MIPS Instruction Set

Dr. Prasenjit Chanak

Department of Computer Science and Engineering
Indian Institute of Technology (BHU), Varanasi
UP, 221005

Instruction Format

- **Instruction Format:** A form of representation of an instruction composed of fields of binary numbers

- *op:* Basic operation of the instruction, traditionally called the **opcode**.
- *rs:* The first register source operand.
- *rt:* The second register source operand.
- *rd:* The register destination operand. It gets the result of the operation.
- *shamt:* Shift amount
- *funct:* Function. This field, oft en called the *function code*, selects the specific variant of the operation in the op field

Instruction Format

- **R-Type** : instructions use opcode **000000**.
  - This group Contains all instructions that do not require an immediate value, target offset, memory address displacement, or memory address to specify an operand.
  - Includes arithmetic and logic with all **operands in registers**, shift instructions, and register direct jump instructions (jalr and jr).
- **I-Type** : Opcodes **except 000000, 00001x, and 0100xx**
  - This group includes instructions with **an immediate operand**, branch instructions, and load and store instructions. In the MIPS architecture, all memory accesses are handled by the main processor, so coprocessor load and store instructions are included in this group.
- **J-Type** : instructions use opcode **00001x**.
  - This group consists of the two **direct jump instructions** (j and jal).
  - These instructions require a memory address to specify their operand.

Machine Language

- Instructions are 32 bits long
- Registers have numbers 0 .. 31, e.g., $t0=8, $t1=9, $s0=16, $s1=17 etc.
- Instruction Format:

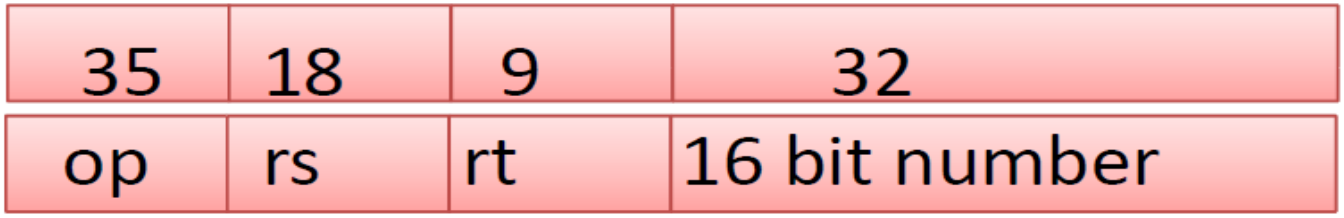
  Example: add $t0, $s1, $s2

000000	10001	10010	01000	00000	100000
op	rs	rt	rd	shamt	funct

Shamt: Shift amount, funct: function

Machine Language

- lw / sw and the regularity principle?
  - New principle: Good design demands a compromise
- new format (I-type), other format was R
- Example: lw $t0, 32($s2)



Lw $dst imm($src) : $dst ← M [$src+imm] Refers from M location, it a type of Addressing

Control Instruction

- Decision making instructions - alter the control flow
- MIPS conditional branch instructions:

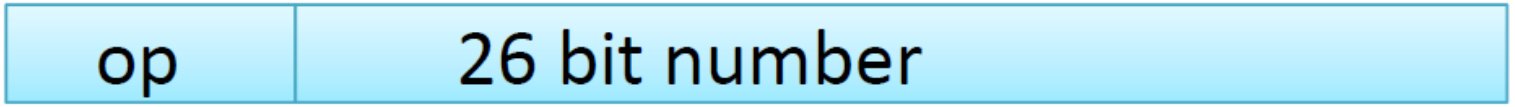
  bne and beq
- Example:

```
if (i == j)                bne $s0, $s1, Label
    h = i + j;             add $s3, $s0, $s1
                    Label: ....
```

Format of Instructions for Control

- beq, bne I - format

- j J - format (new)

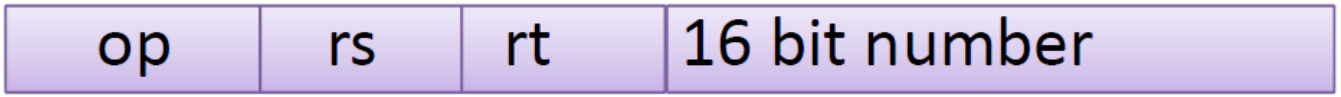


- slt R – format :set-if-less-than

Constants in MIPS instructions

```
addi $29, $29, 4 ‘i’ is for ‘immediate’
slti  $8, $18, 10
andi $29, $29, 6
ori   $29, $29, 4
```

I – Format is used



A special constant

‘0’ is a special constant, hard-wired into register $0 (also called $zero)

add $s2, $s4, $zero

means move from $s4 to $s2