RISC-V

State of the union

Ilia Kurin

Table of contents

RISC-V

- Instruction Set Architecture (ISA)
- Based on Reduced Instruction Set Computer (RISC)

RICS

Simple instructions

Fixed size instructions

More registers

Software oriented

*Compiler divides code into smaller commands

CICS

Complex instructions

Variable size instructions

Less registers

Hardware oriented

ISA

Abstract model of computer

Pseudocode for computer architectures

Defines:

- Instructions
- Data types
- Registers
- I/O model
- Other fundamental features

Advantages

- Free to use for both academic and industrial purposes
  - No need to pay royalties
  - Faster time to market due to open ISA and cores

Advantages

- Free to use for both academic and industrial purposes
  - No need to pay royalties
  - Faster time to market due to open ISA and cores
- Extensibility and customization
  - No need to support outdated instructions

Advantages

- Free to use for both academic and industrial purposes
  - No need to pay royalties
  - Faster time to market due to open ISA and cores
- Extensibility and customization
  - No need to support outdated instructions
- Wide variety of applications
  - From embedded devices
  - To supercomputers

Advantages

- Free to use for both academic and industrial purposes
  - No need to pay royalties
  - Faster time to market due to open ISA and cores
- Extensibility and customization
  - No need to support outdated instructions
- Wide variety of applications
  - From embedded devices
  - To supercomputers
- Avoidance of previous architectures mistakes
  - No delay slots, no register windows and others

History

Berkeley RISC

- First version was released in 1981 UC Berkeley by David Patterson

History

Berkeley RISC

- First version was released in 1981 UC Berkeley by David Patterson
- Second version was released in 1983 and introduced expansion of 16 bit instructions to 32 bit

History

Berkeley RISC

- First version was released in 1981 UC Berkeley by David Patterson
- Second version was released in 1983 and introduced expansion of 16 bit instructions to 32 bit
- Third version was released in 1984 by Patterson's students and was designed to run Smalltalk

History

Berkeley RISC

- First version was released in 1981 UC Berkeley by David Patterson
- Second version was released in 1983 and introduced expansion of 16 bit instructions to 32 bit
- Third version was released in 1984 by Patterson's students and was designed to run Smalltalk
- Fourth version was released in 1988 by Patterson's students and was designed to run Lisp

History

RISC-V

- Work began at 2010 at UC Berkeley in Parallel Computing Laboratory
- In development participated Prof. Krste Asanović, Prof. David Patterson, Yunsup Lee and Andrew Waterman
- First ISA was published at 2011 and contained base instructions with floating-point and compressed extensions

History

RISC-V International

- Non-profit corporation curating ISA development
  - Openly accepting individual members
- Founded in 2015
- Before 2019 named as RISC-V Foundation
- Moved to Switzerland in 2020 to avoid USA trade regulations

Members

Source: [13] RISC-V members

Outline

Architectural decisions

- Smaller instruction set
  - Easier to write and execute code

Architectural decisions

- Smaller instruction set
    - Easier to write and execute code
- Load/store architecture
    - All operation are either memory read/write or between registers
    - Simpler instructions and better pipeline utilization

Architectural decisions

- Smaller instruction set
  - Easier to write and execute code
- Load/store architecture
  - All operation are either memory read/write or between registers
  - Simpler instructions and better pipeline utilization
- No branch delay slot
  - Instruction execution while branch target address is being computed
  - Easier for multicycle CPUs, superscalar CPUs, and long pipelines

Architectural decisions

- Smaller instruction set
  - Easier to write and execute code
- Load/store architecture
  - All operation are either memory read/write or between registers
  - Simpler instructions and better pipeline utilization
- No branch delay slot
  - Instruction execution while branch target address is being computed
  - Easier for multicycle CPUs, superscalar CPUs, and long pipelines
- No register windows
  - Registers are divided into windows where only on visible at a time
  - Context switching becomes expensive

Privilege levels

- Machine
  - Controls all physical resources and interrupts
  - The only mandatory privilege level
- Supervisor
  - Intended for kernels and drivers
  - Can only be used with both User and Machine privilege levels
- User/Application
  - Provides boundary between applications and hardware access

Processor can be in only one of the privilege levels at a time

Specifications

- **Unprivileged ISA**
- Privileged ISA
- Debug
- Trace

Link to current specifications: https://riscv.org/technical/specifications/

Terminology

- Core
  - Independent instruction fetch unit
- Hart
  - Hardware thread
  - Similar to Intel’s Hyper-threading
- Accelerator
  - Non-programmable unit or core with fixed function operating autonomously
- Field-Programmable Gate Array (FPGA)
  - Re-configurable integrated circuit that allows to implement a wide range of custom digital circuits

Naming conventions

- Base pattern is **RV [xxx] [abc. . . xyz]**
- Where xxx - address space size (32, 64, 128)
- abc. . . xyz - extension letters
  - Underscores between extensions are supported
  - Case insensitive
  - Supports custom extensions

General-purpose registers

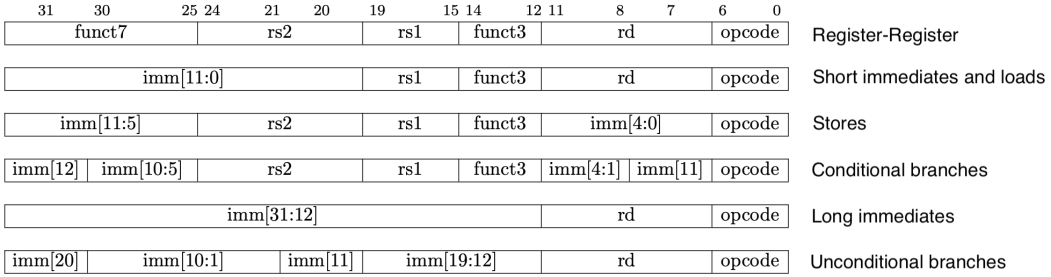
Register	Name	Usage	Preserved
x0	zero	Hardwired to 0, ignores writes	n/a
x1	ra	Return address for jumps	no
x2	sp	Stack pointer	yes
x3	gp	Global pointer	n/a
x4	tp	Thread pointer	n/a
x5-x7	t0-t2	Temporary registers	no
x8-x9	s0/fp-s1	Saved registers	yes
x10-x11	a0-a1	Function arguments / Return values	no
x12-x17	a2-a7	Function arguments	no
x18-x27	s2-s11	Saved registers	yes
x28-x31	t3-t6	Temporary registers	no
pc	(none)	Program counter	n/a

Extensions

Name	Letter	
Integer	**I**	G
Integer multiplication and division	M	G
Atomics	A	G
Single-Precision floating-point	F	G
Double-Precision floating-point	D	G
Quad-Precision floating-point	Q	
16-bit Compressed instructions	C	
Control and status register access	Zicr	
Instruction-Fetch fence	Zifencei	

Other instruction sets are in draft stage at the moment

Encoding variants


31 30 25 24 21 20 19 15 14 12 11 8 7 6 0
funct7 | rs2 | rs1 | funct3 | rd | opcode
Register-Register
imm[11:0] | rs1 | funct3 | rd | opcode
Short immediates and loads
imm[11:5] | rs2 | rs1 | funct3 | imm[4:0] | opcode
Stores
imm[12] | imm[10:5] | rs2 | rs1 | funct3 | imm[4:1] | imm[11] | opcode
Conditional branches
imm[31:12] | rd | opcode
Long immediates
imm[20] | imm[10:1] | imm[11] | imm[19:12] | rd | opcode
Unconditional branches

Source: [1] RISC-V assembly guide

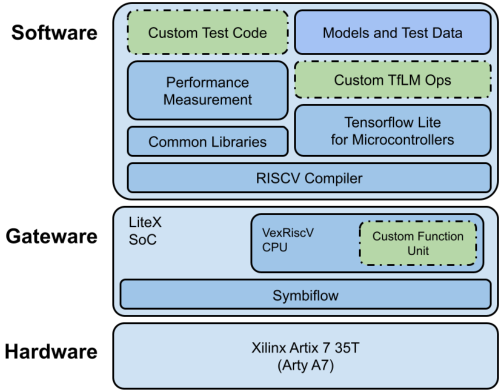
Pseudo-instructions

- Pseudo-instruction is a shorthand for one or more machine instructions
- Most of them use zero register as its usage improves performance due pipeline and command microcode optimizations
- Example:
  - There is no real instruction to compute Two’s complement
  - **neg rd, rs** is being converted to **sub rd, zero, rs**
  - It just subtracts the value (rs) from zero and writes to destination register (rd)

Custom Function Units

- CFU is a hardware core following a specific interface that provides custom instructions
- Goal is to create a solution allowing the creation of reusable extensions for any hardware with no need to wait multiple years for ratification
- Combines advantages of standard and custom extensions
- Work started in 2019 and it’s still in the draft stage

CFU Playground



Source: [11] CFU Playground

- Framework that allows to build and test CFUs for Machine Learning
- Requires LiteX Boards FPGA board or Renode and Verilator to simulate it
- Can only be run on Linux

Outline

Companies

Source: [16] SiFive

SiFive

- Leading RISC-V hardware company
- Founded in 2015
- First company to produce a RISC-V chip
- This March received $175 million funding, valuing the company at over $2.5 billion

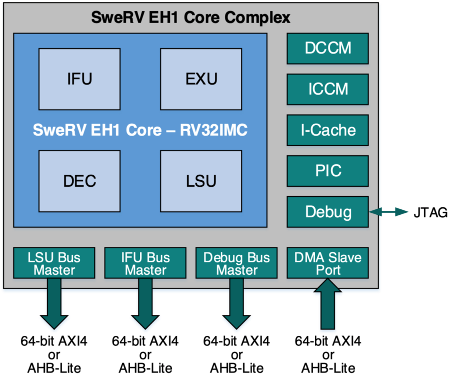
Companies

Source: [17] Andes technology

Andes Technology

- Biggest supplier of embedded RISC-V cores
- Joined RISC-V International in 2016
- Major toolchain contributor
- For the past 15 years, engaged with more than 150 partners

Cores

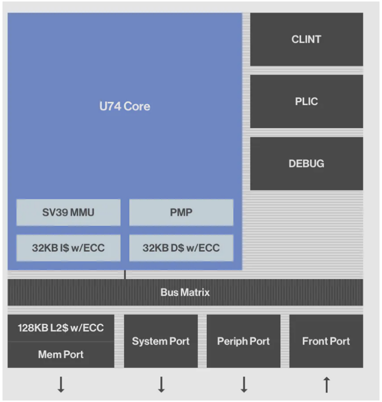


Source: [8] Western Digital SweRV EH1 documentation

Western Digital SweRV EH1

- RV32IMC core with branch predictor
- 1Ghz target frequency
- 4-way set-assotiative instruction cache
- Bus interfaces for instuction fetch, data access, debug access and external Direct Memory Access

Cores

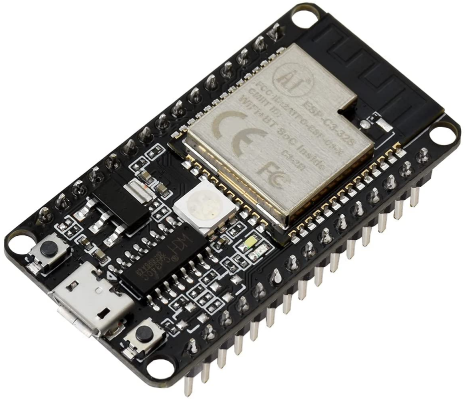


Source: [9] SiFive U74 documentation

SiFive U74

- RV64GC_Zba_Zbb_Sscofpmf core
- 8 region memory psysical memory protection
- Virtual Memory support with up to 47 Physical Address bits
- The L2 Cache can be configured into high speed deterministic SRAMs

Boards


Source: [14] WaveShare ESP-C3-32S-Kit

Waveshare ESP-C3-32S-Kit

SoC: ESP32-C3

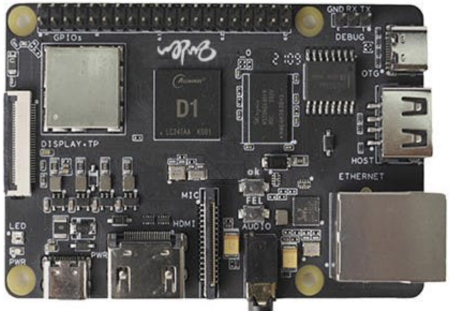
- Single RV32IMC core
- Frequency: 160 MHz
- Memory: 400KB of SRAM
- Wi-Fi and Bluetooth 5 support

Usage: IoT, wearable electronics

Price: 12.99 Euro

Buy: Link

Boards



Source: [15] Sipeed Nezha on Amazon

Sipeed Nezha

SoC: D1

- Single RV64GCV core (XuanTie C906)
- Memory: 1 GByte of DDR3 and 256 MByte of Nand Flash
- HiFi4 Digital Signal Processor
- G2D 2D graphics accelerators

Usage: Linux, IoT

Prices: 128.35 - 250.97 Euro

Buy: Link

Outline

Software ecosystem

- RISC-V has a great software ecosystem
  - Supports Linux, MacOS and Windows
  - Includes GCC, Clang, Glibc, Newlib and others compilers
  - Includes Fedora, Debian, Ubuntu and other Linux distributions
  - Includes ports for Golang, Java, Rust and even Node.JS
  - Includes multiple simulators, IDEs and much more

- Link to the software list: https://github.com/riscvarchive/riscv-software-list

Simulation options

- QEMU
  - Good for software developers
- SPIKE
  - Good for hardware developers
- RARS
  - Complete tool-set to start development
- Jupiter
  - Good for educational purposes

Code example

Multiplication without the required extension

Pipeline:

- riscv64-unknown-elf-gcc -march=**Architecture* -g3 -o test test.c
- riscv64-unknown-elf-objdump -S test

Compiled with GCC 11.1.0 on MacOS

Multiplication example

Architecture: RV64IM

```
lw     a5,-20(s0)
mv     a4,a5
lw     a5,-24(s0)
mulw   a5,a4,a5
sext.w a5,a5
```

```
int multiply(int a, int b) {
    return a * b;
}
```

Multiplication example

Architecture: RV64I

```
0000000000101f8 <__muldi3>:
    mv   a2,a0
    li   a0,0
    andi a3,a1,1
    beqz a3,10204 <__muldi3+0xc>
    add  a0,a0,a2
    srli a1,a1,0x1
    slli a2,a2,0x1
    bnez a1,101fc <__muldi3+0x4>
    ret
```

```
int multiply(int a, int b) {
    int result = 0;
    while (b > 0) {
        if (b & 1) {
            result += a;
        }
        b = b » 1;
        a = a « 1;
    }
    return result;
}
```

Outline

Is it really that free?

- **Not really**
- ISA is free and open-source but hardware can be private.

Is it really extensible?

- **Yes**
- Could be a great feature in a long-run. It can make adding new instruction sets easier.

Does it have future in embedded devices?

- **Probably yes**
- Highly customizable and memory efficient. Can find its way IoT and micro-controller sectors.

Is it ready for High-Performance market?

- **No**
- Almost no hardware at the moment

Final conclusion

- RISC-V is an interesting technology that is still in the development stage
- Today RISC-V is is far behind Intel or ARM in terms of performance
- Market is still not stable so there is room for hardware startups
- If you are enthusiastic RISC-V can be a great opportunity to make your own project

Sources

[1] RISC-V assembly guide, 20.05.22

[2] RISC-V history, 25.05.22

[3] RISC-V specifications, 20.05.22

[4] Design and implementation of RISC I, 20.05.22

[5] RISC II, 20.05.22

[6] Vector intrinsics, 26.05.22

[7] Delay slots, 26.05.22

[8] Western Digital SweRV EH1 documentation, 01.06.22

[9] SiFive U74 documentation, 01.06.22

[10] First RISC-V ISA, 01.06.22

[11] CFU Playground, 01.06.22

[12] RISC-V Cores and SoC Overview, 31.06.22

[13] RISC-V members, 08.06.22

[14] WaveShare ESP-C3-32S-Kit, 13.09.22

[15] Sipeed Nezha on Amazon, 13.09.22

[16] SiFive, 13.09.22

[17] Andes technology, 13.09.22