UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
DISCIPLINA: SISTEMAS DIGITAIS
Prof. Fernanda Lima Kastensmidt

Aula REMOTA NEANDER – parte II

Giordano Souza de Paula - 308054

Projeto do Processador Neander em VHDL

Instrução	Comentário
NOP	nenhuma operação
STA end	MEM(end) ← AC
LDA end	AC← MEM(end)
ADD end	AC← MEM(end) + AC
OR end	AC← MEM(end) OR AC
AND end	AC← MEM(end) AND AC
NOT	AC← NOT AC
JMP end	PC← end
JN end	IF N=1 THEN PC ← end
JZ end	IF Z=1 THEN PC ← end

Código	Instrução	Comentário
0000	NOP	nenhuma operação
0001	STA end	armazena acumulador - (store)
0010	LDA end	carrega acumulador - (load)
0011	ADD end	soma
0100	OR end	"ou" lógico
0101	AND end	"e" lógico
0110	NOT	inverte (complementa) acumulador
1000	JMP end	desvio incondicional - (jump)
1001	JN end	desvio condicional - (jump on negative)
1010	JZ end	desvio condicional - (jump on zero)
1111	HLT	término de execução - (halt)

Projeto do programa do NEANDER

A memória BRAM deve ser inicializada com o **arquivo .coe** que contem o programa projetado em binário ou hexadecimal. Use o simulador do Neander ou Hidra para gerar o binário/hexadecimal.

Projete os seguintes programas no Neander e apresente o .coe a ser usado.

SOMA MATRIZES 3x3

```
;CONSTANTES
;SOMA A MATRIZ A E A MATRIZ B
;COLOCA O RESULTADO NA MATRIZ R
ORG 54
index:
        DB 9
matrizA11:
        DB 1
matrizA12:
        DB 1
matrizA13:
        DB 1
matrizA21:
        DB 1
matrizA22:
        DB 1
matrizA23:
        DB 1
matrizA31:
        DB 1
matrizA32:
        DB 1
matrizA33:
        DB 1

matrizB11:
        DB 2
matrizB12:
        DB 2
matrizB13:
        DB 2
matrizB21:
        DB 2
matrizB22:
        DB 2
matrizB23:
        DB 2
matrizB31:
        DB 2
matrizB32:
        DB 2
matrizB33:
        DB 2
```

```
matrizR11:
        DB 2
matrizR12:
        DB 2
matrizR13:
        DB 2
matrizR21:
        DB 2
matrizR22:
        DB 2
matrizR23:
        DB 2
matrizR31:
        DB 2
matrizR32:
        DB 2
matrizR33:
        DB 2
fim:
        HLT

;PROGRAMA PRINCIPAL
ORG 0
        LDA matrizA11
        ADD matrizB11
        STA matrizR11

        LDA matrizA12
        ADD matrizB12
        STA matrizR12

        LDA matrizA13
        ADD matrizB13
        STA matrizR13

        LDA matrizA21
        ADD matrizB21
        STA matrizR21

        LDA matrizA22
        ADD matrizB22
        STA matrizR22

        LDA matrizA23
        ADD matrizB23
        STA matrizR23

        LDA matrizA31
```

```
        ADD matrizB31
        STA matrizR31

        LDA matrizA32
        ADD matrizB32
        STA matrizR32

        LDA matrizA33
        ADD matrizB33
        STA matrizR33
```

Arquivo .coe

```
memory_initialization_radix=10;
memory_initialization_vector=32,55,48,64,16,73,32,56,48,65,16,74,32,57,48,66,16,75,32,58,4
8,67,16,76,32,59,48,68,16,77,32,60,48,69,16,78,32,61,48,70,16,79,32,62,48,71,16,80,32,63,48,
72,16,81,9,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,0,0,0,0,0,0,0,0,0,240;
```

Multiplicação de dois valores por soma sucessiva

```
;CONSTANTES
ORG 127
zero:
        DB 0
menos1:
        DB -1
dois:
        DB 2
loopindex:
        DB 0
valor1:
        DB 2 ;exemplo
valor2:
        DB 3 ;exemplo
resultado:
        DB 0
fim:
        HLT

;PROGRAMA PRINCIPAL
ORG 1
        LDA valor1
        STA loopindex
somaloop:
        LDA loopindex
        JZ fim
        ADD menos1
```

```
STA loopindex
LDA resultado
ADD valor2
STA resultado
JMP somaloop
```

Arquivo .coe

```
memory_initialization_radix=10;
memory_initialization_vector=0,32,24,16,23,32,23,160,27,48,21,16,23,32,26,48,25,16,26,128,
4,0,255,2,0,2,3,0,240;
```