

## Lab# 12 THE SINGLE CYCLE DATAPATH

---

**Instructor:** I Putu Danu Raharja.

**Objectives:**

Learn how to implement instructions for a CPU.

**Method:**

Learn to implement the single cycle datapath for a subset of 16-bit MIPS-like processor.

**Preparation:**

Read the slides.

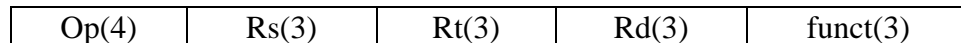
**File To Use:**

### 12.1 OVERVIEW:

Suppose we would like to design a simple 16-bit MIPS-like processor with seven 16-bit general-purpose registers: R1 through R7. R0 is hardwired to zero and cannot be written, so we are left with seven registers. There is also one special-purpose 16-bit register, which is the program counter (PC). All instructions are also 16 bits. There are three instruction formats, R-type, I-type, and J-type as shown below:

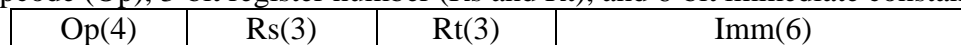
**R-type format:**

4-bit opcode (Op), 3-bit register numbers (Rs, Rt, and Rd), and 3-bit function field (funct)



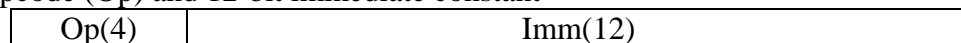
**I-type format:**

4-bit opcode (Op), 3-bit register number (Rs and Rt), and 6-bit immediate constant



**J-type format:**

4-bit opcode (Op) and 12-bit immediate constant



For R-type instructions, Rs and Rt specify the two source register numbers, and Rd specifies the destination register number. The function field can specify at most eight functions for a given opcode. We will reserve opcode 0 and opcode 1 for R-type instructions.

For I-type instructions, Rs specifies a source register number, and Rt can be a second source or a destination register number. The immediate constant is only 6 bits because of

the fixed size nature of the instruction. The size of the immediate constant is suitable for our uses. The 6-bit immediate constant is signed (and sign-extended) for all I-type instructions.

For J-type, a 12-bit immediate constant is used for instructions such as J (jump), JAL (jump-and-link), and LUI (load upper immediate) instructions.

### Instruction Encoding:

Eight R-type instructions, six I-type instructions, and three J-type instructions are defined. These instructions, their meanings, and their encodings are shown below:

Instr	Meaning	Encoding				
		Op	Rs	Rt	Rd	f
SLL	$\text{Reg(Rd)} = \text{Reg(Rs)} \ll \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 000
ROL	$\text{Reg(Rd)} = \text{Reg(Rs)} \text{ rotate} \ll \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 001
SRL	$\text{Reg(Rd)} = \text{Reg(Rs)} \text{ zero} \gg \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 010
SRA	$\text{Reg(Rd)} = \text{Reg(Rs)} \text{ sign} \gg \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 011
AND	$\text{Reg(Rd)} = \text{Reg(Rs)} \& \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 100
OR	$\text{Reg(Rd)} = \text{Reg(Rs)} \mid \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 101
NOR	$\text{Reg(Rd)} = \sim(\text{Reg(Rs)} \mid \text{Reg(Rt)})$	Op = 0000	Rs	Rt	Rd	f = 110
XOR	$\text{Reg(Rd)} = \text{Reg(Rs)} \wedge \text{Reg(Rt)}$	Op = 0000	Rs	Rt	Rd	f = 111
ADD	$\text{Reg(Rd)} = \text{Reg(Rs)} + \text{Reg(Rt)}$	Op = 0001	Rs	Rt	Rd	f = 000
SUB	$\text{Reg(Rd)} = \text{Reg(Rs)} - \text{Reg(Rt)}$	Op = 0001	Rs	Rt	Rd	f = 001
SLT	$\text{Reg(Rd)} = \text{Reg(Rs)} \text{ signed} < \text{Reg(Rt)}$	Op = 0001	Rs	Rt	Rd	f = 010
SLTU	$\text{Reg(Rd)} = \text{Reg(Rs)} \text{ unsigned} < \text{Reg(Rt)}$	Op = 0001	Rs	Rt	Rd	f = 011
JR	PC = lower 12 bits of Reg(Rs)	Op = 0001	Rs	000	000	f = 111
ANDI	$\text{Reg(Rt)} = \text{Reg(Rs)} \& \text{ext(im}_6\text{)}$	Op = 0100	Rs	Rt	Immediate <sup>6</sup>	
ORI	$\text{Reg(Rt)} = \text{Reg(Rs)} \mid \text{ext(im}_6\text{)}$	Op = 0101	Rs	Rt	Immediate <sup>6</sup>	
ADDI	$\text{Reg(Rt)} = \text{Reg(Rs)} + \text{ext(im}_6\text{)}$	Op = 1000	Rs	Rt	Immediate <sup>6</sup>	
SLTI	$\text{Reg(Rt)} = \text{Reg(Rs)} \text{ signed} < \text{ext(im}_6\text{)}$	Op = 1010	Rs	Rt	Immediate <sup>6</sup>	
LW	$\text{Reg(Rt)} = \text{Mem}(\text{Reg(Rs)} + \text{ext(im}_6\text{)})$	Op = 0110	Rs	Rt	Immediate <sup>6</sup>	
SW	$\text{Mem}(\text{Reg(Rs)} + \text{ext(im}_6\text{)}) = \text{Reg(Rt)}$	Op = 0111	Rs	Rt	Immediate <sup>6</sup>	
BEQ	Branch if $(\text{Reg(Rs)} == \text{Reg(Rt)})$	Op = 1001	Rs	Rt	Immediate <sup>6</sup>	
BNE	Branch if $(\text{Reg(Rs)} \neq \text{Reg(Rt)})$	Op = 1011	Rs	Rt	Immediate <sup>6</sup>	
J	PC = Immediate <sup>12</sup>	Op = 1100	Immediate <sup>12</sup>			
JAL	R7 = PC + 1, PC = Immediate <sup>12</sup>	Op = 1101	Immediate <sup>12</sup>			
LUI	R1 = Immediate <sup>12</sup> << 4	Op = 1111	Immediate <sup>12</sup>			

## **12.2 LAB EXERCISE**

Based on the above requirement, implement only the datapath.