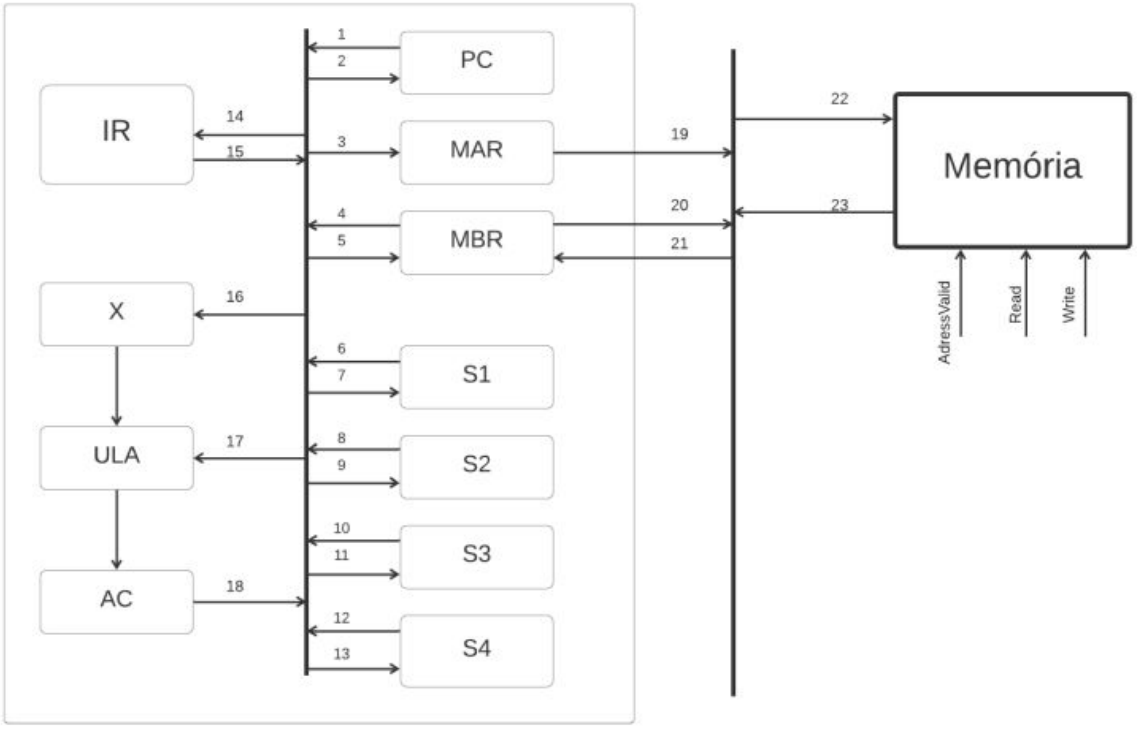

OPCODES (5 bits – 16 opções)

data - 00001

.word - 00010

.text: - 00011

li (load immediate) - 00100

sw (store word) – 00101

move – 00110

add (constante) – 00111

add (registrador) – 01000

sub (constante) - 01001

sub (registrador) - 01010

beq (branch on equal) – 01011

bne (branch on not equal) - 01100

j (jump) – 01101

slt (set on less than) – 01110

la - 01111

lw (load word) - 10000

CONTROLE DE MEMÓRIA (3 bits – 3 opções)

Address Valid – 100

Read – 010

Write – 001

CONDIÇÃO DE PULO (4 bits – 5 opções)

Não pule: 0000

Pule sempre: 0001

Pule se os números forem iguais: 0010

Pule se um número for maior que o outro: 0100

Pule se um número for menor que o outro: 1000

AS PALAVRAS HORIZONTAIS SERÃO FORMADAS POR:

Sinais de Controle + OPCODE + Controle de Memória + Condição de Pulo + Endereço para Pular

Ou seja:

23 bits + 5 bits + 3 bits + 4 bits + Depende do número de linhas do microprograma

Ciclo de busca com barramento compartilhado

t1: MAR <- (PC) **[2, 3, 17]** 01100000000000001000000 00000 000 0000 0

ULA <- (PC)

t2: Memória <- (MAR) **[18, 22, av, r]** 00000000000000000100010 00000 110 0000 0

t3: MBR <- (Memória) **[23, 21, 18, 2]** 01000000000000000100101 00000 000 0000 0

PC <- (AC) = ULA = (PC + 1)

t4: IR <- (MBR) **[4, 14]** 00010000000001000000000 00000 000 0001 endereço

Ciclos de Execução

Ciclo data

.data

t1: Memória <- MAR **[19, 22, r, av]** 00000000000000000010010 00001 110 0000 0

t2: MBR <- Memória **[23, 21, w]** 00000000000000000000101 00001 001 0001 ciclo busca

Ciclo .word

.word 4, 0, 1, 7, 1

t1: MBR <- IR **[15, 5]** 00001000000000100000000 00010 001 0000 0

t2: MEMÓRIA (write) <- MBR **[20, 22]** 00000000000000000001010 00010 001 0001 end ciclo de busca

Ciclo li

li $s1, 0

t1: MBR <- (Memória/read) **[23, 21]** 00000000000000000000101 00100 010 0000 0

t2: S1~4 <- MBR **[4, 7]** 00010010000000000000000 00100 001 0001 endereço busca

Ciclo lw

lw $s1, 0 ($s2)

t1: MBR <- (Memória/read) **[23, 21]** 00000000000000000000101 10000 010 0000 0

t2: S1~4 <- (MBR) **[4, 7]** 00010010000000000000000 10000 001 0001 endereço busca

Ciclo sw

sw $s1, $s2 ($s3)

guarda $s1 = array[i+1] em $s2 = array[i]

com $s3 = i

t1: MBR <- S1~4 **[6, 5]** 00001100000000000000000 00101 001 0000 0

t2: (Memória/write) <- MBR **[20, 22]** 00000000000000000001010 00101 001 0001 endereço

Ciclo move

move $s1 $s2

t1: S1 <- S2 **[8, 7]** 00000011000000000000000 00110 001 0001 endereço

Ciclo add (com uma constante)

ADD $s1, $s2, constante

t1: MAR <- (IRend) **[15,3]** 00100000000000100000000 00000 000 0000 0

t2: Memória(read) <- MAR **[18, 22, av, r]** 00000000000000000100010 00000 110 00000 0

t3: MBR <- (Memória) **[23, 21]** 00000000000000000000101 00000 101 0000 0

t4: ULA <- MBR[X] **[4, 17]** 00010000000000001000000 00000 000 0000 0

t5: ULA <- S2 **[8, 17]** 00000001000000001000000 00111 000 0000 0

t6: S1 <- AC **[18, 7]** 00000010000000000100000 00000 000 0001 endereço

Ciclo add (entre registradores)

ADD $s1, $s2, $s3

t1: X <- S2 **[8, 16]** 00000001000000010000000 01000 001 0000 0

t2: ULA <- S3 **[10, 17]** 00000000010000001000000 01000 001 0000 0

t3: S1 <- AC **[18, 7]** 00000010000000000100000 01000 001 0001 endereço

Ciclo sub (com uma constante)

SUB $s1, $s2, constante

t1: MAR <- (IRend) **[15,3]** 00100000000000100000000 00000 000 0000 0

t2: Memória(read) <- MAR **[18, 22, av, r]** 00000000000000000100010 00000 110 00000 0

t3: MBR <- (Memória) **[23, 21]** 00000000000000000000101 00000 101 0000 0

t4: ULA <- MBR[X] **[4, 17]** 00010000000000001000000 00000 000 0000 0

t5: ULA <- S2 **[8, 17]** 00000001000000001000000 00111 000 0000 0

t6: S1 <- AC **[18, 7]** 00000010000000000100000 00000 000 0001 endereço

Ciclo sub (entre registradores)

SUB $s1, $s2, $s3

t1: X <- S2 **[8, 16]** 00000001000000010000000 01010 001 0000 0

t2: ULA <- S3 **[10, 17]** 00000000010000001000000 01010 001 0000 0

t3: S1 <- AC **[18, 7]** 00000010000000000100000 01010 001 0001 endereço

Ciclo beq (branch on equal)

beq $s3,$s4, L2

t1: X <- S3 **[10, 16]** 00000000010000010000000 01011 001 0000 0

t2: ULA <- S4 **[12, 17]** 00000000000100001000000 01011 001 0000 0

t3: PC <- AC **[18, 2]** 00000000000001000100000 01011 001 0010 endereço l2

se s3 e s4 forem iguais, pula pro endereço da linha L2

Ciclo bne (branch on not equal)

bne $s1,$s2, else

t1: X <- S1 **[6, 16]** 00000100000000010000000 01100 001 0000 0

t2: ULA <- S2 **[8, 17]** 00000001000000001000000111 01100 001 0000 0

t3: PC <- AC **[18, 2]** 00000000000001000100000 01100 001 0010 endereço else

se s1 e s2 não forem iguais, pula pro endereço da linha else

Ciclo j (jump)

j exit

t1: MAR <- IR **[15, 3]** 00100000000000100000000 01101 000 0000 0

t2: Memória <- MAR **[19, 22]** 00000000000000000010010 01101 000 0000 0

t3: MBR <- Memória **[23, 21, av]** 00000000000000000000101 01101 000 0000 0

t4: PC <- MBR **[4, 2]** 01010000000000000000000 01101 001 0001 endereço

Ciclo slt (set less than)

slt $s1,$s2,$s3

t1: X <- S2 **[8, 16]** 00000001000000010000000 01110 001 0000 0

t2: ULA <- S3 **[10, 17]** 00000000010000001000000 01110 001 0000 0

t3: S1 <- AC **[18, 7]** 00000010000000000100000 01110 001 1000 endereço

Ciclo la

la $s1, 7

t1: MAR <- IR **[15, 3]** 00100000000000100000000 01101 010 0001 endereço exit

t2: Memória <- MAR **[18, 22]** 00000000000000000100010 00000 110 00000 0

t3: MBR <- Memória(write) **[21,23, w]** 00000000000000000000101 01111 001 00000 0

t4: S1~4 <- (MBR) **[4, 7]** 00010010000000000000000 01111 000 00000 end busca