - Executes from ZSBL boot ROM
   - Copies FSBL from SD to memory

2. First Stage Bootloader (BBL)
   - Sets up trap table
   - Starts SMP

3. Linux
   - Page table setup
   - Driver loading (from device tree)
   - Environment preparation

4. init (Busybox) - start shell
Getting Running

1. Load FPGA image from open AWS image repository:
   `fpga-load-local-image -S 0 -I agfi-0d87a634f93fe7c83`

2. Open a serial connection:
   `./uart &`

3. Write OS image into FPGA memory:
   `./dma_os bbl_linux.bin`

4. Reset FPGA:
   `./fpga-reset`

5. Go!
Demo
Building Ariane SDK

• Clone our git repository:
  https://github.com/pulp-platform/ariane-sdk.git

• git submodule update --init --recursive

• If toolchain already installed set:
  • export RISCV=/path/to/install/riscv/toolchain
  • Otherwise Makefile will install the right toolchain

• make all
Components

• riscv-gnu-toolchain - GCC

• riscv-pk - Contains modified BBL

• buildroot - Upstream buildroot system
  • Makefiles and patches that automate building a bootable Linux environment

• rootfs - Overlay for rootfs
  • Using initramfs
  • Directory structure overlays rootfs on top
  • Used to include executables and other files into the image

• configs - Custom configuration
Customisation

- Buildroot automates most of the build process
  - Slightly patched Kernel
  - Driver fixes and custom drivers
- Buildroot wraps around that:
  - cd buildroot
  - make linux-menuconfig
  - make linux-savedefconfig
  - Install permanently into SDK
  - cp output/build/linux- */defconfig
    ../configs/linux-defconfig

- (Most) Packages from busybox
  - Lightweight rewrites of GNU applications
- Buildroot wraps around that:
  - cd buildroot
  - make busybox-menuconfig
  - Install permanently into SDK
  - cp output/build/busybox- */.config
    ../configs/busybox.config
Next Steps

• We're looking for assistance!
• Moving from BBL to OpenSBI
• Interested in coreboot, U-Boot for bootloading
• Distro bring-up
  • Debian chroot works, systemd boot in progress
  • Would like to see Fedora and others
• Testing parallel software scalability
This material is based on research sponsored by the NSF under Grants No. CNS-1823222, CCF-1217553, CCF1453112, CCF-1823032, and CCF-1438980, AFOSR under Grant No. FA9550-14-1-0148, Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) under agreement No. FA8650-18-2-7846, FA8650-18-2-7852, and FA8650-18-2-7862 and DARPA under Grants No. N66001-14-1-4040 and HR0011-13-2-0005. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA), the NSF, AFOSR, or the U.S. Government.
QUESTIONS?

@OpenPiton
http://openpiton.org

@pulp_platform
http://pulp-platform.org