logo each.png logo usp.png

Universidade de São Paulo

Escola de Artes, Ciências e Humanidades

EACH - USP

Douglas de Sousa Andrade - 8061692

Marco Antonio de Carvalho - 8061772

Paolo Angelo Martins Zilioti - 5633965

Tiago Martins Pinheiro - 8061841

ACH2034 - Organização de Computadores Digitais - Turma 04

Prof: Dr. João Bernardes

São Paulo

2015

**Convenções:**

ULA: 3 bits

PULO: 3 bits

END: 9 bits

**ULA**

|  |  |
| --- | --- |
| 000 → soma (+) | 100 → incrementar |
| 001 → sub (-) | 101 → decrementar |
| 010 → div (/) | 110 → invalido |
| 011 → mult (\*) | 111 → mod (%) |

**PULO -> olhar o flag ou pulo sem condição**

|  |  |
| --- | --- |
| 000 → sem condição | 100 → sem pulo |
| 001 → flag zero com 1 | 101 → sem pulo |
| 010 → flag sinal com 1 | 110 → sem pulo |
| 011 → flag zero ou sinal com 1 | 111 → sem pulo |

**ADRESS VALID**

|  |  |
| --- | --- |
| 0 → endereço invalido | 1 → endereço valido |

**READ/WRITE (só vale quando adress valid = 1)**

|  |  |
| --- | --- |
| 0 → read | 1 → write |

**OPCODE -> endereço do ciclo de execução especifico do comando**

**9 BITS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ADD | SUB | CMP | MOV |
| AX, 5 | 1001 | 1100111 | 100011011 | 101101000 |
| AX, BX | 1101 | 1101011 | 100011110 | 101101010 |
| AX, [MEM] | 10001 | 1101111 | 100100001 | 101101100 |
| AX, [BX] | 10111 | 1110101 | 100100110 | 101110001 |
| [MEM], 5 | 11110 | 1111100 | 100101100 | 101110110 |
| [MEM], BX | 100101 | 10000100 | 100110010 | 101111010 |
| [MEM], [MEM] | 101100 | 10001100 | 100111000 | 101111110 |
| [MEM], [BX] | 110111 | 10011000 | 101000001 | 110000100 |
| [AX], 5 | 1000010 | 10100100 | 101001010 | 110001010 |
| [AX], BX | 1001001 | 10101100 | 101010000 | 110001110 |
| [AX], [MEM] | 1010000 | 10110100 | 101010110 | 110010010 |
| [AX], [BX] | 1011011 | 10111111 | 101011111 | 110011000 |
|  | | | | |
|  | MUL | DIV | INC | DEC |
| BX | 11001111 | 11100100 | 11111001 | 100001010 |
| [MEM] | 11010011 | 11101001 | 11111100 | 100001101 |
| [AX] | 11011001 | 11110001 | 100000011 | 100010100 |
| 5 | 11001011 | 11011111 | -------------- | ------------- |
|  | | | | |
| JMP | 110011110 | | | |
| JZ | 110100000 | | | |
| JNZ | 110100100 | | | |
| JL | 110101000 | | | |
| JG | 110101100 | | | |
| JLE | 110110000 | | | |
| JGE | 110110100 | | | |

**REGISTER CODE - > endereço da porta do comando** (só AX, BX, CX, DX)

|  |  |  |
| --- | --- | --- |
| **Registradores** | **Portas** | |
| **Entrada** | **Saída** |
| **AX** | 5 | 9 |
| **BX** | 6 | 10 |
| **CX** | 7 | 11 |
| **DX** | 8 | 12 |

**Comando horizontal:**

00000000000000000000000000 000 0 0 000 000000000

<mov. dados internos > <ULA> <Adress Valid> <R/W> <PULO> < end. controle>

**Ciclo de Busca:**

**T1**: MAR ← PC

**T2**: MBR ← [MAR]

AC ← PC++

**T3**: IR ← MBR

PC ← AC

**T1:** 01100000000000000000000000 110 0 0 100 000000000

**T2:** 01000000000000000001010010 100 1 0 111 000000000

10000000000000000000100101 110 0 0 100 000000000

**T3:** 00001000000001000000000000 110 0 0 000 000000111

**Ciclo de Execução:**

**T1:** 00000000000000000000000000 110 0 0 000 <OPCODE>

**Ciclo de Execução do ADD:**

**Os 12 Tipos:**

ADD AX, 5

ADD AX, BX

ADD AX, [mem.]

ADD AX, [BX]

ADD [mem.], 5

ADD [mem.], BX

ADD [mem.], [mem.]

ADD [mem.], [BX]

ADD [AX], 5

ADD [AX], BX

ADD [AX], [mem.]

ADD [AX], [BX]

**Ex: ADD AX, 5**

**T1:** X <- [P1]

**T2:** ULA <- P2 (CODIGO ULA = SOMA)

**T2:** [P1] <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000010001000000 000 0 0 100 000000000

**T3:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: ADD AX, BX**

**T1:** X <- [P1]

**T2:** ULA <- [P2] (CODIGO ULA = SOMA)

**T3:** [P1] <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000001000000 000 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: ADD AX, [mem.]**

**T1:** MAR <- P2

**T2:** X <- [P1]

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SOMA)

**T4:** [P1] <- AC

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000010010010 110 1 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 000 0 0 100 000000000

**T4:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: ADD AX, [BX]**

**T1:** X <- [P1]

**T2:** MAR <- [P2]

**T3:** MBR <- [MAR]

**T4:** ULA <- MBR (CODIGO ULA = SOMA)

**T5:** [P1] <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T4:** 00001000000000000001000000 000 0 0 100 000000000

**T5:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: ADD [MEM.], 5**

**T1:** MAR <- P1

**T2:** X <- P2

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SOMA)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000010010010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 000 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: ADD [mem.], BX**

**T1:** MAR <- P1

**T2:** X <- [P2]

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SOMA)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000010010010 110 1 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 000 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: ADD [mem.], [mem.]**

**T1:** MAR <- P2

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- P1

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SOMA)

**T7:**  MBR <- AC

**T8:** [MAR] <- MBR

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000100000000 110 0 0 100 000000000

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 000 0 0 100 000000000

**T7:** 00010000000000000000110010 110 1 1 100 000000000

**T8:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: ADD [mem.], [BX]**

**T1:** MAR <- [P2]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- P1

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SOMA)

**T7:** MBR <- AC

**T8:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000100000000 110 0 0 100 000000000

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 000 0 0 100 000000000

**T7:** 00010000000000000000110010 110 1 1 100 000000000

**T8:** 00000000000000000000001010 110 0 0 000 000000010

**ADD [AX], 5**

**T1:** MAR <- [P1]

**T2:** X <- P2

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SOMA)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000010010010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 000 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**ADD [AX], BX**

**T1:** MAR <- [P1]

**T2:** X <- [P2]

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SOMA)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000010010010 110 1 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 000 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**ADD [AX], [mem.]**

**T1:** MAR <- P2

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- [P1]

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SOMA)

**T7:** MBR <- AC

**T8:** [MAR] <- MBR

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 000 0 0 100 000000000

**T7:** 00010000000000000000110010 110 1 1 100 000000000

**T8:** 00000000000000000000001010 110 0 0 000 000000010

**ADD [AX], [BX]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- [P2]

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = ADD)

**T7:** MAR <- [P1]

**T8:** MBR <- AC

**T9:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 000 0 0 100 000000000

**T7:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T8:** 00010000000000000000110010 110 1 1 100 000000000

**T9:** 00000000000000000000001010 110 0 0 000 000000010

**Ciclo de Execução SUB**

**Os 12 tipos:**

SUB AX, 5

SUB AX, BX

SUB AX, [mem.]

SUB AX, [BX]

SUB [mem.], 5

SUB [mem.], BX

SUB [mem.], [MEM]

SUB [mem.], [BX]

SUB [AX], 5

SUB [AX], BX

SUB [AX], [mem.]

SUB [AX], [BX]

**Ex: SUB AX, 5**

**T1:** X <- [P1]

**T2:** ULA <- P2 (CODIGO ULA = SUB)

**T2:** [P1] <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000010001000000 001 0 0 100 000000000

**T3:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: SUB AX, BX**

**T1:** X <- [P1]

**T2:** ULA <- [P2] (CODIGO ULA = SUB)

**T3:** [P1] <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000001000000 001 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: SUB AX, [mem.]**

**T1:** MAR <- P2

**T2:** X <- [P1]

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SUB)

**T4:** [P1] <- AC

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000010010010 110 1 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 001 0 0 100 000000000

**T4:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: SUB AX, [BX]**

**T1:** X <- [P1]

**T2:** MAR <- [P2]

**T3:** MBR <- [MAR]

**T4:** ULA <- MBR (CODIGO ULA = SUB)

**T5:** [P1] <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T4:** 00001000000000000001000000 001 0 0 100 000000000

**T5:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: SUB [mem.], 5**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- P2 (CODIGO ULA = SUB)

**T5:** MBR <- AC

**T6:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000010001000000 001 0 0 100 000000000

**T5:** 00010000000000000000110010 110 1 1 100 000000000

**T6:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [mem.], BX**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- [P2] (CODIGO ULA = SUB)

**T5:** MBR <- AC

**T6:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000000001000000 001 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00010000000000000000110010 110 1 1 100 000000000

**T6:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [mem.], [mem.]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- P2

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T7:** MAR <- P1

**T8:** MBR <- AC

**T9:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000010000000000 110 0 0 100 000000000

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 100 000000000

**T7:** 00100000000000000100000000 110 0 0 100 000000000

**T8:** 00010000000000000000110010 110 1 1 100 000000000

**T9:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [mem.], [AX]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- [P2]

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T7:** MAR <- P1

**T8:** MBR <- AC

**T9:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 100 000000000

**T7:** 00100000000000000100000000 110 0 0 100 000000000

**T8:** 00010000000000000000110010 110 1 1 100 000000000

**T9:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [AX], 5**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- P2 (CODIGO ULA = SUB)

**T5:** MBR <- AC

**T6:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000010001000000 001 0 0 100 000000000

**T5:** 00010000000000000000110010 110 1 1 100 000000000

**T6:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [AX], BX**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- [P2] (CODIGO ULA = SUB)

**T5:** MBR <- AC

**T6:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000000001000000 001 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00010000000000000000110010 110 1 1 100 000000000

**T6:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [AX], [mem.]**

**T1:** MAR <- P2

**T2:** MBR <- [MAR]

**T3:** ULA <- MBR

**T4:** MAR <- [P1]

**T5:** MBR <- [MAR]

**T6:** X <- MBR (CODIGO ULA = SUB)

**T7:** MBR <- AC

**T8:** [MAR] <- MBR

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000010000000 001 0 0 100 000000000

**T7:** 00010000000000000000110010 110 1 1 100 000000000

**T8:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: SUB [AX], [BX]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- [P2]

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T7:** MAR <- [P1]

**T8:** MBR <- AC

**T9:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 100 000000000

**T7:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T8:** 00010000000000000000110010 110 1 1 100 000000000

**T9:** 00000000000000000000001010 110 0 0 000 000000010

**Ciclo de Execução MUL**

**Os 4 Tipos:**

MUL 5

MUL BX

MUL [mem.]

MUL [AX]

**Ex: MUL 5**

**T1:** X <- P1

**T2:** ULA <- AX (ULA = MULTIPLICAR)

**T3:** AX <- AC

**T1:** 00000000000000000110000000 110 0 0 100 000000000

**T2:** 00000000010000000001000000 011 0 0 100 000000000

**T3:** 00000100000000000000100000 110 0 0 000 000000010

**Ex: MUL BX**

**T1:** X <- [P1]

**T2:** ULA <- AX (ULA = MULTIPLICAR)

**T3:** AX <- AC

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000010000000001000000 011 0 0 100 000000000

**T3:** 00000100000000000000100000 110 0 0 000 000000010

**Ex: MUL [mem.]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

X <- AX

**T3:** ULA <- MBR (ULA = MULTIPLICAR)

**T4:** AX <- AC

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000010000000010000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 011 0 0 100 000000000

**T4:** 00000100000000000000100000 110 0 0 000 000000010

**Ex: MUL [AX]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

X <- AX

**T3:** ULA <- MBR (ULA = MULTIPLICAR)

**T4:** AX <- AC

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000010000000010000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 011 0 0 100 000000000

**T4:** 00000100000000000000100000 110 0 0 000 000000010

**Ciclo de Execução DIV**

**Os 4 Tipos:**

DIV 5

DIV BX

DIV [mem.]

DIV [AX]

**Ex: DIV 5**

**T1:** X <- AX

**T2:** ULA <- P1 (ULA = DIVIDIR)

**T3:** AX <- AC (ULA = MOD/RESTO)

**T4:** DX <- AC

**T1:** 00000000010000000010000000 110 0 0 100 000000000

**T2:** 00000000000000000101000000 010 0 0 100 000000000

**T3:** 00000100000000000000100000 111 0 0 100 000000000

**T4:** 00000000100000000000100000 110 0 0 000 000000010

**Ex: DIV BX**

**T1:** X <- AX

**T2:** ULA <- [P1] (ULA = DIVIDIR)

**T3:** AX <- AC (ULA = MOD/RESTO)

**T4:** DX <- AC

**T1:** 00000000010000000010000000 110 0 0 100 000000000

**T2:** 00000000000000000001000000 010 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T3:** 00000100000000000000100000 111 0 0 100 000000000

**T4:** 00000000100000000000100000 110 0 0 000 000000010

**Ex: DIV [mem.]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- AX

**T4:** ULA <- MBR (ULA = DIVIDIR)

**T5:** AX <- AC (ULA = MOD/RESTO)

**T6:** DX <- AC

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00000000010000000010000000 110 0 0 100 000000000

**T4:** 00001000000000000001000000 010 0 0 100 000000000

**T5:** 00000100000000000000100000 111 0 0 100 000000000

**T6:** 00000000100000000000100000 110 0 0 000 000000010

**Ex: DIV [AX]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- AX

**T4:** ULA <- MBR (ULA = DIVIDIR)

**T5:** AX <- AC (ULA = MOD/RESTO)

**T6:** DX <- AC

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00000000010000000010000000 110 0 0 100 000000000

**T4:** 00001000000000000001000000 010 0 0 100 000000000

**T5:** 00000100000000000000100000 111 0 0 100 000000000

**T6:** 00000000100000000000100000 110 0 0 000 000000010

**Ciclo de Execução INC**

**Os 3 Tipos:**

INC AX

INC [mem.]

INC [AX]

**Ex: INC AX**

**T1:** ULA <- [P1] (ULA = INC)

**T2:** [P1] <- AC

**T1:** 00000000000000000001000000 100 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: INC [mem.]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** ULA <- MBR (ULA = INC)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 100 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: INC [AX]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** ULA <- MBR (ULA = INC)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 100 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**Ciclo de Execução DEC**

**Os 3 Tipos:**

DEC AX

DEC [mem.]

DEC [AX]

**Ex: DEC AX**

**T1:** ULA <- [P1] (ULA = DEC)

**T2:** [P1] <- AC

**T1:** 00000000000000000001000000 101 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000100000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: DEC [mem.]**

**T1:** MAR <- P1

**T2:**  MBR <- [MAR]

**T3:** ULA <- MBR (ULA = DEC)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 101 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: DEC [AX]**

**T1:** MAR <- [P1]

**T2:**  MBR <- [MAR]

**T3:** ULA <- MBR (ULA = DEC)

**T4:** MBR <- AC

**T5:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 101 0 0 100 000000000

**T4:** 00010000000000000000110010 110 1 1 100 000000000

**T5:** 00000000000000000000001010 110 0 0 000 000000010

**Ciclo de Execução do CMP**

**Os 12 Tipos:**

CMP AX, 5

CMP AX, BX

CMP AX, [mem.]

CMP AX, [BX]

CMP [mem.], 5

CMP [mem.], BX

CMP [mem.], [mem.]

CMP [mem.], [BX]

CMP [AX], 5

CMP [AX], BX

CMP [AX], [mem.]

CMP [AX], [BX]

**Ex: CMP AX, 5**

**T1:** X <- [P1]

**T2:** ULA <- P2 (CODIGO ULA = SUB)

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000010001000000 001 0 0 000 000000010

**Ex: CMP AX, BX**

**T1:** X <- [P1]

**T2:** ULA <- [P2] (CODIGO ULA = SUB)

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000001000000 001 0 0 000 000000010

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**Ex: CMP AX, [mem.]**

**T1:** MAR <- P2

**T2:** X <- [P1]

MBR <- [MAR]

**T3:** ULA <- MBR (CODIGO ULA = SUB)

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000010010010 110 1 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000001000000 001 0 0 000 000000010

**Ex: CMP AX, [BX]**

**T1:** X <- [P1]

**T2:** MAR <- [P2]

**T3:** MBR <- [MAR]

**T4:** ULA <- MBR (CODIGO ULA = SUB)

**T1:** 00000000000000000010000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T4:** 00001000000000000001000000 001 0 0 100 000000000

**Ex: CMP [mem.], 5**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- P2 (CODIGO ULA = SUB)

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000010001000000 001 0 0 000 000000010

**Ex: CMP [mem.], BX**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- [P2] (CODIGO ULA = SUB)

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000000001000000 001 0 0 000 000000010

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**Ex: CMP [mem.], [mem.]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- P2

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000010000000000 110 0 0 100 000000000

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 000 000000010

**Ex: CMP [mem.], [AX]**

**T1:** MAR <- P1

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- [P2]

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 000 000000010

**Ex: CMP [AX], 5**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- P2 (CODIGO ULA = SUB)

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000010001000000 001 0 0 000 000000010

**Ex: CMP [AX], BX**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** ULA <- [P2] (CODIGO ULA = SUB)

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00000000000000000001000000 001 0 0 000 000000010

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**Ex: CMP [AX], [mem.]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- P2

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000010000000000 110 0 0 100 000000000

**T5:** 00000000000000000000010010 110 0 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 000 000000010

**Ex: CMP [AX], [BX]**

**T1:** MAR <- [P1]

**T2:** MBR <- [MAR]

**T3:** X <- MBR

**T4:** MAR <- [P2]

**T5:** MBR <- [MAR]

**T6:** ULA <- MBR (CODIGO ULA = SUB)

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000010000000 110 0 0 100 000000000

**T4:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T5:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T6:** 00001000000000000001000000 001 0 0 000 000000010

**Ciclo de Execução do MOV**

**Os 12 Tipos:**

MOV AX, 5

MOV AX, BX

MOV AX, [mem.]

MOV BX, [AX]

MOV [mem.], 5

MOV [mem.], BX

MOV [mem.], [mem.]

MOV [mem.], [AX]

MOV [AX], 5

MOV [AX], BX

MOV [AX], [mem.]

MOV [BX], [AX]

**Ex: MOV AX, 5**

**T1:** [P1] <- P2

**T1:** 00000000000000010000000000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: MOV AX, BX**

**T1:** [P1] <- [P2]

**T1:** 00000000000000000000000000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**Ex: MOV AX, [mem.]**

**T1:** MAR <- P2

**T2:** MBR <- [MAR]

**T3:** [P1] <- MBR

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000000000000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: MOV AX, [BX]**

**T1:** MAR <- [P2]

**T2:** MBR <- [MAR]

**T3:** [P1] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00001000000000000000000000 110 0 0 000 000000010

-> o bit correspondente ao end de entrada de P1 fica com 1 (fazer no Java)

**Ex: MOV [mem.], 5**

**T1:** MAR <- P1

**T2:** MBR <- P2

**T3:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00010000000000010000010010 110 1 1 100 000000000

**T3:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [mem.], BX**

**T1:** MAR <- P1

**T2:** MBR <- [P2]

**T3:** [MAR] <- MBR

**T1:** 00100000000000000100000000 110 0 0 100 000000000

**T2:** 00010000000000000000010010 110 1 1 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [mem.], [mem.]**

**T1:** MAR <- P2

**T2:** MBR <- [MAR]

MAR <- P1

**T3:** [MAR] <- MBR

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00100000000000000100000101 110 0 0 100 000000000

**T3:** 00000000000000000000010010 110 1 1 100 000000000

00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [mem.], [AX]**

**T1:** MAR <- [P2]

**T2:** MBR <- [MAR]

MAR <- P1

**T3:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00100000000000000100000101 110 0 0 100 000000000

**T3:** 00000000000000000000010010 110 1 1 100 000000000

00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [AX], 5**

**T1:** MAR <- [P1]

**T2:** MBR <- P2

**T3:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00010000000000010000010010 110 1 1 100 000000000

**T3:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [AX], BX**

**T1:** MAR <- [P1]

**T2:** MBR <- [P2]

**T3:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T2:** 00010000000000000000010010 110 1 1 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T3:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [AX], [mem.]**

**T1:** MAR <- P2

**T2:** MBR <- [MAR]

**T3:** MAR <- [P1]

**T4:** [MAR] <- MBR

**T1:** 00100000000000010000000000 110 0 0 100 000000000

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00100000000000000000010010 110 1 1 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T4:** 00000000000000000000001010 110 0 0 000 000000010

**Ex: MOV [BX], [AX]**

**T1:** MAR <- [P2]

**T2:** MBR <- [MAR]

**T3:** MAR <- [P1]

**T4:** [MAR] <- MBR

**T1:** 00100000000000000000000000 110 0 0 100 000000000

-> o bit correspondente ao end de saida de P2 fica com 1 (fazer no Java)

**T2:** 00000000000000000000010010 110 1 0 100 000000000

00000000000000000000000101 110 0 0 100 000000000

**T3:** 00100000000000000000010010 110 1 1 100 000000000

-> o bit correspondente ao end de saida de P1 fica com 1 (fazer no Java)

**T4:** 00000000000000000000001010 110 0 0 000 000000010

**Execução dos JUMPS**

**Ex: JMP 5**

**T1:** PC <- P1

**T1:** 10000000000000000100000000 110 0 0 000 000000010

**Ciclo de Execução JZ**

**Ex: JZ 5**

**T1:** PC <- P1 SE FLAG ZERO = 1

**T1:** 00000000000000000000000000 110 0 0 001 110001101

**T2:** 00000000000000000000000000 110 0 0 000 000000010

**T3:** 10000000000000000100000000 110 0 0 000 000000010

**Ciclo de Execução com JNZ**

**T1:** PC <- P1 SE FLAG ZERO = 0

**T2:** PULO PRO CICLO DE BUSCA

**T1:** 00000000000000000000000000 110 0 0 001 110100110

**T2:** 10000000000000000100000000 110 0 0 000 000000010

**T3:** 00000000000000000000000000 110 0 0 000 000000010

**Ciclo de Execução com JL**

**T1:** PC <- P1 SE FLAG SINAL = 1

**T2:** PULO PRO CICLO DE BUSCA

**T1:** 00000000000000000000000000 110 0 0 010 110101010

**T2:** 00000000000000000000000000 110 0 0 000 000000010

**T3:** 10000000000000000100000000 110 0 0 000 000000010

**Ciclo de Execução com JG**

**T1:** PC <- P1 SE FLAG SINAL = 0 E FLAG ZERO = 0

**T2:** PULO PRO CICLO DE BUSCA

**T1:** 00000000000000000000000000 110 0 0 011 110101110

**T2:** 10000000000000000100000000 110 0 0 000 000000010

**T3:** 00000000000000000000000000 110 0 0 000 000000010

**Ciclo de Execução com JLE**

**T1:** PC <- P1 SE FLAG SINAL = 1 OU FLAG ZERO = 1

**T2:** PULO PRO CICLO DE BUSCA

**T1:** 00000000000000000000000000 110 0 0 011 110110010

**T2:** 00000000000000000000000000 110 0 0 000 000000010

**T3:** 10000000000000000100000000 110 0 0 000 000000010

**Ciclo de Execução do com JGE**

**T1:** PC <- P1 SE FLAG SINAL = 0

**T2:** PULO PRO CICLO DE BUSCA

**T1:** 00000000000000000000000000 110 0 0 010 110110110

**T2:** 10000000000000000100000000 110 0 0 000 000000010

**T3:** 00000000000000000000000000 110 0 0 000 000000010