Prova P2 **Disciplina: Organização e Arquitetura de Computadores I**

T148 **Professor: Ney Laert Vilar Calazans**

Aluno: 28/junho/2006

1. [3,5 pontos] Complete a primeira tabela abaixo com o código objeto das instruções do trecho de programa dado. A seguir, preencha a tabela de microssimulação, mostrando o resultado de executar 1 vez cada instrução do trecho de programa na organização Cleópatra vista em aula mostrando o efeito de cada microinstrução componente destas instruções. O conteúdo dos registradores em cada linha corresponde ao resultado após executar a microinstrução daquela linha. Assuma que todos os registradores contenham inicialmente o valor 00H. A primeira linha está preenchida a título de exemplo. Usa-se o caracter "-" para simbolizar *don´t-cares*. O número de linhas da segunda tabela é ilustrativo, pode haver mais ou menos microinstruções. Para acelerar o preenchimento da tabela de microssimulação, somente escreva um valor para um registrador quando este mudar no ciclo de relógio correspondente à linha da tabela.

Memória de Programa

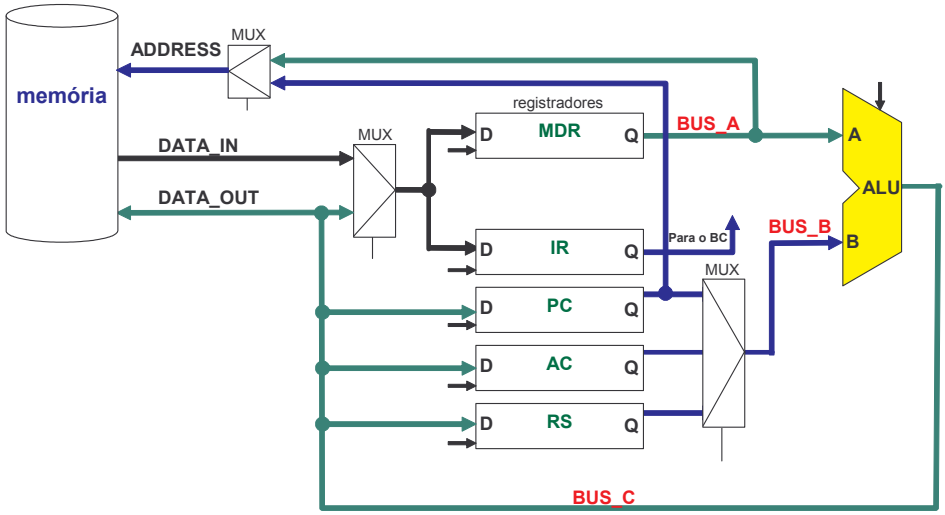
Endereço de Memória	Rótulos	Mnemônico/Dado	Código objeto
00H	Main:	LDA #080H	
02H	Xuxu:	ADD 37H	
04H		JV Xuxu,R	

Memória de Dados

37H		DB #08H	

CK	DATAIN	DATAOUT	MDR	MAR	IR	PC	AC	n/z/c/v	*Microinstrução*
1	-	-	00	00	00	00	00	0/0/0/0	MAR ← PC
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23									
24									
25									
26									
27									

2. **[4 pontos] Na Figura abaixo é proposta uma organização alternativa para o Bloco de Dados da arquitetura Cleópatra, havendo como modificações em relação à organização original: (i) o acesso à porta A da ALU se dá apenas pelo registrador MDR; (ii) o MAR desapareceu; (iii) o barramento de endereços pode ser o BUS_A ou diretamente a saída do PC; (iv) o bloco de decodificação de leitura de registradores foi eliminado; (v) o bloco de decodificação de escrita de registradores foi alterado para dar suporte a esta nova organização (não mostrado na Figura). Considere que: (1) a organização dos *flags* não foi alterada (não se detalha eles na Figura), (2) o controle de escrita e leitura da memória continua usando os mesmos sinais (CE e RW) com a mesma interpretação, (3) a ULA ainda executa as mesmas operações, e (4) o Bloco de Controle foi devidamente alterado para dar suporte a este novo Bloco de Dados. PEDE-SE:**



a) Diga qual o número mínimo de bits da palavra de microinstrução gerada pelo Bloco de Controle para comandar o novo Bloco de Dados? Ela tem mais, menos ou o mesmo número de bits que a palavra de microinstrução original? **Justifique sua resposta!!** (**1 ponto**)

b) Descreva em linguagem de micromontagem a execução completa das instruções **ADD indireto** e **JMP relativo**, incluindo a busca e a execução de cada uma delas. **Utilize o menor número possível de ciclos de relógio permitido pela organização proposta!!** (**3 pontos**)

3. **[2,5 pontos] Para a nova organização Cleópatra, proposta na questão anterior, escreva como ficaria a descrição dos barramentos BUSA e BUSB, em VHDL. Note que não existem mais dispositivos tristate (Por quê?). Parta do código VHDL original, dado abaixo. Se necessário use novos sinais, mas descreva sua natureza e codificação usando suas próprias palavras.**

```
1. --
2. -- Trecho da descrição VHDL do Bloco de Dados do processador Cleópatra
3. -- original que implementa os barramentos de entrada da ALU (BUSA
4. -- e BUSB)
5. --
6.      busB <= mdr     when r_mdr='1' else (others=>'Z');
7.      busB <= ir_int  when r_ir='1'  else (others=>'Z');
8.      busA <= pc      when r_pc='1'  else (others=>'Z');
9.      busA <= ac      when r_ac='1'  else (others=>'Z');
10.     busA <= rs      when r_rs='1'  else (others=>'Z');
```

GABARITO

1 [3,5 pontos] **Complete a primeira tabela abaixo com o código objeto das instruções do trecho de programa dado. A seguir, preencha a tabela de microssimulação, mostrando o resultado de executar 1 vez cada instrução do trecho de programa na organização Cleópatra vista em aula mostrando o efeito de cada microinstrução componente destas instruções. O conteúdo dos registradores em cada linha corresponde ao resultado após executar a microinstrução daquela linha. Assuma que todos os registradores contenham inicialmente o valor 00H. A primeira linha está preenchida a título de exemplo. Usa-se o caracter "-" para simbolizar *don´t-cares*. O número de linhas da segunda tabela é ilustrativo, pode haver mais ou menos microinstruções. Para acelerar o preenchimento da tabela de microssimulação, somente escreva um valor para um registrador quando este mudar no ciclo de relógio correspondente à linha da tabela.**

Memória de Programa

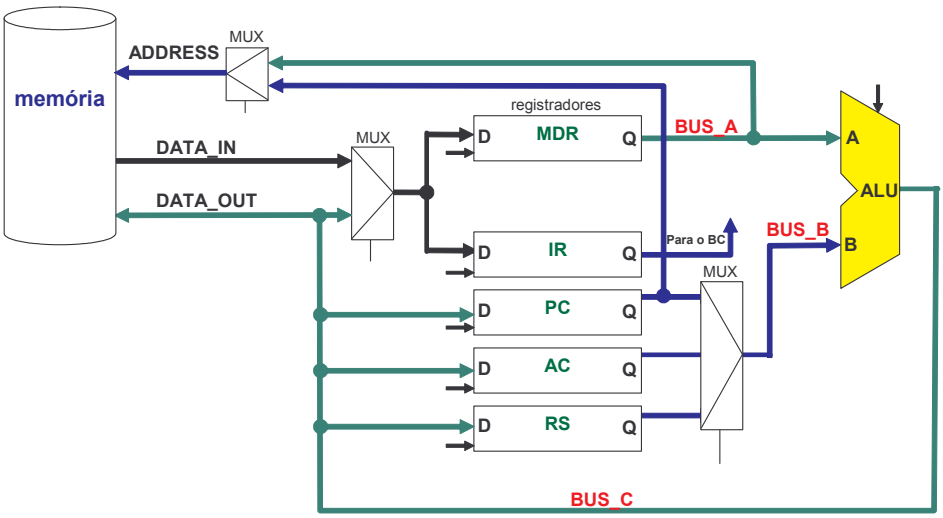
Endereço de Memória	Rótulos	Mnemônico/Dado	Código objeto
00H	Main:	LDA #080H	40 80
02H	Xuxu:	ADD 37H	54 37
04H		JV Xuxu,R	EC FC

Memória de Dados

37H		DB #08H	08

CK	DATAIN	DATAOUT	MDR	MAR	IR	PC	AC	n/z/c/v	*Microinstrução*
1	-	-	00	00	00	00	00	0/0/0/0	MAR ← PC
2	40		40			01			MDR ← PMEM(MAR); PC++
3					40				IR ← MDR
4				01					MAR ← PC
5	80		80			02			MDR ← PMEM(MAR); PC++
6							80	1/0/0/0	AC ← MDR; LNZ
7				02					MAR ← PC
8	54		54			03			MDR ← PMEM(MAR); PC++
9					54				IR ← MDR
10				03					MAR ← PC
11	37		37			04			MDR ← PMEM(MAR); PC++
12				37					MAR ← MDR
13	08		08						MDR ← PMEM(MAR)
14							88	1/0/0/0	AC ← AC+MDR; LNZ; LCV
15				04					MAR ← PC
16	EC		EC			05			MDR ← PMEM(MAR); PC++
17					EC				IR ← MDR
18				05					MAR ← PC
19	FC		FC			06			MDR ← PMEM(MAR); PC++

2 [4 pontos] **Na Figura abaixo é proposta uma organização alternativa para o Bloco de Dados da arquitetura Cleópatra, havendo como modificações em relação à organização original: (i) o acesso à porta A da ALU se dá apenas pelo registrador MDR; (ii) o MAR desapareceu; (iii) o barramento de endereços pode ser o BUS_A ou diretamente a saída do PC; (iv) o bloco de decodificação de leitura de registradores foi eliminado; (v) o bloco de decodificação de escrita de registradores foi alterado para dar suporte a esta nova organização (não mostrado na Figura). Considere que: (1) a organização dos *flags* não foi alterada (não se detalha eles na Figura), (2) o controle de escrita e leitura da memória continua usando os mesmos sinais (CE e RW) com a mesma interpretação, (3) a ULA ainda executa as mesmas operações, e (4) o Bloco de Controle foi devidamente alterado para dar suporte a este novo Bloco de Dados. PEDE-SE:**



a) Diga qual o número mínimo de bits da palavra de microinstrução gerada pelo Bloco de Controle para comandar o novo Bloco de Dados? Ela tem mais, menos ou o mesmo número de bits que a palavra de microinstrução original? **Justifique sua resposta!!** (**1 ponto**)

Resposta: São necessários exatamente um mínimo de 13 bits na palavra de microinstrução para o novo Bloco de Dados, pois ainda são necessários 3 bits para o decodificador de escrita (gerando 5 sinais para escrita individual de cada registrador e 2 sinais de escrita simultânea em PC e IR e PC e MDR, devido às microinstruções de busca IR ← PMEM (PC); PC++ e de busca de operando, MDR←PMEM(PC);PC++), 3 bits para as operações da ULA, 4 sinais individuais inalterados LNZ e LCV, CE e RW. Além disto, é necessário 1 sinal de 1 bit para controlar o mux que produz o endereço de memória e no mínimo 2 bits para controlar o mux que gera o BUSB. Este é o mesmo número de bits na palavra de microinstrução original.

c) Descreva em linguagem de micromontagem a execução completa das instruções **ADD indireto** e **JMP relativo**, incluindo a busca e a execução de cada uma delas. **Utilize o menor número possível de ciclos de relógio permitido pela organização proposta!!** (**3 pontos**)

Resposta:

Busca (1 ciclo):

IR ← PMEM(PC); PC++

Execução ADD ,I (4 ciclos):

MDR ← PMEM(PC); PC++ -- busca operando, end do end do dado

MDR ← PMEM(MDR) -- busca end do dado

MDR ← PMEM(MDR) -- busca dado

AC ← AC + MDR; LNZ; LCV

Execução JMP ,R (2 ciclos):

MDR ← PMEM(PC); PC++ -- busca operando, valor a somar ao PC

PC ← PC + MDR

3 [3 pontos] **Para a nova organização Cleópatra, proposta na questão anterior, escreva como ficaria a descrição dos barramentos BUSA e BUSB, em VHDL. Note que não existem mais dispositivos tristate (Por quê?). Parta do código VHDL original, dado abaixo. Se necessário use novos sinais, mas descreva sua natureza e codificação usando suas próprias palavras.**

Não existe necessidade de uso de dispositivos tristate porque o BUS_A vem de apenas 1 valor (MDR) e porque o BUS_B é agora saída de um MUX 3:1.

```
1. --
2. -- Trecho da descrição VHDL do Bloco de Dados do processador Cleópatra
3. -- original que implementa os barramentos de entrada da ALU (BUSA
4. -- e BUSB)
5.  busA <= mdr;
6.  busB <= pc when (mux_busB=”00”) else ac when (mux_busB=”01”) else rs;
```

O sinal mux_busB substitui o sinal read_reg. A codificação de valores usada na solução para este sinal é a seguinte: quando ele vale “00” lê-se para BUS_B o valor do PC; quando ele vale “01” lê-se o valor do AC para BUS_B; senão, lê-se o valor do RS para o BUS_B.