|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sintaxe** | **Função (Descrição da Operação)** |  | **Formato Binário** | **Tipo** |
| **Instruções Aritméticas e Lógicas** |  |  |  |  |
| ADD Rd, Rs1, Rs2 | Rd <- Rs1 + Rs2 | 0000 | 0000 dd s1s1 s2s2 00 | R-Type |
| SUB Rd, Rs1, Rs2 | Rd <- Rs1 - Rs2 | 0000 | 0000 dd s1s1 s2s2 01 | R-Type |
| AND Rd, Rs1, Rs2 | Rd <- Rs1 & Rs2 | 0000 | 0000 dd s1s1 s2s2 10 | R-Type |
| OR Rd, Rs1, Rs2 | `Rd <- Rs1 | Rs2` | 0000 | 0000 dd s1s1 s2s2 11 |
| ADDI Rd, Rs1, imm | Rd <- Rs1 + SignExtend(imm4) | 0001 | 0001 dd s1s1 iiii | I-Type |
| SLT Rd, Rs1, Rs2 | Rd <- (Rs1 < Rs2) ? 1 : 0 | 0101 | 0101 dd s1s1 s2s2 xx | R-Type |
| NOT Rd, Rs1 | Rd <- ~Rs1 | 0110 | 0110 dd s1s1 xxxx | R-Type |
| **Instruções de Acesso à Memória** |  |  |  |  |
| LW Rd, imm(Rs1) | Rd <- Memoria[Rs1 + SignExtend(imm4)] | 0010 | 0010 dd s1s1 iiii | I-Type |
| SW Rs2, imm(Rs1) | Memoria[Rs1 + SignExtend(imm4)] <- Rs2 | 0011 | 0011 s1s1 s2s2 iiii | S-Type |
| **Instruções de Desvio e Salto** |  |  |  |  |
| BEQ Rs1, Rs2, off | Se (Rs1==Rs2) então PC <- PC+1+SignExtend(off4) | 0100 | 0100 s1s1 s2s2 oooo | B-Type |
| BNE Rs1, Rs2, off | Se (Rs1!=Rs2) então PC <- PC+1+SignExtend(off4) | 1010 | 1010 s1s1 s2s2 oooo | B-Type |
| JUMP offset | PC <- PC + 1 + SignExtend(offset8) | 1001 | 1001 oooooooo | J-Type |
| JAL Rd, offset | Rd <- PC + 1; PC <- PC+1+SignExtend(offset6) | 0111 | 0111 dd oooooo | JAL-Type |
| **Shifts:** |  |  |  |  |
| SLL Rd, Rs1, Rs2 | Rd <- Rs1 << Rs2 (Shift lógico para a esquerda) | 1100 | 1100 dd s1s1 s2s2 xx | R-Type |
| SRL Rd, Rs1, Rs2 | Rd <- Rs1 >> Rs2 (Shift lógico para a direita) | 1101 | 1101 dd s1s1 s2s2 xx | R-Type |