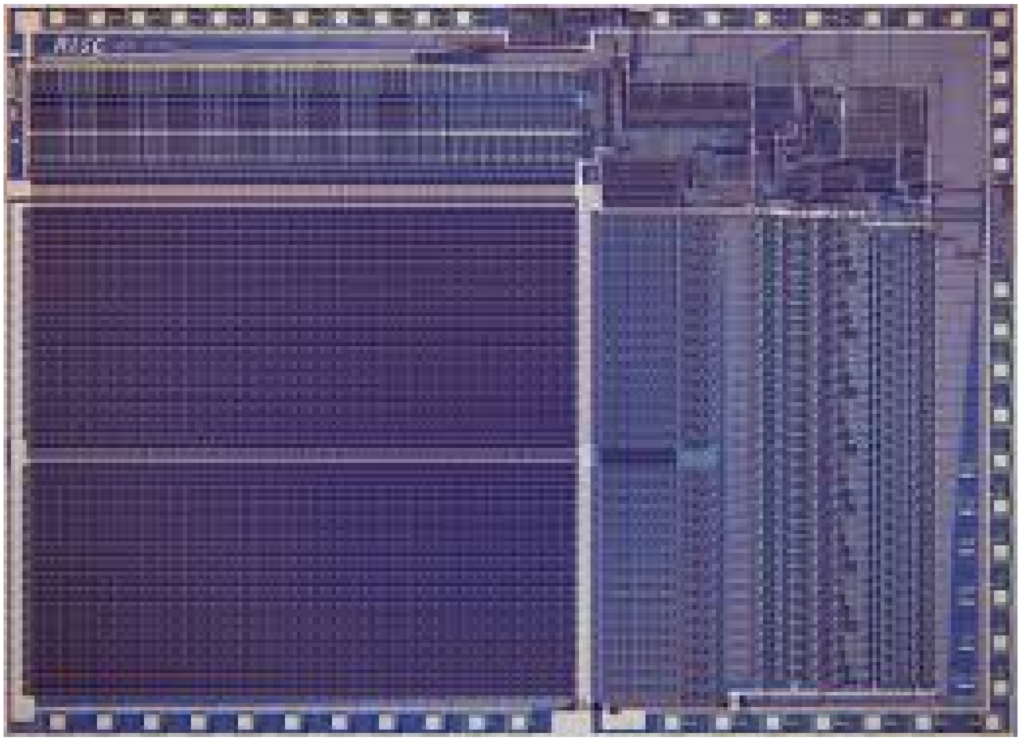
Raven: A CHISEL designed 28nm RISC-V Vector Processor with Integrated Switched-Capacitor DC-DC Converters & Adaptive Clocking

Yunsup Lee, Brian Zimmer, Andrew Waterman, Alberto Puggelli, Jaehwa Kwak, Ruzica Jevtic, Ben Keller, Stevo Bailey, Milovan Blagojevic, Pi-Feng Chiu, Henry Cook, Rimas Avizienis, Brian Richards, Elad Alon, Borivoje Nikolic, Krste Asanovic, Peter Ateshian*

University of California, Berkeley
***UCB Alumni, Naval Postgraduate School**

Take away: CHISEL = Agile Hardware Development Methodology



BLUF: RISC-V compared to ARM Cortex A5

10% higher in DMIPS/MHz, 49% more area-efficient

Our Physical Design Flow – A novel EDP

CHISEL Source Code/Scala
CHISEL Compiler
RTL Code (Verilog)
Synthesis
Place-and-Route
Gate-level Netlist
Signed-Off Design
Formality
Formal Verification
PrimeTime/StarRC
Static Timing Analysis
VCS Post-PNR
Gate-level Simulation
Hierarchical SYN & PNR
UPF-based MV SYN & PNR

What’s CHISEL?

(Constructing Hardware In a Scala Embedded Language).

CHISEL is a hardware construction language embedded in the high-level programming language Scala. Why? CHISEL enables very rapid SoC design iterations with less work.

A separate Chisel tutorial document provides a gentle introduction to using Chisel, and should be read first. This manual provides a comprehensive overview and specification of the Chisel language, which is really only a set of special class definitions, predefined objects, and usage conventions within Scala. When you write a Chisel program you are actually writing a Scala program. You are parameterizing Verilog/VHDL.
In this manual, we presume that you already understand the basics of Scala. If you are unfamiliar with Scala, we recommend you consult one of the excellent Scala books ([3], [2]).

References

[1] Bachrach, J., Vo, H., Richards, B., Lee, Y., Waterman, A., Avižienis, Wawrzynek, J., Asanović Chisel: Constructing Hardware in a Scala Embedded Language in DAC ’12.

[2] Odersky, M., Spoon, L., Venners, B. Programming in Scala by Artima.
Chisel: Constructing Hardware in an Scala Embedded Language
https://chisel.eecs.berkeley.edu/2.2.0/manual.html[2/4/2015 9:14:28 AM]

[3] Payne, A., Wampler, D. Programming Scala by O’Reilly books

Motivation

▪ Energy efficiency constrains everything

- SoCs are designed with an increasing number of voltage domains for better power management

- Dynamic voltage & frequency scaling (DVFS)

- Maximizes energy efficiency meets performance constraints

• Off-chip conversion

✖ Few voltage domains

✖ Costly off-chip components

✖ Slow mode transitions

• **On-chip conversion**

★ Many domains
★ No off-chip components
★ Fast transitions

Off-chip conversion example

iPhone 6/7

Credits: iFixit

Voltage Regulators $

Primary Power
Management IC PMIC $$

Secondary Power
Management IC PMIC $

Raven Project Goals

Build a microprocessor that is:

- Fine-grained DVFS
- High conversion efficiency

Energy-efficient

- Entirely on-chip converter
- Low area overhead

Low-cost

Talk Outline

- Motivation/Raven Project Goals
- On-Chip Switched-Capacitor DC-DC Converters
- Raven3 Chip Architecture
- Raven3 Implementation *Rapid CHISEL Iteration- novel EDP*
- Raven3 Evaluation
- RISC-V Chip Building at UC Berkeley
- Summary

For more details on the Switched-Capacitor DC-DC Converters, please take a
look at HC23 tutorial “Fully Integrated Switched Capacitor DC-DC Conversion” by Elad Alon

Switched-Capacitor (SC) DC-DC Converters SC DC-DCC

- Partition capacitor and switches into many “unit cells” for better *analog* modularity
- Keeps the custom design modular
- Makes it easier to floorplan SC DC-DC converter

Traditional Approach: Interleaved Switching

- Switch one unit cell at a time to smooth out voltage ripple
- Pros

- Voltage ripple at the output is suppressed

- Great for digital designs with fixed frequency clocks

- Cons

- Each unit cell charge shares with other unit cells

- Causes an efficiency loss beyond typical switching losses

Raven’s Approach: Simultaneous Switching

- Switch all unit cells simultaneously when V_{out} reaches a lower bound V_{ref}
- Add an adaptive clock generator so that clock frequency tracks the voltage ripple
- Pros
  - Simplifies the design
  - No charge sharing losses
  - Better energy efficiency
- Cons
  - Need to deal with big ripple on voltage output

Self-Adjusting Clock Generator

- Replica tracks critical path with voltage ripple
- Controller quantizes clock edge

(Tunable Replica Path)

DLL 2GHz input, 16 phases

Controller Select closest edge

Tunable Replica Path Vout

Adaptive system clock *Vout*

Reconfigurable SC Converters for DVFS

- Simple lower bound control

Lower-bound Controller Operational Waveform

Raven3 Chip Architecture

- RISC-V is a new, open, and completely free general purpose instruction set architecture (ISA) developed at UC Berkeley starting in 2010
- RISC-V is simple and a clean-slate design

- The base (enough to boot Linux and run modern software stack) has less than 50 instructions

- RISC-V is modular and has been designed to be flexible and extensible

- Better integrate accelerators with host cores

- RISC-V software stack

- GNU tools (GCC/Binutils/glibc/newlib/GDB), LLVM/Clang, Linux, Yocto (OpenJDK, Python, Scala)

- Checkout http://riscv.org for more details

Rocket Scalar Core

PC IF ID EX MEM To Vector Accelerator

- 64-bit 5-stage single-issue in-order pipeline
- Design minimizes impact of long clock-to-output delays of SRAMs
- 64-entry BTB, 256-entry BHT, 2-entry RAS
- MMU supports page-based virtual memory
- IEEE 754-2008-compliant FPU
- Supports SP, DP Fused-Multiply-Add (FMA) with HW support for sub-normals

WB

ARM Cortex-A5 vs. RISC-V Rocket

Category

ARM Cortex-A5

RISC-V Rocket

ISA

32-bit ARM v7

64-bit RISC-V v2 Architecture (32, 64, 128)

Single-Issue In-Order

Single-Issue In-Order 5-stage

Performance

1.57 DMIPS/MHz

1.72 DMIPS/MHz

Process

TSMC 40GPLUS

TSMC 40GPLUS

Area w/o Caches

0.27 mm2

0.14 mm2

Area with 16K Caches

0.53 mm2

0.39 mm2

Area Efficiency

2.96 DMIPS/MHz/mm2

4.41 DMIPS/MHz/mm2

Frequency

>1GHz >1GHz

Dynamic Power

<0.08 mW/MHz

0.034 mW/MHz

- PPA reporting conditions
- 85% utilization, use Dhrystone for benchmark, frequency/power at TT
0.9V 25C, all regular Vt transistors
Overall Result:

10% higher in DMIPS/MHz,
49% more area-efficient

Five 28nm & Six 45nm RISC-V Chips Taped Out So Far

Raven
Raven-1
Raven-2
Raven-3
Raven-3.5
EOS14
EOS16
EOS18
EOS20
EOS22
EOS24

Agile Hardware Development Methodology

C++

FPGA

ASIC Flow

Tape-in

Tape-out

Big Chip

Tape-out

- “Tape-in”: Designs that could be taped out

- LVS clean & DRC sane

- Pass RTL/gate-level simulation and timing

- Fully scripted CHISEL ASIC flow

- RTL change to chip <1day

- Get early feedback

- Automatic nightly regressions

- Identify source of subtle bugs

- Check longer programs on FPGA

- Iterate quickly on RTL with using the C++ emulator (Checkout chisel.eecs.berkeley.edu for more details)

Summary

- 28nm Raven3 processor features:

- Fine-grained, wide-range DVFS (20ns, 0.45-1V)
- Entirely on-chip voltage conversion
- High system efficiency (>80%)
- Extreme energy efficiency (34/26 GFLOPS/W)

- Key enablers

- RISC-V, a simple, yet powerful ISA (free and open!)
- Agile development, build hardware like software
- Chisel, lets designers do more things with less effort

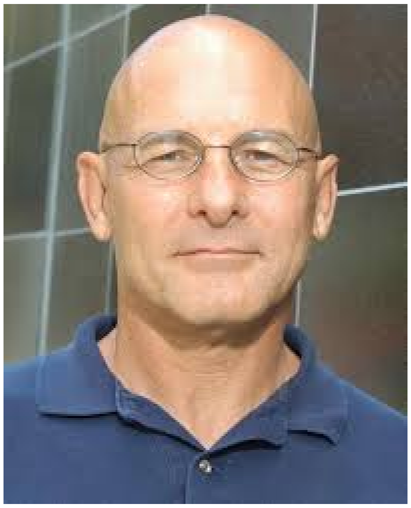
- RISC-V core generators and software tools open-sourced at http://riscv.org
- Energy-efficient, low-cost on-chip DC/DC converters are buildable!

Acknowledgements

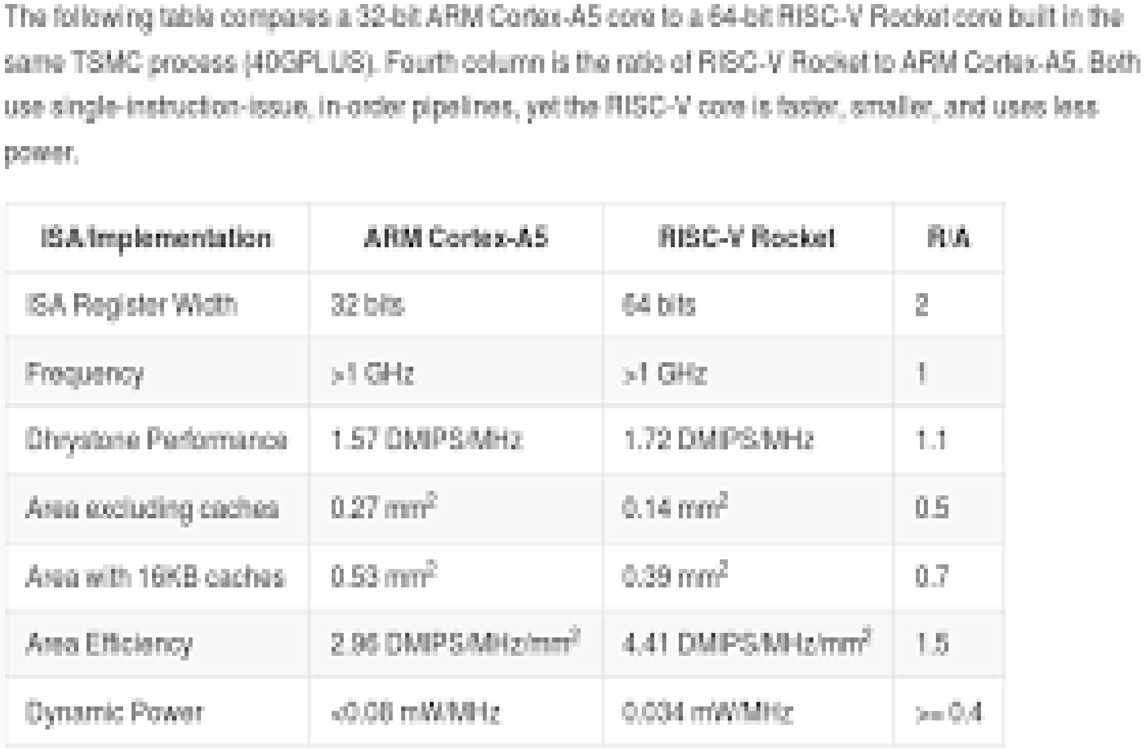
- Funding: BWRC members,
ASPIRE members,
DARPA PERFECT Award Number
HR0011-12-2-0016,

RISC-V

RISC-V



David Patterson



The following table compares a 32-bit ARM Cortex-A5 core to a 64-bit RISC-V Rocket core built in the same TSMC process (40GPLUS). Fourth column is the ratio of RISC-V Rocket to ARM Cortex-A5. Both use single-instruction-issue, in-order pipelines, yet the RISC-V core is faster, smaller, and uses less power.

ISA/Implementation	ARM Cortex-A5	RISC-V Rocket	R/A
ISA Register Width	32 bits	64 bits	2
Frequency	>1 GHz	>1 GHz	1
Dhrystone Performance	1.57 DMIPS/MHz	1.72 DMIPS/MHz	1.1
Area excluding caches	0.27 mm^2	0.14 mm^2	0.5
Area with 16KB caches	0.53 mm^2	0.39 mm^2	0.7
Area Efficiency	2.96 DMIPS/MHz/mm^2	4.41 DMIPS/MHz/mm^2	1.5
Dynamic Power	<0.08 mW/MHz	0.034 mW/MHz	>= 0.4

Intel ARO, AMD, GRC, Marie Curie FP7,
NSF GRFP, NVIDIA Fellowship

- Fabrication donation by STMicroelectronics
- Tom Burd, James Dunn, Olivier Thomas, & Andrei Vladimirescu

Glossary

- DVFS: Dynamic Voltage and Frequency Scaling
- SC: Switched Capacitor
- DLL: Delay-Locked Loop
- LDO: Low-Dropout Regulator
- FDSOI: Fully Depleted Silicon-on-Insulator
- FMA: Fused-Multiply-Add
- BTB: Branch Target Buffer
- BHT: Branch History Table
- RAS: Return Address Stack