# RISC-V Cheatsheet

Marquette University
Department of Computer Science – Systems Lab

## Registers

| Register | ABI Name | Saver | Description |
|---|---|---|---|
| x0 | zero | — | Hard-wire zero |
| x1 | ra | Caller | Return address |
| x2 | sp | Callee | Stack pointer |
| x3 | gp | — | Global pointer |
| x4 | tp | — | Thread pointer |
| x5-x7 | t0-t2 | Caller | Temporaries |
| x8 | s0/fp | Callee | Saved register/frame pointer |
| x9 | s1 | Callee | Saved register |
| x10-11 | a0-a1 | Caller | Function arguments return values |
| x12-17 | a2-a7 | Caller | Function arguments |
| x18-27 | s2-s11 | Callee | Saved registers |
| x28-31 | t3-t6 | Caller | Temporaries |

## Arithmetic

| Format | Description | Pseudocode |
|---|---|---|
| `add rd, rs1, rs2` | Adds two registers | `rd` ← `rs1 + rs2` |
| `addi rd, rs1, const` | Adds a register and a constant value | `rd` ← `rs1 + const` |
| `sub rd, rs1, rs2` | Subtracts two registers | `rd` ← `rs1 - rs2` |
| `subi rd, rs1, const` | Subtracts a register and a constant value | `rd` ← `rs1 - const` |
| `mv rd, rs` | Moves the contents of one register into antoher | `rd` ← `rs` |

## Loads and Stores

| Format | Description | Pseudocode |
|---|---|---|
| `ld rd, const(rs)` | Load doubleword (8 bytes, 64-bit only) | `rd` ← `rs`[*const*] |
| `lw rd, const(rs)` | Load word (4 bytes) | `rd` ← `rs`[*const*] |
| `lb rd, const(rs)` | Load byte | `rd` ← `rs`[*const*] |
| `li rd, const` | Load immediate | `rd` ← *const* |
| `sd rd, const(rs)` | Store doubleword (8 bytes, 64-bit only) | `rd` → `rs`[*const*] |
| `sw rd, const(rs)` | Store word (4 bytes) | `rd` → `rs`[*const*] |
| `sb rd, const(rs)` | Store byte | `rd` → `rs`[*const*] |

## Jumps and Bracnhes

| Format | Description | Pseudocode |
|---|---|---|
| `j label` | Jumps to *label*. Does not store return address | *label*() |
| `jal rd, label` | Jumps to offset of *const*. Stores return address in `rd` | `rd = pc + 4`<br>`pc = pc + const` |
| `jalr rd, const(rs)` | Jumps to `rs` with an offset of *const*. Stores return address in `rd` | `rd = pc + 4`<br>`pc = pc + rs[const]` |
| `beq rs1, rs2, label` | Branch if equal | `if(rs1==rs2) { label(); }` |
| `bne rs1, rs2, label` | Branch if not equal | `if(rs1!=rs2) { label(); }` |
| `bge rs1, rs2, label` | Branch if greater than or equal to | `if(rs1>=rs2) { label(); }` |
| `blt rs1, rs2, label` | Branch if less than | `if(rs1<rs2) { label(); }` |

## Bitwise

| Format | Description | Pseudocode |
|---|---|---|
| `xori rd, rs, const` | XOR with *const* | `rd` ← `rs` ⊕ *const* |
| `xor rd, rs1, rs2` | XOR with two registers | `rd` ← `rs1` ⊕ *rs2* |
| `andi rd, rs, const` | And with register and a *const* value | `rd` ← `rs`&*const* |
| `and rd, rs1, rs2` | And with two registers | `rd` ← `rs1`&*rs2* |
| `ori rd, rs, const` | Or with a register and a *const* value | `rd` ← `rs`‖*const* |
| `or rd, rs1, rs2` | Or with two registers | `rd` ← `rs1`‖*rs2* |
| `slli rd, rs1, const` | Shift left with a register and a *const* value | `rd` ← `rs1` << *const* |
| `sll rd, rs1, rs2` | Shift left with two registers | `rd` ← `rs1` << `rs2` |
| `srli rd, rs1, const` | Shift right with a register and a *const* value | `rd` ← `rs1` >> *const* |
| `srl rd, rs1, rs2` | Shift right with two registers | `rd` ← `rs1` >> `rs2` |

**v1.0**