|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Op-code(6)** | **Rs(5)** | **Rt(5)** | **Rd(5)** | **Shamt(5)** | **Funct(6)** |
| 100000 | 00000 | 00001 | 00000 | 00001 | 000000 |
| 100000 | 00000 | 00010 | 00000 | 00001 | 000100 |
| 001110 | 00001 | 00011 | 00000 | 00000 | 001111 |
| 000000 | 00011 | 00001 | 00000 | 00000 | 011010 |
| 000011 | 00010 | 00000 | 00100 | 00000 | 001001 |
| 000001 | 00101 | 10001 | 00000 | 00000 | 000101 |
| 000001 | 00000 | 10001 | 00000 | 00000 | 000101 |

Pré-requisitos: Colocar fffffffc no registrador 5.

1) LB rt,offset(rs): rt=\*(char\*)(offset+rs) => Guarda no registrador 1 o valor do byte menos significativo do endereço 64 (64/4 = 16 = 10 em hex) = 00000010

2) LB rt,offset(rs): rt=\*(char\*)(offset+rs) => Guarda no registrador 2 o valor do byte menos significativo do endereço 68 (68/4 = 17 = 11 em hex) = 01001000

3) XORI rt,rs,imm: rt=rs^imm => Guarda no registrador 3 o valor do registrador 1 ou-exclusivo imm = 00000000000000000000000000000010 ^ 00000000000000000000000000001111 = 00000000000000000000000000001101 = D (hex) = 13 (dec)

4) DIV rs,rt: HI=rs%rt; LO=rs/rt => HI = valor do registrador 3 % valor do registrador 1 = 13%2 (dec) = 1 (dec) = 1 (hex); LO = valor do registrador 3 / valor do registrador 1 = 13/2 (dec) = 6 (dec) = 6 (hex)

5) JALR rs: rd=pc; pc=rs => Registrador 4 recebe o valor atual do PC + 4, e PC recebe valor do registrador 2.

6) BGEZAL rs,offset: r31=pc; if(rs>=0) pc+=offset\*4 => Se o valor em rs é maior ou igual a zero, o endereço da próxima instrução (PC + 4) é salvo no registrador $31 (também conhecido como $ra). O PC é alterado para PC + 4 + (offset << 2). No caso, rs = 5 e seu valor é negativo, assim, a única alteração feita é PC += 4.

7) BGEZAL rs,offset: r31=pc; if(rs>=0) pc+=offset\*4 => Se o valor em rs é maior ou igual a zero, o endereço da próxima instrução (PC + 4) é salvo no registrador $31 (também conhecido como $ra). O PC é alterado para PC + 4 + (offset << 2). No caso, rs = 0 e seu valor é 0. Logo, colocamos o valor de PC+4 em r31 e PC += 4 + 20, pois 101 << 2 = 10100 = 20 em decimal.

Estado d: 000001000100000000010011

Estado e: 000001000000000110010011

Estado f: 000010001000000001000001