Assembly Language Lecture Series: RISC-V: An Introduction

Roger Luis Uy
College of Computer Studies
De La Salle University
Manila, Philippines

RISC-V resources

- RISC-V website: https://riscv.org/
- RISC-V specifications and manual (vol 1): https://riscv.org/technical/specifications/

- RARS (RISC-V Assembler, Runtime and Simulator) – assemble and simulate the execution of RISC-V assembly language
- RARS software: https://github.com/TheThirdOne/rars/releases
- Current version: v1.5
- RARS is distributed as an executable jar. Requires at least Java 8 to run it.
- Wiki:

https://github.com/TheThirdOne/rars/wiki

RISC-V

- RISC-V is an open-source RISC-based Instruction-set Architecture (ISA)
- Designed to support computer architecture research and education
- Also, standard free and open architecture for industry implementation
- An ISA separated into a small base integer ISA, usable by itself as a base for customized accelerators or for educational purposes, and optional standard extensions, to support general purpose software development

RISC-V

- RISC-V was chosen to represent the fifth major RISC ISA design from UC Berkeley
  - RISC-I (1981)
  - RISC-II (1981)
  - SOAR (RISC-III) (1984)
  - SPUR (RISC-IV) (1988)
  - RISC-V (2011)

RISC-V base

- RISC-V consists of a mandatory base integer ISA and optional extensions
- The base specifies
  - registers (and their sizes)
  - memory and addressing
  - instructions (and their encoding)
  - control flow
  - logic (i.e., integer) manipulation
  - ancillaries
- The base alone can implement a simplified general-purpose computer, with full software support, including a general-purpose compiler.

RISC-V base

Name	Description	Version*	Status
RV32I	Base integer instruction set, 32-bit	2.1	Ratified
RV32E	Base integer instruction set (embedded), 32-bit, 16 registers	1.9	Draft
RV64I	Base integer instruction set, 64-bit	2.1	Ratified
RV128I	Base integer instruction set, 128-bit	1.7	Draft

*as of December 13, 2019

*The ISA modules marked Ratified have been ratified at this time.
*The modules marked Draft are expected to change before ratification.
*The modules marked Frozen are not expected to change significantly before being put up for ratification.

RISC-V extension

Name	Description	Version	Status
M	Standard extension for integer multiplication and division	2.0	Ratified
A	Standard extension for atomic instructions	2.1	Ratified
F	Standard extension for single-precision floating-point	2.2	Ratified
D	Standard extension for double-precision floating-point	2.2	Ratified
Zicsr	Control and Status register (CSR)	2.0	Ratified
Zifencei	Instruction-fetch fence	2.0	Ratified

The base, four extensions, CSR and instruction-fetch fence are collectively called RV32G

RISC-V extension

Name	Description	Version	Status
Q	Standard extension for quad-precision floating-point	2.2	Ratified
C	Standard extension for compressed instructions	2.0	Ratified
Counters		2.0	Draft
L	Standard extension for decimal floating-point	0.0	Draft
B	Standard extension for bit manipulation	0.0	Draft
J	Standard extension for dynamically translated languages	0.0	Draft
T	Standard extension for transactional memory	0.0	Draft
P	Standard extension for packed-SIMD instructions	0.2	Draft
V	Standard extension for vector operations	0.7	Draft
Zam	Standard extension for misaligned atomics	0.1	Draft
Ztso	Standard extension for total store ordering	0.1	Frozen

RISC-V Assembly program

```
.globl main
.data
  ANS: .word  0
  COUNT: .word 5
  NUMB: .word 1,2,3,4,5
.text
main:
    addi x5, x0, 0    # initialize answer
    lw x6, COUNT      # count
    la x7, NUMB       # x7 points to NUMB
L2: lw x8, (x7)
    add x5, x5, x8    # add
    addi x6, x6, -1   # decrement count
    beq x6, x0,  L1
    addi x7, x7, 4    # increment pointer to next number
    j L2
L1: la x10, ANS
    sw x5, (x10)       # store result to ANS
```

```
#print
  mv a0, x5
  li a7, 1
  ecall
#exit program
   li a7, 10
   ecall
```