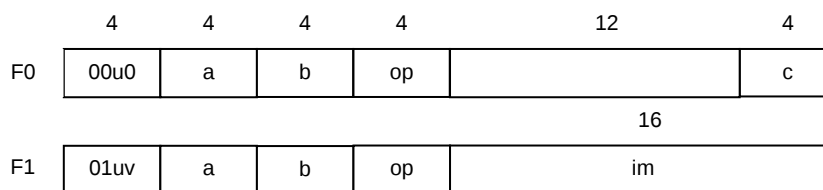# The RISC Architecture

## 0. Resources and registers

From the viewpoints of the programmer and the compiler designer the computer consists of an arithmetic unit, a control unit and a store. The arithmetic unit contains 16 registers R0 – R15, with 32 bits each. The control unit consists of the instruction register IR, holding the instruction currently being executed, and the program counter PC, holding the word-address of the instruction to be fetched next. All branch instructions are conditional. The memory consists of 32-bit words, and it is byte-addressed. Furthermore, there are 4 flag registers N, Z, C and V, called the *condition codes*.

There are four types of instructions and instruction formats. *Register instructions* operate on registers only and feed data through a shifter or the arithmetic logic unit ALU. *Memory instructions* fetch and store data in memory. *Branch instructions* affect the program counter.
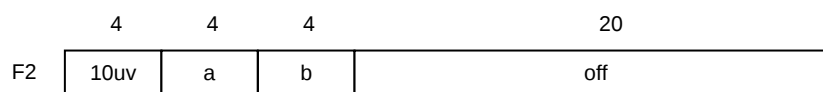
## 1. Register instructions          (formats F0 and F1)

Register instructions assign the result of an operation to the destination register R.a. The first operand is the register R.b. The second operand *n* is either register R.c or is the constant *im*.

| | 4 | 4 | 4 | 4 | 12 | 4 |
|---|---|---|---|---|---|---|
| F0 | 00u0 | a | b | op | | c |

| | 4 | 4 | 4 | 4 | 16 |
|---|---|---|---|---|---|
| F1 | 01uv | a | b | op | im |

| | | | |
|---|---|---|---|
| 0 | MOV a, n | R.a := n | |
| 1 | LSL  a, b, n | R.a := R.b ← n | (shift left by n bits) |
| 2 | ASR  a, b, n | R.a := R.b → n | (shift right by n bits with sign extension) |
| 3 | ROR  a, b, n | R.a := R.b rot n | (rotate right by n bits) |
| 4 | AND  a, b, n | R.a := R.b & n | logical operations |
| 5 | ANN  a, b, n | R.a := R.b & ~n | |
| 6 | IOR   a, b, n | R.a := R.b or n | |
| 7 | XOR  a, b, n | R.a := R.b xor n | |
| 8 | ADD  a, b, n | R.a := R.b + n | integer arithmetic |
| 9 | SUB  a, b, n | R.a := R.b – n | |
| 10 | MUL  a, b, n | R.a := R.a x n | |
| 11 | DIV   a, b, n | R.a := R.b div n | |
| 12 | FAD  a, b, c | R.a := R.b + R.c | floating-point arithmetic |
| 13 | FSB  a, b, c | R.a := R.b – R.c | |
| 14 | FML  a, b, c | R.a := R.a x R.c | |
| 15 | FDV  a, b, c | R.a := R.b / R.c | |

Immediate values are extended to 32 bits with 16 v-bits to the left. Apart from R.a these instructions also affect the flag registers N (negative) and Z (zero). The ADD and SUB instructions also set the flags C (carry, borrow) and V (overflow).

## 2. Memory instructions          (format F2)

| | 4 | 4 | 4 | 20 |
|---|---|---|---|---|
| F2 | 10uv | a | b | off |

| | | | |
|---|---|---|---|
| LD | a, b, im | R.a := Mem[R.b + off] | u = 0 |
| ST | a, b, im | Mem[R.b + off] := R.a | u = 1 |

If v = 0, access is for a word (4 bytes). If v = 1, a single byte is accessed.

## 3. Branch instructions    (Format F3)

| 4 | 4 | 16 | 4 | 4 |
|---|---|----|---|---|
| 110v | cond | | 0000 | c |

| F3 | 111v | cond | off |
|----|------|------|-----|

    Bcond   dest

If u = 0, the destination address is taken from register R.c. If u = 1, it is PC+1 + offset. If v = 1, the link address PC+1 is deposited in register R15.

| code | cond | | condition | code | cond | | condition |
|------|------|--|-----------|------|------|--|-----------|
| 0000 | MI | negative (minus) | N | 1000 | PL | positive (plus) | ~N |
| 0001 | EQ | equal (zero) | Z | 1001 | NE | positive (plus) | ~Z |
| 0010 | CS | carry set (lower) | C | 1010 | CC | carry clear | ~C |
| 0011 | VS | overflow set | V | 1011 | VC | overflow clear | ~V |
| 0100 | LS | lower or same | ~C|Z | 1100 | HI | higher | ~(~C}Z) |
| 0101 | LT | less than | N≠V | 1101 | GE | greater or equal | ~(N≠V) |
| 0110 | LE | less or equal | (N≠V)|Z | 1110 | GT | greater than | ~((N≠V)|Z) |
| 0111 | | always | true | 1111 | | never | false |

## 4. Special features

Modifier bit u = 1 changes the effect of certain instructions as follows:

| | |
|--|--|
| ADD', SUB' | add, subtract also carry C |
| MUL' | unsigned multiplication |
| MOV' form 0, v = 0: | R.a := H |
| MOV' form 0, v = 1: | R.a := [N, Z, C, V] |
| MOV' form 1 | R.a := [imm 16'b0]    (imm left shifted 16 bits) |

The MUL instruction deposits the high 32 bits of the product in the auxiliary register H. The DIV instruction deposits the remainder in H.

## 5. Interrupts

The addition of an interrupt facility required the addition of two new instruction, as well as the status register *intenb* (interrupt enable). The instructions are

RTI          1100 0111 xxxx xxxx xxxx xxxx 0001  Rn          return from interrupt

STI/CLI       1100 1111 xxxx xxxx xxxx xxxx 0010 000e      set / clear interrupt, intenb := e

These instructions are encoded as branch instructions with bits 4 or 5 set.

## The RISC0 implementation

The RISC architecture has been implemented on a Xilinx FPGA contained on the development board Spartan. RISC0 stands at the origin of an evolving series of extensions. It represents a Harvard architecture, and it uses FPGA-internal RAM for its memory, which is restricted to 8K words of program and 8K words for data. It does not feature byte access, and the floating-point instructions are not available..

RISC0's external devices are the following:

| adr | hex | input | output |
|-----|------|-------|--------|
| -64 | 0FFFC0H | millisecond counter | |
| -60 | 0FFFC4H | switches | LEDs |
| -56 | 0FFFC8H | RS-232 data | RS-232 data |
| -52 | 0FFFCCH | RS-232 status* | RS-232 control |

* bit 0: receiver ready, bit 1: transmitter ready

## The RISC5 implementation

RISC5 is an extension of RISC0 based on a von Neumann architecture and uses the same instruction set. The memory consists of the board-internal SRAM with a capacity of 1 MB. Byte access is available, and so are floating-point instructions.

RISC5's external devices are the following:

| adr | hex | input | output |
|-----|------|-------|--------|
| -64 | 0FFFC0H | millisecond counter | |
| -60 | 0FFFC4H | switches | LEDs |
| -56 | 0FFFC8H | RS-232 data | RS-232 data |
| -52 | 0FFFCCH | RS-232 status | RS-232 control |
| -48 | 0FFFD0H | disk, net  SPI data | SPI data |
| -44 | 0FFFD4H | disk, net  SPI status | SPI control |
| -40 | 0FFFD8H | keyboard data  (PS2) | |
| -36 | 0FFFDCH | mouse (and kbd status) | |

A further added device is the video controller. It maps memory at 0E7F00H - 0FFEFFH onto the display (1024 x 768 pixels).