ISA Design

**UPDATED**

## **Introduction**

We are to design 8 bit ISA covering 3 Instruction formats

* R-type
* I-type
* J-type

R-type:

7-5 4-3 2-1 0

|  |  |  |
| --- | --- | --- |
| **OP Code** | **Rs** | **rd** |

4 bit 4 bit 4 bit

I-type:

|  |  |  |
| --- | --- | --- |
| **OP Code** | **rs** | **Immediate/address** |

4 bit 4 bit 4 bit

J-Type:

|  |  |
| --- | --- |
| **OP Code** | **Target** |

4 bit 8 bit

**Operands:**

**There are 2 operands. Those operands are mixed type.**

**Operation:**

There are total operations divided into 5 types.

|  |  |  |
| --- | --- | --- |
| Operation Type | Operations | Register Type |
| Arithmetic | ADD, SUB | R-type |
| Arithmetic | ADDi | I-type |
| Data transfer | LW, SW | I-type |
| Conditional Branch | BEQ, SLTi | I-type |
| Unconditional Jump | J, DISP | J-type |

**R-type:**

For R-type, opcode is 3 bit in length, $rs and $rd is 2 bit in length, and function field is 1 bit in length.

$rd is the destination register and $rs id the source register.

**I-type:**

For I-type, opcode is 3 bit in length, $rs is the destination register

**J- type:**

In j-type instruction format 3 bit is assigned for opcode and rest of the 5 bit is assigned for address or target.

**Operation List:**

**R-type operation list:**

For R-type operation the opcode is always 000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operation | Opcode | Function code | Syntax | Meaning |
| ADD | 000 | 0 | Add $t0, $t2 | $t0=$t0+$t2 |
| SUB | 000 | 1 | Sub $t0,$t2 | $t0=$t0-$t2 |

**I-type operation list:**

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Opcode | Syntax | Meaning |
| ADDI | 0001 | Addi $t0, 4 | $t0=$t0+4 |
| LW | 0010 | Lw 1($t0) | $t3=Mem[4+$t0]  #here $t3 is a declared register which is used to load the data from memory |
| SW | 0011 | Sw 1($t0) | Mem[4+$t0] = $t3  # memory gets the value of $t3 |
| BEQ | 0100 | Beq $t3, address | If($t3==$zero) goto address |
| SLTi | 0101 | Slti $t0,4 | If($t0<4) then $t4=0  Else $t4=1  #here t3 is a declares register with assigned value |

**J-type**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | Opcode | Syntax | Meaning |
| J | 110 | J address | Jump to target address |
| disp | 111 | Disp $t3 | Display the content of **$t3**  Will be displayed. |

List of registers:

|  |  |  |
| --- | --- | --- |
| Register Name | Value | Comment |
| $t0 | 0000 | Used directly for performing operation like arithmetic or branch type |
| $t1 | 0001 | Same as above |
| $t2 | 0010 | Same as above |
| $t3 | 0011 | $t3 is reserved for sw, lw operations only |
| $t4 | 0100 | $t4 is reserved for beq, bne operations only |
| $t5 | 0101 | Used directly for performing operation like arithmetic or branch type |
| $t6 | 0110 | Same as above |
| $t7 | 0111 | Same as above |
| $t8 | 1000 | Same as above |
| $t9 | 1001 | Same as above |
| $t10 | 1010 | Same as above |
| $t11 | 1011 | Same as above |
| $t12 | 1100 | Same as above |
| $t13 | 1101 | Same as above |
| $t14 | 1110 | Same as above |
| $zero | 1111 | Contain fixed value zero. |
|  |  |  |

**\*\*** Apart from those there is a special register **$zero holding 0** value, value which is used in performing beq operation only