

EA 869 – Turma A – 1. Semestre 2008.
Prova 5 – 26/06/2008 – Prof. Léo Pini Magalhães
 (com consulta a 1 folha A4)

Nome: Ricardo Drogo Ryghetto Número: 06A144

Q1. (3,0) Considere a interface serial para entrada de dados ENTR⁺.

(a) (0,75) inicialmente defina todos os seus registradores de forma a que ela possa operar de forma síncrona ou assíncrona, com 1 ou 2 StopBits (sempre 1 StartBit), ASCII de 7 ou 8 bits, 4 baud-rates diferentes (A, B, C, D), funcionamento por interrupção (ou não).

(b) (1,25) descreva sucintamente como esta interface opera (seu relacionamento com a CPU e seu relacionamento com o dispositivo). Lembre-se que o usuário poderá desejar operar condicionalmente, incondicionalmente ou por interrupção.

(c) (1,0) programe a sua interface para realizar saída em modo condicional, baud-rate C, 1 Stop-bit, ASCII de 7 bits. Escreva o programa executando na CPU para controle da saída de dados, considerando que a CPU oferece uma arquitetura de E/S mapeada e é uma máquina de 2 endereços.

a) [RC]:

7	6	5	4	3	2	1	0
---	---	---	---	---	---	---	---

 Registrador de Controle

Descrição: 0 - Síncrona / Assíncrona

1 - $\frac{\text{Start bit} + \text{Assistive}}{\text{Stop bit} + 2 \text{ Stop bits}}$

2- ASCII de $\frac{1}{8}$ bits na transmissão

4 e 3- Band Rate: 00-A

01-B

10-2

11-D

5 - Funcionamento por interrupção

6-Mode Conditional

7- Modo Indicativo

Apenas 1 doles pode estar setado

[RE]:

7	6	5	4	3	2	1	0
---	---	---	---	---	---	---	---

 Registrador de Estado

Descrição: 0 - Indica se dispositivo pode receber dado
1, 2, 3, 4, 5, 6, 7 - Don't Care

[RD]

7	6	5	4	3	2	1	0
---	---	---	---	---	---	---	---

 Registrador de Datos

Interface é serial - recebe 1 bit de cada vez

b) Modo por interrupção: quando o dispositivo está pronto para receber o dado, ele comunica a interface, que por sua vez gera uma interrupção na CPU. Esta interrupção aciona uma rotina de serviço que disponibiliza o dado para a interface. A interface inicia a transmissão para o dispositivo, bit a bit (serial, simplex, ASCII e apenas words). A transmissão começa pelo start bit, envia n caracteres em de acordo com o baud rate escolhido (4 possibilidades) e encerra com os stop bits, que podem ser 1 ou 2.

- Modo condicional: a CPU fica em loop aguardando a interface setar o bit de "OK" no RE. Quando esta condição é satisfeita, o dado é disponibilizado para a interface, que os envia da maneira descrita acima.

continua no verso 6

Q2. (1,0) Sejam os seguintes trechos em linguagem Assembly :

	Programa 1	Programa 2	
1700H	ORG 1700H	ORG 602H	
	BX: ADR Y	B: DW 40	602H, 603H
	:	X: EQU B	604H
	ORG 1000H	BX: ADR X	
1000H	Y: EQU B	:	
	B: DW 40		
	:		

Forneça a Tabela de Símbolos para cada um dos programas acima.

Programa 1:		Programa 2:		OBS: DW usa 2 bytes para armazenar o valor!
Símbolo	Valor	Símbolo	Valor	
BX	1700H ✓	B	602H ✓	
Y	1000H ✓	X	602H ✓	
B	1000H ✓	BX	604H ✓	

Q3. (3,0) Considere o seguinte programa em linguagem assembly:

Hexa ORG 4144 ; xxxx são seus 4 últimos dígitos de matrícula
 4144 início: MOVE cont, D1 ; D1 ← (cont): move (cont) para o registrador D1
 4147 ADDI #6, D1 ; D1 ← (D1) + 6: adiciona o valor 6 à (D1)
 4149 MOVE D1, saída ; saída ← (D1) : move (D1) para a variável saída
 414C STOP
 414D cont: DW 3 ; define a var. "cont" com valor 3
 414F saída: DS 2 ; define espaço para a variável saída

TIM

TPI

Mnemônico	C.O. (hexadec.)	Compr.(bytes)	Mnemônico	Compr.(bytes)
ADDI , D1	E0	2	ORG	--
MOVE , D1	A3	3	DW	2
MOVE D1,	B3	3	DS I	2 * I
STOP	D1	1		

Obtenha o programa em linguagem de máquina do programa acima. Dê a sua resposta usando a base hexadecimal (final do passo 1 e final do passo 2). Justifique a necessidade ou não dos passos 1 e 2 para processar este programa.

Passo 1: Processar Pseudo-Instruções e gerar Tabela de Símbolos

Tabela de Símbolos:

Símbolo	Valor
início	4144H
cont	414DH
saída	414FH

OBS: acompanhamento de posição colocado ao lado do código, baseado em TIM e TPI

• Modo incondicional: o envio de dados através da interface fica a critério do programador, que pode fazê-lo em qualquer ponto no código do programa de acordo com sua necessidade e natureza dos dispositivo.

Do ponto de vista do dispositivo, este sabe que a transmissão começou ou terminou através dos start bits. A interface deve ser programada (RC) de acordo com as especificações deste (Baud rate, Acct de 7 ou 8 bits, 1 ou 2 stop bits...), e portanto devem ser conhecidos pelo programador da CPU, caso contrário os dados não serão compreendidos.

No modo condicional, a interface limpa o bit 0 do RE quando o dispositivo confirma o recebimento. Isto também pode ser implementado no funcionamento por interrupção.

transformação serial - paralela \odot

c) : MOVE #^{7 6 5 4 3 2 1 0}01010000, RC; programa a interface configurando o RC (obs: modo condicional e naturalmente síncrono)
LOOP: MOVE RE, Acc; insere estado da interface no acumulador
JZ 0, LOOP; se o bit 0 não estiver setado, volta para o loop e continue aguardando. Dispositivo não está pronto!
MOVE DADO, RD; se dispositivo estiver pronto, sai do LOOP e insere dado na interface, que o envia serialmente ao dispositivo seguindo as especificações programadas em RC.

Q4. (0,5) A separação das atividades de ligação daquelas de carregamento, leva à existência de um programa carregador e um programa ligador. Qual o objetivo desta separação?

X O objetivo desta separação é permitir que o programa possa ser realocado em pontos diferentes da memória a cada execução.

Q5. (2,5) Considere um montador de ligação direta como aquele abordado em classe. Este montador ao final do processo de montagem gerou os seguintes módulos objeto que foram passados ao carregador de ligação direta com endereço inicial de carga (IPLA) igual a \$03B2:

PRINC: MOVE.B B, D0	0 -ESD.'PRINC'. 'SD'.00.1C
CMPI.B #10, D0	1 -TXT.00.6.103900000000
BLT PROX	2 -TXT.06.4.0C00000A
RTS	3 -TXT.0A.2.6D02
PROX: ADDI.B #'0', D0	4 -TXT.0C.2.4E75
MOVE.B D0, VAR1	5 -TXT.0E.4.06000030
RTS	6 -TXT.12.6.13C00000001A
VAR1: DS.W 1	7 -TXT.18.2.4E75
END PRINC	8 -TXT.1A.2.0000
	9 -RLD.02.4.'B'
	10 -RLD.14.4.'PRINC'
	11 -END.00
	12 -ESD.'SEC'. 'SD'.0C
SEC: MOVE.W A, D0	13 -ESD.'B'. 'LD'.08.2
RTS	14 -TXT.00.6.30390000000A
B: DS.W 1	15 -TXT.06.2.4E75
A: DS.W 1	16 -TXT.08.2.0000
END SEC	17 -TXT.0A.2.0000
	18 -RLD.02.4.'SEC'
	19 -END

obs.: “,” é usado para separar campos

(a) (0,5) Mostre a tabela GEST gerada ao final da fase 1 do carregador. Justifique resumidamente cada valor colocado na tabela.

GEST:	Símbolo	Valor
	PRINC	\$03B2 → SIMB='PRINC', valor = PLA=IPLA, Scomp=1C
	SEC	\$03CE → SIMB='PRINC', valor = PLA=IPLA+1C=03CE, Scomp=
	B	\$03D6 → SIMB='B', valor = PLA+POS=03CE+08=

03B2+
001C
03CE
0008
03DA

✓

Passo 2: Gerar o código-objeto, consultando-se TIM, TPI e TS

Posição (Hexa)	Código-Objeto
4144	A3 → MOVE cont, DI
4145	41
4146	4D } Endereço cont
4147	E0 → ADDI #6, DI
4148	06 → valor 6
4149	B3 → MOVE DI, sorda
414A	41
414B	4F } Endereço sorda
414C	D1 → STOP
414D	00
414E	03 } DW 3
414F	00
4150	00 } Reservados (DW 2)
	00
	00
	00

Este programa requer a montagem em 2 passos, pois as variáveis "cont" e "sorda" são utilizadas antes de serem definidas. Isto significa que, para gerar o código de máquina corretamente, o montador precisa das referências contidas na Tabela de Símbolos, que, portanto, deve ser gerada num passo (leitura do código-fonte) anterior.

(b) (2,0) Mostra na tabela abaixo o código obtido após o passo 2 do carregador.

- Preencha na 1. coluna o endereço inicial (por exemplo 010- para 1000)
- Indique quais posições foram ajustadas pelo carregador (use 2 linhas no campo, a superior com o conteúdo inicial e a inferior com o conteúdo final)

endereço	0-1	2-3	4-5	6-7	8-9	A-B	C-D	E-F
03B-	X	1039	0000 03D6	0000	0C00	000A	6D02	4E75
03C-	0600	0030	13C0	0000 03B2	001A 03B2 03CC	4E75	0000	3039
03D-	0000 03CE	000A 03CE 03D8	4E75	0000	0000			

Indique obrigatoriamente seu raciocínio aqui.

- POS = 00 SIZE = 6 CODE = 1039 00000000
PLA + POS = 03B2
- POS = 06 SIZE = 4 CODE = 0C00000A
PLA + POS = 03B8
- POS = 0A SIZE = 2 CODE = 6D02
PLA + POS = 03BC
- POS = 0C SIZE = 2 CODE = 4E75
PLA + POS = 03BE
- POS = 0E SIZE = 4 CODE = 06000030
PLA + POS = 03C0
- POS = 12 SIZE = 6 CODE = 13C00000001A
PLA + POS = 03C4
- POS = 18 SIZE = 2 CODE = 4E75
PLA + POS = 03CA
- POS = 1A SIZE = 2 CODE = 0000
PLA + POS = 03CC
- POS = 02 SIMB = 'B' POS + PLA = 03B4 = ENDR
VALOR = 03D6 (GEST)
(ENDR) = (03B4) = ((03B4) + 03D6 = 0000 + 03D6 = 03D6
- POS = 14 SIMB = 'PRINC' ENDR = POS + PLA = 03C6
VALOR = 03B2 (GEST)
(ENDR) = (03C6) = ((03C6) + 03B2 = 0000 + 03B2 = 03B2
- POS = 00 EXE = 03B2 + 00 = 03B2 PLA = 03B2 + SCOMP = 03B2 + 1C = 03CE
- SCOMP = 0C
- definição LD.
- POS = 00 SIZE = 6 CODE = 3039 0000000A
PLA + POS = 03CE + 00 = 03CE
- POS = 06 SIZE = 2 CODE = 4E75
PLA + POS = 03CE + 06 = 0201