0000 0010 0000 0000 0010 0000 1000 0011 lw 02 00 20 83 lw x1, 32(x0)

0000 0010 0100 0000 0010 0001 0000 0011 lw 02 40 21 03 lw x2, 36(x0)

0000 0010 1000 0000 0010 0001 1000 0011 lw 02 80 21 83 lw x3, 40(x0)

0100 0000 0010 0000 1000 0000 1011 0011 sub 40 20 80 b3 sub x1, x1, x2

0000 0000 0010 0000 1111 0010 0011 0011 and 00 20 f2 33 and x4, x1, x2

0000 0000 0011 0000 1111 0010 1011 0011 and 00 30 f2 b3 and x5, x1, x3

0000 0010 0001 0000 0010 0010 0010 0011 sw 02 10 22 23 sw x1, 36(x0)

0000 0000 0100 0010 1010 0110 0010 0011 sw 00 42 a6 23 sw x4, 12(x5)

0000 0000 0101 0010 0110 0011 0011 0011 or 00 52 63 33 or x6, x4, x5

1111 1100 0110 0010 1000 1110 1110 0011 beq fc 62 8e e3 beq x5, x6, -36

lw rd, imm(rs1)

Bits<12> imm = $encoding[31:20];

Bits<5> xs1 = $encoding[19:15];

Bits<5> xd = $encoding[11:7];

XReg virtual\_address = X[xs1] + $signed(imm);

X[xd] = $signed([read\_memory](https://riscv-software-src.github.io/riscv-unified-db/manual/html/isa/isa_20240411/funcs/funcs.html" \l "read_memory-func-def)<32>(virtual\_address, $encoding));

and rd, rs1, rs2

Bits<5> xs2 = $encoding[24:20];

Bits<5> xs1 = $encoding[19:15];

Bits<5> xd = $encoding[11:7];

X[xd] = X[xs1] & X[xs2];

beq rs1, rs2, imm

**Bits<13>** imm = {$encoding[31], $encoding[7], $encoding[30:25], $encoding[11:8], 1'd0};

**Bits<5>** xs2 = $encoding[24:20];

**Bits<5>** xs1 = $encoding[19:15];

**XReg** lhs = X[xs1];

**XReg** rhs = X[xs2];

**if** (lhs == rhs) {

[jump\_halfword](https://riscv-software-src.github.io/riscv-unified-db/manual/html/isa/isa_20240411/funcs/funcs.html#jump_halfword-func-def)($pc + $signed(imm));

}

Branch to PC + imm if the value in register xs1 is equal to the value in register xs2.

Raise a MisalignedAddress exception if PC + imm is misaligned.