Lab 04 - Resolução

Nome: João Paulo nunes Andrade
RA: 25002703
Entrega: 3 Setembro de 2025

1) MOV R1, R0
0000 0100
PC -> 00000010
R0 -> 01011011
R1 -> 00000001
1) Fetch -> Mem <- PC | <- 00000010
READ
PC <- PC + 1 | <- 00000011
RI <- MEM[PCantigo] | <- 00000010
2) Decodificação -> Move o valor de R0 para R1
3) Busca de operando -> Não tem
4) Execução -> R1 <- R0 | <- 01011011
—----------

2) MOV R2, 16
1000 0000
PC -> 00000010
R2 -> 01011011
5) Fetch -> Mem <- PC | <- 00000010
READ
PC <- PC + 1 | <- 00000011
RI <- MEM[PCantigo] | <- 00000010
6) Decodificação -> Move o valor 16 em R2
7) B.O -> Mem <- PC | <- 00000011
READ
PC <- PC + 1 | <- 00000100
8) Execução -> R2 <- MEM[PCantigo] | <- Fh
—----------

3) MOV R3, [4]
1001 1100
PC -> 00000000
R3 -> 01011011
9) Fetch -> Mem <- PC | <- 0000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
10) Decodificação -> Passar o endereço de memória [4] para R3
11) B.O -> Mem <- PC | <- 00000001
READ
Y <- PC

PC <- PC + 1 | <- 00000010
12) Execução -> R3 <- Y | <- 00000001
—----------
4) MOV [4], R2
1010 1000
PC -> 00000000
R2 -> 00000001
1) Fetch -> Mem <- PC | <- 0000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo[+] | <- 00000000
2) Decodificação -> Passar o R3 para o endereço de memória [4]
3) B.O -> Mem <- PC | <- 00000001
READ
PC <- PC + 1 | <- 00000010
4) Execução -> MEM[PCantigo] <- R2 | <- 00000001
—----------
5) ADD R1, R2
0001 0110
PC -> 00000000
R1 -> 00010001
R2 -> 00000110
5) Fetch -> Mem <- PC | <- 0000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
6) Decodificação -> Somar R1 com R2
7) B.O -> Não tem
8) Execução -> Y <- R1 | <- 00010001
Z <- Y + R2 | <- 00010111
R1 <- Z | <- 00010111
—----------
6) AND R0, 10
1101 0000
PC -> 00000000
R0 -> 00000010
1) Fetch -> Mem <- PC | <- 0000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
2) Decodificação -> AND com R0, 10
3) B.O -> MEM <- PC | <- 00000001
READ
PC <- PC + 1 | <- 00000010
4) Execução -> Y <- R0 | <- 00000010
Z <- Y AND MEM[PCantigo] | <- 00000010 AND 00000010
R0 <- Z | <- 00000010
—----------

7) JMP 6
1110 0000
PC -> 00000000
1) Fetch -> Mem <- PC | <- 00000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
2) Decodificação -> Jump de instrução para a 6
3) B.O -> MEM <- PC | <- 00000001
READ
PC <- PC + 1 | <- 00000010
4) Execução -> PC <- MEM[PCantigo] | <- 00000001
—----------

8) PUSH R2
0101 1000
PC -> 00000000
R2 -> 00000110
SP -> 00000111
1) Fetch -> Mem <- PC | <- 00000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
2) Decodificação -> Push de R2 para o topo da pilha
3) B.O -> Não tem
4) Execução -> SP <- SP - 1 | <- 00000110
MEM[SP] <- R2 | <- 00000110
WRITE
—----------

9) POP R3
0110 1100
PC -> 00000000
R3 -> 00000110
SP -> 00000111
1) Fetch -> Mem <- PC | <- 00000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
2) Decodificação -> Leitura do topo da pilha para R3
3) B.O -> Não tem
4) Execução -> R3 <- MEM[SP] | <- 00000111
SP <- SP + 1 | <- 00001000
—----------

10) CALL 3
1111 0000
PC -> 00000000
SP -> 00000111
1) Fetch -> Mem <- PC | <- 00000000
READ

PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
2) Decodificação -> Salta para o endereço indicado e empilha o endereço de retorno
3) B.O -> MEM <- PC | <- 00000001
READ
PC <- PC + 1 | <- 00000010
4) Execução -> SP <- SP - 1 | <- 00000110
MEM[SP] <- PC | <- 00000010
WRITE
PC <- MEM[PCantigo] | <- 00000001
—----------

11) RET
0111 0000
PC -> 00000000
SP -> 00000111
1) Fetch -> Mem <- PC | <- 00000000
READ
PC <- PC + 1 | <- 00000001
RI <- MEM[PCantigo] | <- 00000000
2) Decodificação -> Passar o endereço do topo da pilha para PC como próxima instrução
3) B.O -> Não tem
4) Execução -> PC <- MEM[SP] | <- 00000111
SP <- SP + 1 | 00001000