The RISC-V Instruction Set

Andrew Waterman, Yunsup Lee, Rimas Avizienis, Henry Cook,
David Patterson, Krste Asanovic

www.riscv.org

Why a new ISA?

- Provide a ***realistic*** but ***simple*** ISA that captures important details of commercial general-purpose ISA designs and is suitable for hardware implementation.
- Provide a ***completely open*** ISA around which a community can grow.
- Avoid "over-architecting" for a particular microarchitectural style or implementation technology, but which allows efficient implementation in any of these.

RISC-V Base User-Level ISA

31–27	26–22	21–17	16–12	11–10	9–7	6–0	
rd	rs1	rs2	funct10			opcode	R-type
rd	rs1	rs2	rs3	funct5		opcode	R4-type
rd	rs1	imm[4:0]	imm[11:5]		funct3	opcode	I-type
imm[4:0]	rs1	rs2	imm[11:5]		funct3	opcode	B-type
rd	LUI-imm[19:0]					opcode	L-type
jump offset [24:0]						opcode	J-type

- Straightforward 32-bit instruction encoding
- Consistent register specifier locations
- Supports compressed encodings and extended-length instructions via ISA extensions

RISC-V Base User-Level ISA

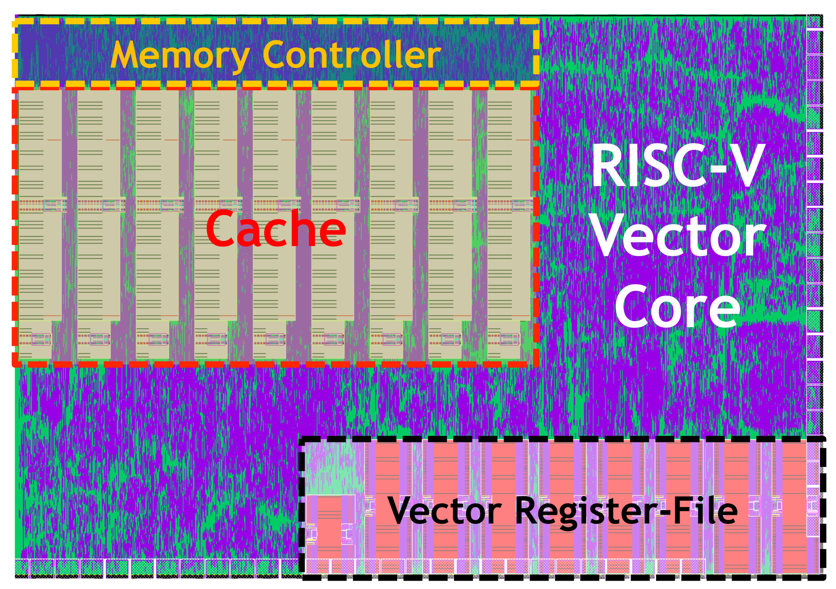
inst[4:2] / inst[6:5]	000	001	010	011	100	101	110	111 (> 32)
00	LOAD	LOAD-FP	*custom-0*	*custom-1*	OP-IMM	AUIPC	OP-IMM-32	
01	STORE	STORE-FP	AMO	MISC-MEM	OP	LUI	OP-32	
10	MADD	MSUB	NMSUB	NMADD	OP-FP	*reserved*	*custom-2/rv128*	
11	BRANCH	JALR	J	JAL	*reserved*	SYSTEM	*custom-3/rv128*	

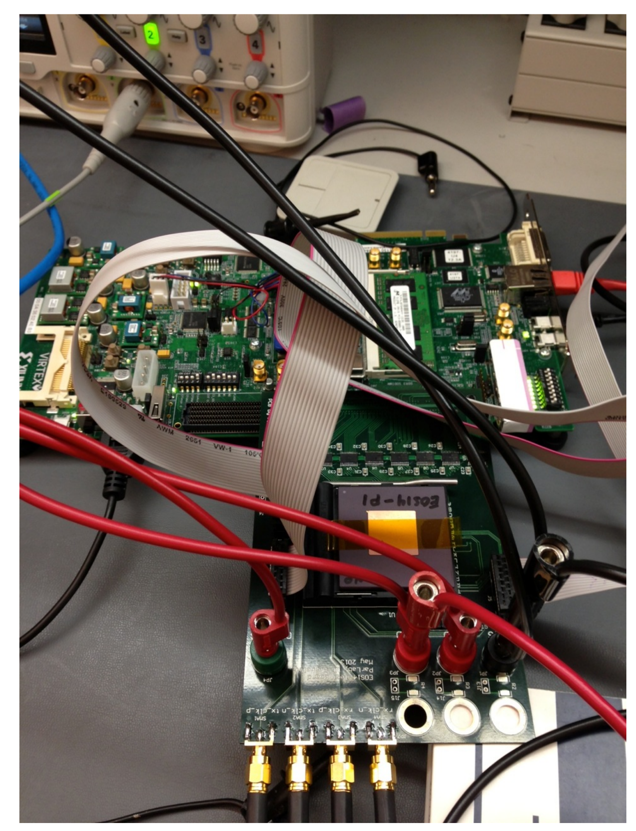
- 45 instructions in the RV32I base ISA
  - Additional 12 instructions for the RV64I base ISA
- Load-store architecture
- Multiprocessor synchronization via fetch-and-*op* and load-reserved/store-conditional
- Efficient position-independent code support
- ***No architecturally-visible delay slots***

Additional RISC-V Goals

- Support 64-bit address spaces for desktops and servers.
- Support 32-bit address spaces for small, low-power implementations.
- Implement the revised 2008 IEEE 754 floating-point standard.
- Easy to add ISA extensions and specialized variants.
- Fully virtualizable to ease hypervisor development.
- Support highly-parallel multicore or manycore implementations.
- Simple to subset for educational purposes and to reduce complexity of bringing up new implementations.

45nm RISC-V Processor





Fabricated in IBM 45nm SOI

Runs at > 1GHz

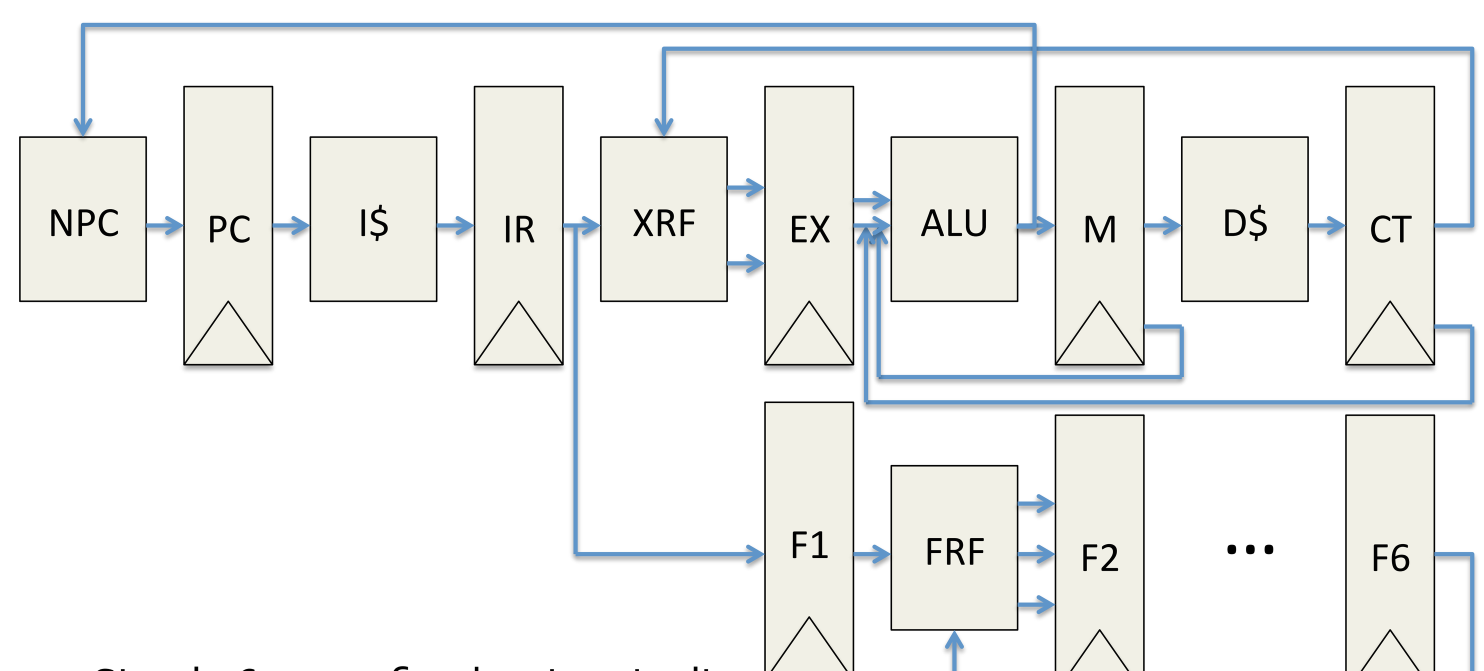
A RISC-V Supervisor ISA

- User and supervisor ISAs defined separately to facilitate different supervisor implementations (e.g. for µcontrollers)
- Current RV64 supervisor ISA:
  - 3-level page table; page sizes 8KB/8MB/8GB
  - Interprocessor interrupts
  - Timers
  - Tethered to host machine for disk, frame buffer

RISC-V Software Ecosystem

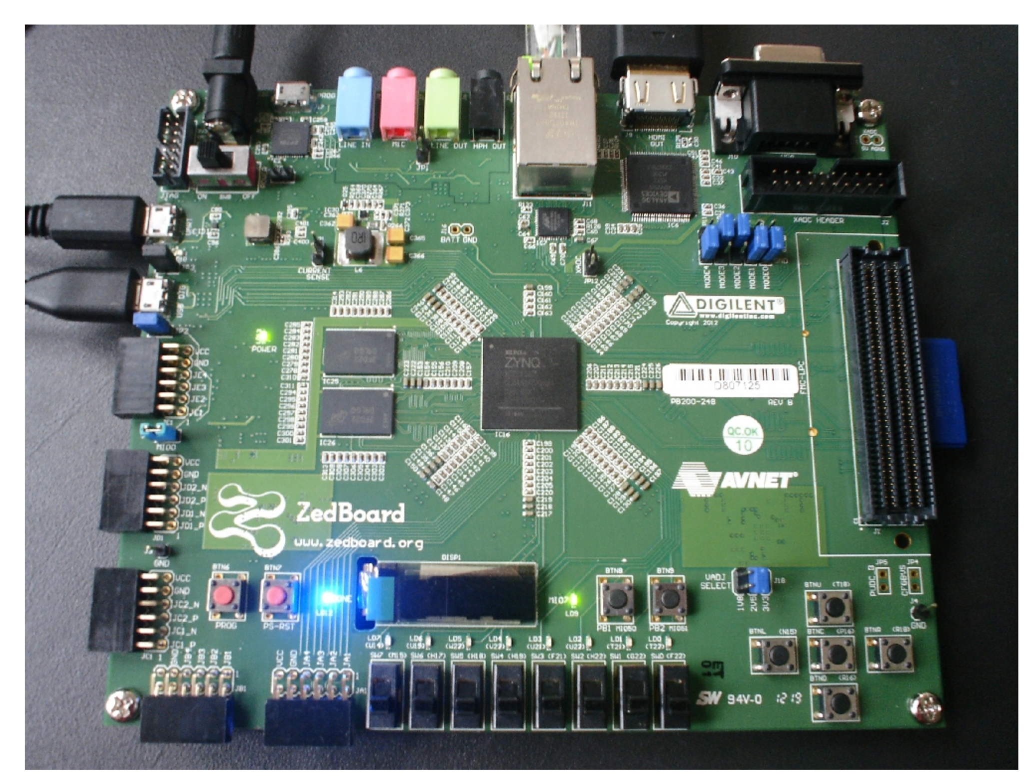
- GCC 4.6.1 with newlib and glibc C libraries
- "Proxy Kernel" to support POSIX calls by forwarding to a Linux host machine
- Akaros cloud research operating system
- Linux operating system
- python 3
- LLVM support in the near future

Rocket: A single-issue in-order RISC-V Implementation



- Simple 6-stage fixed-point pipeline
- Full supervisor support
- Comparable to a Cortex A5; 1.6 DMIPS/MHz (better than A5)
- 64-bit fixed-point datapath, double-precision FPU

DEMO



- Rocket core mapped to a ZedBoard running Linux

Research funded by DARPA Award Number HR0011-12-2-0016. Approved for public release; distribution is unlimited. The content of this paper does not necessarily reflect the position or the policy of the US government and no official endorsement should be inferred.