LICENCIATURA EM ENGENHARIA INFORMÁTICA E DE COMPUTADORES

Introdução à Arquitetura de Computadores

Guia de Laboratório

2016/ 2017

INSTITUTO SUPERIOR TÉCNICO

Plano das aulas

1ª Aula: Introdução ao Unix
Laboratório sobre ambiente Unix: comandos da shell e editor de texto.

2ª Aula: Resolução de Exercícios
Resolução de Exercícios sobre Operações Aritméticas

3ª Aula: 1º Trabalho
Implementação de circuito combinatório usando um simulador lógico.

4ª Aula: 2º Trabalho
Análise e simulação de uma Unidade Lógica e Aritmética (ALU). Compreensão e análise das várias unidades que constituem a ALU.

5ª Aula: 3º Trabalho
Instruções aritméticas, lógicas e de salto. Concepção, teste e correção de pequenos programas.

6ª Aula: 4º Trabalho
Noção de rotinas em Assembly e de métodos de passagem de parâmetros.

7ª Aula: 5º Trabalho
Interação com dispositivos de entrada e saída. Análise do sistema de interrupções do processador P3.

8ª Aula: 1ª parte do Projeto
"Checkpoint" do projecto. Avaliar a capacidade dos alunos de conceber, desenvolver e testar um programa em linguagem Assembly, utilizando os conceitos adquiridos nas aulas anteriores.

9ª Aula: 6º Trabalho – parte I
Introdução à micro-programação. Análise de uma instrução Assembly em termos de micro-programação. Modificação do funcionamento de uma instrução Assembly.

10ª Aula: 6º Trabalho – parte II
Conclusão da aula anterior. Apoio ao projeto.

11ª Aula: Visualização do Projeto

12ª Aula: Discussões

Notas Gerais

Entrega da ficha de trabalho

Os alunos deverão ser portadores da ficha do trabalho em papel, devidamente preparada, para que a possam entregar no fim da aula de laboratório. Essa ficha terá de vir preenchida com as respostas às perguntas que compõem o trabalho de casa, e será completada durante a aula, em particular com as respostas às dúvidas que poderão ser colocadas ao docente de laboratório.

Entrega de esquemas e programas

Quando houver lugar à entrega de esquemas lógicos ou programas em Assembly, estes devem vir também já impressos, podendo ser corrigidos ou modificados manualmente no laboratório.

Acesso às máquinas dos laboratórios

Os trabalhos de laboratório serão feitos em Linux. Algumas máquinas dos laboratórios têm dual-boot (Windows e Linux). Se a máquina não estiver em Linux deve fazer reboot e escolher o sistema operativo Linux.
O login é feito com o IST ID (credenciais do fénix), e a diretoria pessoal de cada utilizador consiste na sua área pessoal do AFS disponibilizada pela DSI. É portanto obrigatório ter o serviço AFS ativado, o que pode ser feito aqui:
https://ciist.ist.utl.pt/servicos/self_service/
Mais informações sobre os laboratórios podem-se obter aqui:
https://www.rnl.ist.utl.pt/

Editor de texto

Para editar os programas em Assembly pode usar qualquer editor de texto (gedit, emacs, vi, etc). Recomendamos usar um editor de texto adequado à edição de código fonte. O bloco de notas do Windows não é adequado para esse fim.

Ficheiros de apoio aos laboratórios

Os ficheiros de apoio aos laboratórios encontram-se na página da disciplina.

1º Trabalho

Objectivos

Introdução ao ambiente de laboratório. Introdução aos circuitos combinatórios. Familiarização com o ambiente de simulação LOGISIM.

Tópicos

1. **Circuitos combinatórios simples**
   - Concepção, implementação e verificação
   - Familiarização com o simulador lógico LOGISIM
2. **Somador em cascata**

Trabalho de casa

Em cada semana, uma parte do trabalho de laboratório consiste num trabalho de casa, que servirá de preparação do restante trabalho a ser efetuado durante aula. O trabalho de casa é realizado em grupo, e avaliado no início da aula. Nesta avaliação, para além de ter em conta a resolução escrita, o docente fará perguntas **individuais a cada elemento do grupo** sobre esta mesma resolução.

No primeiro trabalho de laboratório, os pontos **1.1, 1.2, 1.3, 1.4, 2.1, 2.2, e 2.3** do enunciado constituem o trabalho de casa. O ponto 2.2 será realizado com recurso ao simulador lógico LOGISIM, o qual deve ser carregado e instalado no computador que é usado para o trabalho de casa. Este simulador pode ser obtido em http://www.cburch.com/logisim/. Na página da cadeira pode também encontrar um manual do LOGISIM.

No início da aula, os alunos têm de trazer uma impressão em papel da Ficha 1 parcialmente preenchida com as questões do trabalho de casa. No final da aula, será entregue a Ficha 1 completamente preenchida.

Enunciado

1. Introdução à manipulação algébrica

Considere as seguintes funções lógicas correspondentes a um somador completo de 1 bit. Os termos Ai e Bi representam os bits i das parcelas que se pretendem somar, Ci-1 e Ci representam respetivamente o carry-in e o carry-out ao nível do bit i, Si representa o bit i da soma.

$$S_i = \overline{A_i}\,\overline{B_i}C_{i-1} + \overline{A_i}B_i\overline{C_{i-1}} + A_i\overline{B_i}\,\overline{C_{i-1}} + A_iB_iC_{i-1}$$
$$C_i = \overline{A_i}B_iC_{i-1} + A_i\overline{B_i}C_{i-1} + A_iB_i\overline{C_{i-1}} + A_iB_iC_{i-1}$$

1.1. Construa a tabela de verdade para as duas funções S_i e C_i.

1.2. Simplifique as funções S_i e C_i algebricamente com recurso a manipulação algébrica.

1.3. Negue as expressões para Si e Ci apresentadas e represente o resultado na forma conjuntiva utilizando as leis de Morgan.

1.4. A partir da tabela de verdade da alínea 1.1, apresente o resultado das alíneas anteriores na forma canónica normal conjuntiva.

2. Introdução à edição e simulação de esquemas lógicos no LOGISIM

Considere as expressões simplificadas de S_i e C_i obtidas na alínea 1.2.

2.1. Construa um esquema lógico que permita implementar as funções S_i e C_i.

2.2. Simule o circuito lógico e verifique o funcionamento do circuito para todas as combinações de entrada

2.3. Carregue o ficheiro aula1.cir no LOGISIM. O circuito implementado corresponde a um somador de 4 bits construído com base em somadores completos de 1 bit ligados em cascata. Considere a representação de números em complemento para 2. Simule as operações da tabela completando os espaços em branco.

2.4 Discuta os resultados obtidos com o docente de laboratório, e escreva um comentário aos resultados, que pode incorporar observações dessa discussão

Ficha 1 - Respostas às questões do 1º Trabalho

Grupo: _____	Turno: _____
Nº _____	Nome: _____
Nº _____	Nome: _____
Nº _____	Nome: _____

1. Introdução à manipulação algébrica

1.1. Tabelas de Verdade

A_i	B_i	C_{i-1}	C_i
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A_i	B_i	C_{i-1}	S_i
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

1.2. Simplificação algébrica

$S_i = \overline{A_i}\,\overline{B_i}C_{i-1} + \overline{A_i}B_i\overline{C_{i-1}} + A_i\overline{B_i}\,\overline{C_{i-1}} + A_iB_iC_{i-1}$

$C_i = \overline{A_i}B_iC_{i-1} + A_i\overline{B_i}C_{i-1} + A_iB_i\overline{C_{i-1}} + A_iB_iC_{i-1}$

1.3. Leis de Morgan

1.4. Forma canónica normal disjuntiva

2. Introdução à edição e simulação de esquemas lógicos no LOGISIM

2.1. Esquema lógico

2.2. Verificação somador completo de 1 bit

A_i	B_i	C_{i-1}	C_i
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A_i	B_i	C_{i-1}	S_i
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

2.3. Simulação somador completo de 4 bits

A	B	Cout	S
4	1		
2	-3		
-1	-3		
5	4		

A	B	Cout	S
0100	0001		

2.4. Comente os resultados obtidos:

2º Trabalho

Objectivos

Estudo da ALU: unidades aritmética, lógica e de deslocamento. Familiarização com o ambiente de simulação LOGISIM.

Tópicos

1. **Unidade Lógica e Aritmética (ALU)**
   - Análise das unidades Lógica, de Deslocamento e Aritmética
   - Simulação usando o simulador lógico LOGISIM

Trabalho de casa

Uma parte do trabalho de laboratório consiste num trabalho de casa, que servirá de preparação do restante trabalho a ser efetuado durante aula. O trabalho de casa é realizado em grupo, e avaliado no início da aula. Nesta avaliação, para além de ter em conta a resolução escrita, o docente fará perguntas individuais a cada elemento do grupo sobre esta mesma resolução. Neste 2º trabalho de laboratório, os pontos **1.1, 1.2, 1.3, 1.4 e 2.1** do enunciado constituem o trabalho de casa. No início da aula, os alunos têm de trazer uma impressão em papel da Ficha 2 parcialmente preenchida com as questões do trabalho de casa. No final da aula, será entregue a Ficha 2 completamente preenchida.

Enunciado

1. Análise da ALU

O primeiro exercício deste trabalho consiste na análise e alteração da uma ALU para realizar operações de 4 bits de números em complemento para 2.

1.1. Carregue o ficheiro aula2.cir no simulador LOGISIM

1.2. Analise o circuito fornecido e identifique as Unidades Aritmética, Lógica e de Deslocamento

a) Na Unidade Lógica indique quais as operações realizadas para cada combinação dos valores de FS = F2 F1 F0, por observação do respetivo circuito.

b) Na Unidade de Deslocamento indique quais as operações realizadas (novamente para cada valor de FS = F2 F1 F0) por observação do respetivo circuito.

c) Na Unidade Aritmética realize os testes que considerar necessários para identificar as operações implementadas. Considere a tabela fornecida e indique para cada combinação de F1 F0 qual a operação realizada.

d) Complete a tabela que caracteriza as operações implementadas por esta ALU em função de FS = F2 F1 F0

1.3. Considere a tabela fornecida e determine, com recurso ao simulador, os resultados das operações indicadas.

1.4. Explique o funcionamento da Unidade Aritmética, isto é, de que forma é que os circuitos lógicos da Unidade Aritmética permitem implementar as operações atrás determinadas. A sua resposta deverá ser descrita de forma sucinta na ficha de laboratório.

2. Modificação da ALU

Altere a ALU apresentada de modo a executar a função "decremento de A" (A - 1) quando FS = 000, mantendo toda a restante funcionalidade.

2.1. Implemente estas alterações com recurso ao LOGISIM.

2.2. Explique ao docente da aula de laboratório as alterações efetuadas. Mostre o circuito em funcionamento para algumas combinações de inputs. Discuta e descreva também duas possíveis implementações para a função incremento (A + 1). Após a explicação das implementações da função incremento (A+1), descreva-as por escrito de forma sucinta na ficha de laboratório.

Ficha 2 - Respostas às questões do 2º Trabalho

Grupo: _____	Turno: _____
Nº _____	Nome: _____
Nº _____	Nome: _____
Nº _____	Nome: _____

1. Análise da ALU

1.2 Analise o circuito fornecido.

a) Unidade de Deslocamento

FS	Operação da UD
000	
001	
010	
011	
100	
101	
110	
111	

b) Unidade Lógica

FS	Operação da UL
000	
001	
010	
011	
100	
101	
110	
111	

c) Unidade Aritmética.

F1 F0	Operação da UA
00	
01	
10	
11	

d) Unidade Aritmética e Lógica

FS	Operação da ALU
000	
001	
010	
011	
100	
101	
110	
111	

1.3. Verificação do funcionamento da ALU

FS	A	B	Resultado
000	0101	0001	
001	0101	0001	
010	0101	0001	
011	0101	0001	
100	0101	0001	
101	0101	0001	
110	0101	0001	
111	0101	0001	

1.4. Explicação sucinta do funcionamento da ALU

_____

_____

_____

_____

_____

_____

_____

2. Modificação da ALU

2.1. Desenhe as alterações realizadas no circuito original. (Pode anexar uma impressão pelo simulador do circuito modificado devidamente anotada.)

2.2. Descrição de duas alternativas para implementar a função A+1

_____

_____

_____

_____

_____

3º Trabalho

Objectivos

Instruções aritméticas, lógicas e de salto. Concepção, teste e correção de pequenos programas.

Tópicos

Instruções aritméticas, lógicas e de salto

1. **Programação Assembly**
   - Análise de um programa
   - Construção de pequenos programas a partir de uma especificação simples
2. **Familiarização com os métodos de teste e de correção de programas. Utilização dos comandos: Pontos de Paragem e Continua**

Trabalho de casa

Uma parte do trabalho de laboratório consiste num trabalho de casa, que servirá de preparação do restante trabalho a ser efetuado durante aula. O trabalho de casa é realizado em grupo, e avaliado no início da aula. Nesta avaliação, para além de ter em conta a resolução escrita, o docente fará perguntas individuais a cada elemento do grupo sobre esta mesma resolução. Neste 3º trabalho de laboratório, os pontos 1.1, 1.2, 1.3, 2.1 e 2.2 do enunciado constituem o trabalho de casa. Os pontos deste trabalho serão realizados com recurso ao simulador do processador P3 (p3sim), o qual deve ser carregado e instalado no computador que é usado para o trabalho de casa. Este simulador pode ser obtido na página da cadeira. Na página da cadeira pode também encontrar um tutorial do simulador p3sim. No início da aula, os alunos têm de trazer uma impressão em papel da Ficha 3 parcialmente preenchida com as questões do trabalho de casa. No final da aula, será entregue a Ficha 3 completamente preenchida.

Enunciado

1. Instruções de salto. Análise de um programa

Copie para uma pasta temporária o ficheiro "aula3.as" que está disponível na página da cadeira e se encontra listado no Anexo I.

1.1. Por inspeção do referido ficheiro, identifique:

a) As instruções de salto incondicional. Para cada instrução identificada indique em que condições é que o salto ocorre e para onde.

b) As instruções de salto condicional. Para cada instrução identificada indique em que condições é que o salto ocorre e para onde.

c) A função realizada pelo programa desde o início até à etiqueta `Meio`.

d) A função realizada pelo programa desde a etiqueta `Meio` até `Fim`.

1.2. Com recurso ao simulador p3sim execute o programa até à etiqueta `Meio`. Para tal, proceda como a seguir se indica:

- Comece por localizar, na janela de código, a linha correspondente a essa etiqueta (`Meio`), recorrendo à informação existente no ficheiro de referências (aula3.lis).
- Introduza um ponto de paragem nessa linha. Para tal, selecione a linha, selecione o comando **Pontos de Paragem**, existente no menu **Depuração**, e selecione os botões **Adiciona** e **Fecha**.
- Seguidamente, execute o programa, selecionando o botão **Corre**.

Confirme a função identificada em 1.1.c), por análise do conteúdo da janela de memória.

1.3. Finalize a execução do programa, selecionado o botão Continua até ser atingida a etiqueta Fim.

Confirme a função identificada em 1.1.d), por análise do conteúdo da janela de memória.

2. Instruções aritméticas e lógicas. Concepção de programas

Utilizando a linguagem Assembly do simulador P3, conceba um programa para realizar cada uma das funções que a seguir se descrevem.

2.1. Soma de dois números positivos de 32 bits.

Os números a somar deverão estar em memória. Como cada posição de memória só contém 16 bits, cada número ocupará duas posições de memória. Assuma que o primeiro número começa na posição de memória com endereço `Num1` e o segundo em `Num2`, devendo o resultado ser armazenado em duas posições de memória a partir do endereço `Soma`.

Preencha os valores iniciais das posições de memória dos operandos através do comando **Escreve Memória**.

NOTA IMPORTANTE

A palavra mais significativa de cada número ocupa a posição de memória de endereço mais elevado.

EXEMPLO

Se quiser somar os números `12018091h` com `4f018061h` teremos em memória:

```
Num1        STR   8091h, 1201h
Num2        STR   8061h, 4f01h
Soma        TAB   2
```

2.2. Descompactação de uma sequência de bits memória. Cada bit de uma sequência de bits será colocado numa posição de memória separada.

NOTAS IMPORTANTES

Existe uma posição de memória com o número de palavras ocupadas pela sequência de bits. Tem que ser reservado espaço em memória onde fiquem colocados os dados da descompactação.

EXEMPLO

Se em Assembly tiver o seguinte código:

```
DadosIniciais    STR   F0F1h
NumDados         WORD  1
DadosFinais      TAB   16
```

Em memória, com início na posição `DadosIniciais`, fica:

```
F0F1
```

Após descompactação, a memória a partir da posição DadosFinais, fica:

```
0001 0001 0001 0001 0000 0000 0000 0000 0001 0001 0001 0001
0000 0000 0000 0001
```

2.3. Modifique o programa anterior para contar o número de bits a um nos dados iniciais e colocar o resultado numa posição de memória denominada NumeroBitsUm

Ficha 3 - Respostas às questões do 3º Trabalho

Grupo: _____	Turno: _____
Nº _____	Nome: _____
Nº _____	Nome: _____
Nº _____	Nome: _____

1. Instruções de salto. Análise de um programa

1.1 Analise o programa "aula3.as" e identifique:

a) Instruções de salto incondicional (indique quando salta e para onde)

b) Instruções de salto condicional (indique quando salta e para onde)

c) Função do programa até à etiqueta Meio.

d) Função do programa da etiqueta Meio até Fim.

2. Instruções aritméticas e lógicas. Concepção de programas

2.1 Soma de dois números positivos de 32 bits.
Listagem do programa desenvolvido (ou anexe folha):

2.2 Descompactação de palavras de 16 bits.
Listagem do programa desenvolvido (ou anexe folha):

2.3 Descompactação de palavras de 16 bits.
Listagem do programa desenvolvido (ou anexe folha):

4º Trabalho

Objectivos

Noção de rotinas em Assembly e de métodos de passagem de parâmetros.

Tópicos

1. **Rotinas e métodos de passagem de parâmetros. Instruções de manipulação da pilha.**
   - Instruções Assembly associadas à utilização de rotinas
   - Instruções de manipulação da pilha.
   - Métodos de passagem de parâmetros.
2. **Concepção de programas com rotinas que permitam aplicar os conceitos anteriores.**

Trabalho de casa

Neste 4º trabalho de laboratório, os pontos 1.1, 1.2, 1.3 e 1.5 do enunciado constituem o trabalho de casa. O horário de laboratório destina-se à verificação das alíneas 1.4. e 1.6, e à resolução de eventuais problemas. No final da aula deverá ser entregue a Ficha 4 devidamente preenchida.

Enunciado

1. **Análise e alteração de um programa que usa rotinas.**

1.1 Analise o programa "aula4.as" e identifique as rotinas existentes, as suas funcionalidades e os métodos de passagem de parâmetros utilizados.

1.2 Introduza um ponto de paragem no início da rotina EscString, utilizando o procedimento descrito na alínea 1.2 do trabalho anterior. Execute o programa até ao ponto de paragem e, a partir daí, execute a rotina EscString até ao fim instrução a instrução, incluindo a rotina EscCar e respectivo retorno (RETN). Analise a evolução da pilha e preencha a respetiva tabela.

1.3 Analise o programa e identifique o método de passagem de parâmetros da rotina CountBits.

1.4 Abra a Janela de Texto do simulador e execute o programa. Verifique que o número de bits a '1' da palavra processada está correto (número introduzido premindo uma tecla entre 0 e 9).

1.5 Modifique o programa de modo a que a passagem de parâmetros da rotina CountBits seja feita pela pilha.

1.6 Abra a Janela de Texto do simulador e execute o programa. Verifique que o funcionamento é idêntico ao obtido na alínea 1.4.

Ficha 4 - Respostas às questões do 4º Trabalho

Grupo: _____	Turno: _____
Nº _____	Nome: _____
Nº _____	Nome: _____
Nº _____	Nome: _____

1. Análise e alteração de um programa que usa rotinas.

1.2 Preencha a seguinte tabela com o conteúdo da pilha à medida que vai executando o código instrução a instrução. Colocar apenas os valores que fazem parte da pilha!

	Antes de PUSH R1	Após PUSH R1	Após PUSH R2	Após PUSH R1	Após CALL	Após PUSH R1	Após POP R1	Após RETN 1	Após POP R2	Após POP R1
FDFF										
FDFE										
FDFD										
FDFC										
FDFB										
FDFA										
FDF9										
FDF8										
SP =										

Indique que valor foi inserido na pilha após a instrução CALL.

_____

_____

_____

Indique qual a função da instrução MOV R1, M[SP+3] na rotina EscCar.

_____

_____

_____

1.3 Indique o método de passagem de parâmetros (entrada e saída) da rotina CountBits.

_____

_____

_____

1.5 Modifique o programa de modo a que a passagem de parâmetros da rotina CountBits seja feita pela pilha.

Apresente o código relativo à rotina e à sua chamada (ou anexe folha).

5º Trabalho

Objectivos

Interação com os restantes dispositivos de entrada e saída. Aprofundamento do sistema de interrupções do processador P3. Familiarização com a placa que emula o funcionamento do P3.

Tópicos

1. Interrupções

- Interrupções na arquitetura P3
- Instruções Assembly associadas às interrupções

Entradas e Saídas

- Utilização da placa que emula funcionamento do P3

Trabalho de casa

Uma parte do trabalho de laboratório consiste num trabalho de casa, que servirá de preparação do restante trabalho a ser efetuado durante aula. O trabalho de casa é realizado em grupo, e avaliado no início da aula. Nesta avaliação, para além de ter em conta a resolução escrita, o docente fará perguntas individuais a cada elemento do grupo sobre esta mesma resolução. Neste 5º trabalho de laboratório, os pontos 1.1, 1.2, 1.3, 1.4, 1.5, 2.1 e 2.2 do enunciado constituem o trabalho de casa. Na aula, serão efetuadas as perguntas 1.6 e 2.3. No início da aula, os alunos têm de trazer uma impressão em papel da Ficha 5 parcialmente preenchida com as questões do trabalho de casa. No final da aula, será entregue a Ficha 5 completamente preenchida.

Enunciado

1. Interrupções

1.1 Copie para a diretoria do grupo o ficheiro “aula5.as” que se encontra listado no Anexo I.

1.2 Analise o programa e identifique:

a) O programa principal e a rotina de serviço à interrupção;

b) A zona do programa que preenche a tabela de vectores de interrupção;

c) A função da rotina de serviço à interrupção.

1.3 Execute o programa e confirme a sua funcionalidade.

1.4 Justifique a existência das instruções ENI e DSI na rotina EscCont.

1.5 Altere na tabela de interrupções a posição correspondente ao botão 0 (I0) para o valor 10h (na posição 0 da tabela de interrupções é colocado o valor 10h). Que alterações tem que efetuar no assembly para o programa ter o mesmo comportamento quando se executa.

1.6 Execute o programa na placa que emula o P3. Compare a execução na placa com a execução no simulador.

2. Entradas/saídas e interrupções

2.1 Implemente um relógio digital usando os displays de 7 segmentos, em que os dois dígitos da esquerda representam os minutos e os dois dígitos da direita representam os segundos. A contagem total dos segundos deve aparecer também nos leds, em binário.

2.2 Altere o programa anterior para que ao clicar no botão 1 (I1) o relógio pare. O relógio retoma a contagem quando se clicar de novo no botão 1.

2.3 Execute o programa na placa que emula o P3.

> NOTA
>
> O programa a desenvolver será visualizado na placa dedicada que emula o funcionamento do P3. Para carregar o programa para a placa deve executar o seguinte comando: P3_Loader <ficheiro.exe>. Se executar P3_Loader sem argumentos entra no modo interativo.

Ficha 5 - Respostas às questões do 5º Trabalho

Grupo: _____	Turno: _____
Nº _____	Nome: _____
Nº _____	Nome: _____
Nº _____	Nome: _____

1. Interrupções

1.2 Analise o programa "aula5.as".

a) Identifique o programa principal e a rotina de serviço à interrupção (indique as respectivas etiquetas).

_____

_____

b) Transcreva a parte do programa que preenche a tabela de vectores de interrupção.

_____

_____

c) Indique a função da rotina de serviço à interrupção.

_____

_____

1.4 Justifique a existência das instruções ENI e DSI na rotina EscCont.

_____

_____

_____

_____

_____

1.5 Altere no vector de interrupção a posição correspondente ao botão 0 (I0) para o valor 10h. Que alterações tem que efetuar no assembly para o programa ter o mesmo comportamento quando se executa.

_____

_____

_____

_____

1.6 Que diferenças existem entre as execuções? Justifique o porquê das diferenças.

_____

_____

_____

_____

2. Entradas/saídas e interrupções - Relógio Digital (com todas as funcionalidades)

Apresente o código desenvolvido ou anexe folha.

1ª Parte do Projecto

Objectivos

Avaliar a capacidade dos alunos de conceber, desenvolver e testar um programa em linguagem Assembly, utilizando os conceitos adquiridos nas aulas anteriores.

6º Trabalho

Objectivos

Introdução à microprogramação. Análise do microprograma de uma instrução Assembly. Modificação do funcionamento de uma instrução Assembly.

Tópicos

1. Introdução à microprogramação (parte I do trabalho de laboratório)

- A microprogramação na arquitectura P3
- Registos associados à microprogramação

2. Análise de uma instrução Assembly (parte I do trabalho de laboratório)

- Formatos e tipos de instruções Assembly
- Microinstruções
- Fluxograma de execução de uma instrução
- Microprograma de uma instrução Assembly

3. Modificação de uma instrução Assembly (parte II do trabalho de laboratório)

Nota: O trabalho deverá ser preparado fora do horário de laboratório, destinando-se as horas de laboratório à resolução de eventuais problemas, e à demonstração do trabalho realizado. No final da aula deverá ser entregue a Ficha 6 devidamente preenchida.

Trabalho de casa

O **trabalho de casa consiste na realização dos pontos 1.1 a 1.6 (parte I) e 2.1 a 2.4 (parte II)** , e correspondente preenchimento da ficha de avaliação.
No início da aula de laboratório, cada grupo deverá mostrar ao docente a ficha parcialmente preenchida com as respostas às questões. Durante a aula o docente fará ainda **perguntas individuais a cada elemento** do grupo. No final da aula, cada grupo deverá entregar a ficha completamente preenchida.

Enunciado

1. Introdução à microprogramação. Análise de uma instrução Assembly

1.1 Copie para a diretoria de grupo o ficheiro "aula6.as" que se encontra listado no Anexo I. Proceda à geração do ficheiro executável.

1.2 Selecione no simulador a janela que contém a informação referente à microprogramação. Para tal, selecione a opção **Ver Controlo** existente no menu **Ver**.

1.3 Inicie a execução do programa começando por executar apenas a primeira instrução (`MOV R1,1000h`). Para isso, premir uma vez o botão **Instrução**.

1.4 Prossiga a execução do programa ciclo a ciclo de relógio, premindo uma vez o botão **Clock**. Tendo por base o conteúdo do registo RI identifique:

a) O tipo de instrução Assembly.

b) O conteúdo de cada um dos seus campos.

c) O modo de endereçamento utilizado, baseado nos valores obtidos na alínea anterior.

1.5 Continue com a execução do programa, ciclo a ciclo de relógio, e preencha a Tabela 1 da Ficha 6. Para cada microinstrução, identifique a que zona do Fluxograma 1

(representado na mesma ficha) está associada (IF,EX,F1,F2,WB ou IH), descreva as ações realizadas usando Linguagem de Transferência de Registos (RTL) e indique a sua codificação hexadecimal. Preencha apenas os valores dos registos quando eles mudam de valor; quando mantêm o valor deixe os campos em branco.

1.6 Recorrendo ao conjunto de informação existente no "Manual do Simulador do Processador P3", identifique o microprograma da instrução em causa (`INC R1`). Preencha as Tabelas 2, 3 e 4 da Ficha 6 de acordo com a informação obtida:

a) Transferência de registos

b) Microprogramação

c) Conteúdo da ROM de controlo

1.7 Repita os pontos 1.1 a 1.5 para explicar ao docente a vossa resposta a estas questões.

2. Modificação de uma instrução Assembly

2.1. Pretende-se modificar o microprograma analisado na pergunta 1.6, de forma a multiplicar operando por 5 (em vez de somar o valor 1 ao operando). Para realizar esta operação sugere-se a utilização do seguinte procedimento: (i) multiplique o operando no registo RD por 4 realizando dois deslocamentos e guardando o resultado num registo auxiliar; (ii) adicione ao resultado anterior o valor do operando em RD; (iii) coloque o resultado final no registo R13 (RD).2.1 Indique em que fase(s) do Fluxograma 1 presente na Ficha 6 é que se efetuam as alterações. Justifique.

2.2 Analise a arquitetura do P3 representada na Ficha 6. Indique os sinais que devem ser ativados, e com que valor, de forma a implementar as microinstruções que realizam:

a) R9←INC RD, SBR←CAR+1, CAR←F1.

b) RD←RD+R9, flags ZCNO, CAR←WB.

2.3 Modifique o microprograma analisado em 1.6 de modo a modificar a instrução INC para passar a somar o valor 2 ao operando. Na Tabela 3 da Ficha 6 indique apenas as microinstruções novas/modificadas. Para cada uma dessas microinstruções indique o respetivo endereço e as respetivas ações usando linguagem de transferência de registos.

2.4 Introduza no simulador as alterações que efetuou no microcódigo.

Para tal, gere um ficheiro "control.roms" com a alteração a efetuar no microcódigo. Cada linha do ficheiro deve conter a seguinte informação:

<endereço da microinstrução que vai ser substituída> <nova microinstrução>

Notas

A. Os endereços e microinstruções devem estar em hexadecimal mas não devem incluir a letra ´h´. Exemplo de uma linha:

```
024  000A009F
```

B. Introduza apenas as linhas que pretende modificar.

- Efetue o carregamento do referido ficheiro no simulador, selecionando a opção Carrega ROM de Controlo que existe no menu Ficheiro.

- Execute o programa (para as duas primeiras instruções use o botão Instrução e, a partir daí, execute ciclo a ciclo de relógio - botão **Clock**). Recorra à informação da janela de microprogramação para efetuar o teste do microcódigo alterado.

2.5 Verifique, e demonstre ao docente que a instrução `INC M[Valor]` também funciona de acordo com o esperado após a alteração da microprogramação da instrução `INC`.

Ficha 6 - Respostas às questões do 6º Trabalho

Grupo: _____	Turno: _____
Nº _____	Nome: _____
Nº _____	Nome: _____
Nº _____	Nome: _____

1. Introdução à microprogramação. Análise de uma instrução Assembly

1.4 (Nota: Consulte o anexo A do "Manual do Simulador do Processador P3") Execute um ciclo da instrução INC R1. Analise o registo RI e indique:

a) Se o tipo de instrução Assembly é de um ou dois operandos

_____

b) O conteúdo dos campos da instrução

_____

_____

c) O modo de endereçamento

_____

1.5 Execute a instrução INC R1 ciclo a ciclo de relógio e preencha a tabela seguinte. Para cada microinstrução, identifique a que zona do Fluxograma 1 (representado na mesma ficha) está associada (IF,EX,F1,F2,WB ou IH), descreva as ações realizadas usando Linguagem de Transferência de Registos (RTL) e indique a sua codificação hexadecimal. Preencha apenas os valores dos registos quando eles mudam de valor; quando mantêm o valor deixe os campos em branco.

Fluxo-grama	Ling. de Transf. de Registos (RTL)	Micro-instrução	CAR	SBR	RI	R11 (SD)	R12 (EA)	R13 (RD)	R14 (SP)	R15 (PC)
IF	IR ← M[PC]	8060001F	000							

Tabela 1 - Execução de uma instrução ciclo a ciclo (valores em heaxdecimal)

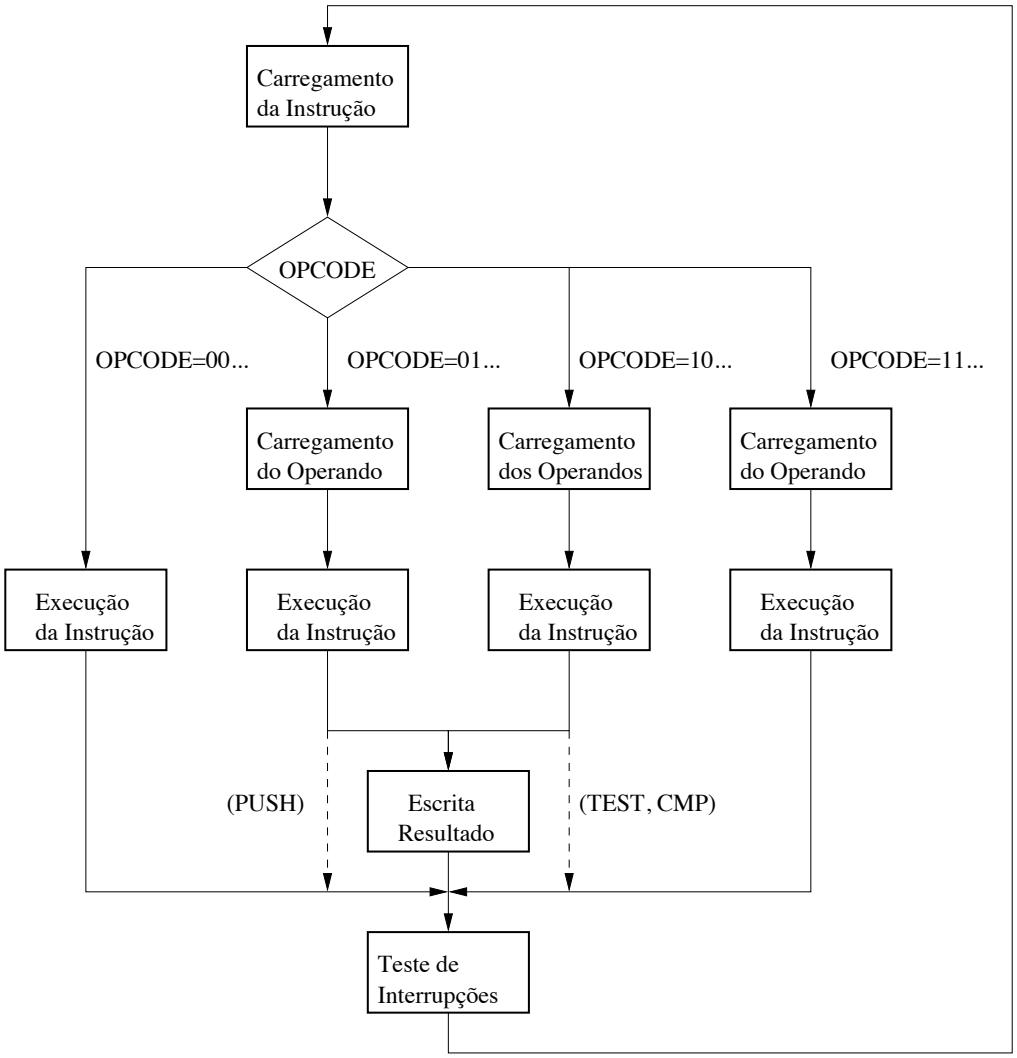


Figura 1 - Ciclo de execução de uma instrução

1.6 Microprograma da instrução INC R1.

a) Indique as ações de cada microinstrução em linguagem de transferência de registos na Tabela 2.

b) Apresente a codificação de cada uma das microinstruções na Tabela 3. Na codificação de cada microinstrução assinale apenas os uns (1) e os zeros (0). Deixe em branco as posiçõess das indiferenças.

c) Preencha a Tabela 4 indicando a codificação de cada microinstrução e seu endereço na ROM de controlo.

NOTA: Faça corresponder cada linha das Tabelas 3 e 4 às linhas da Tabela 2.

2. Modificação de uma instrução Assembly

2.1 Indique a fase do ciclo de instrução em que são efectuadas as alterações ao microprograma (ver Figura 1).

_____

_____

	Endereço Simbólico	Transferência de Registos
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

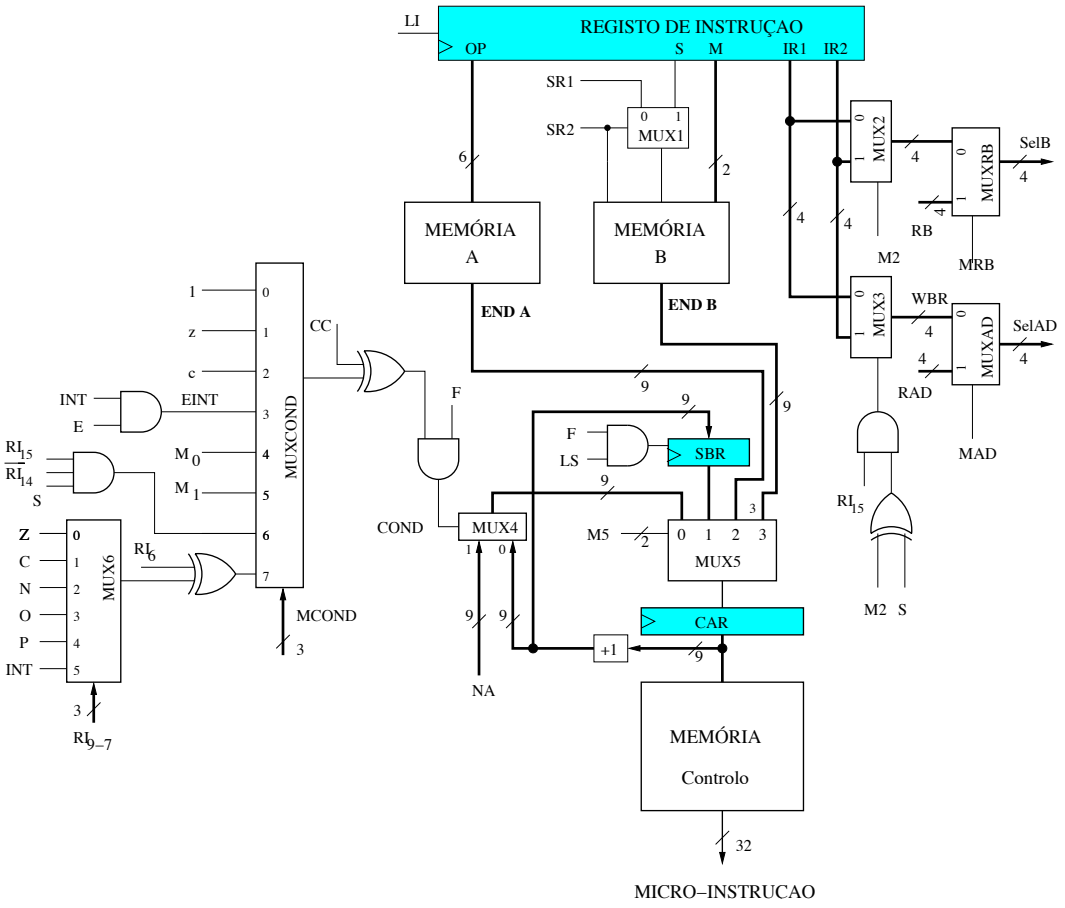
Tabela 2 - Transferência de Registos

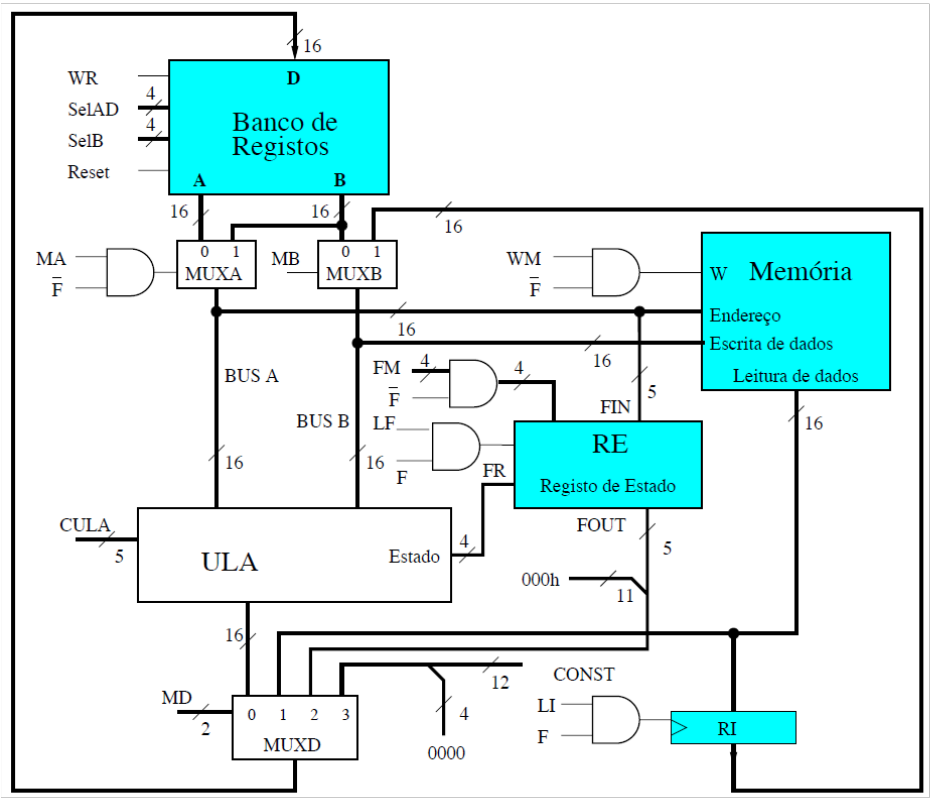
	F	M5		SR1	SR2	IAK	FM				CALU					MA	MB	M2	MRB	RB			WM	WR	MD		MAD	RAD			
						LS	MCOND			CC	LI	LF	CONST/NA																			
1																															
2																															
3																															
4																															
5																															
6																															
7																															
8																															
9																															
10																															

Tabela 3 – Microprogramação

	Endereço ROM	Conteúdo ROM
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Tabela 4 - Conteúdo da ROM de Controlo


LI
REGISTO DE INSTRUÇAO
OP
S
M
IR1
IR2
SR1
SR2
MUX1
MEMÓRIA A
MEMÓRIA B
END A
END B
MUX2
MUXRB
SelB
RB
M2
MRB
MUX3
WBR
MUXAD
SelAD
RAD
MAD
MUXCOND
CC
INT
EINT
E
S
F
LS
SBR
COND
MUX4
M5
MUX5
CAR
+1
NA
MCOND
MUX6
Z
C
N
O
P
INT
MEMÓRIA
Controlo
32
MICRO–INSTRUCAO


WR
SelAD
SelB
Reset
Banco de Registos
D
A
B
MA
MB
MUXA
MUXB
WM
Memória
Endereço
Escrita de dados
Leitura de dados
BUS A
BUS B
FM
LF
FIN
RE
Registo de Estado
FR
FOUT
CULA
ULA
Estado
000h
MD
MUXD
0000
CONST
LI
RI

2.2 Analise a arquitetura do P3 representada na Ficha 6. Indique os sinais que devem ser ativados, e com que valor, de forma a implementar uma microinstrução que realize:

R9←INC RD, SBR←CAR+1, CAR←F1.

RD←RD+R9, flags ZCNO, CAR←WB.

2.3 Indique as alterações a efetuar ao microprograma analisado em 1.6. Indique o endereço das microinstruções novas/modificadas e as respectivas ações usando linguagem de transferência de registos (RTL – Register Transfer Language). Indique, também, a codificação (em hexadecimal) de cada microinstrução.

Endereço (hex)	**Microinstrução (RTL)**	**Microinstrução (codificação em hexadecimal)**

Tabela 5- Alterações ao microprograma

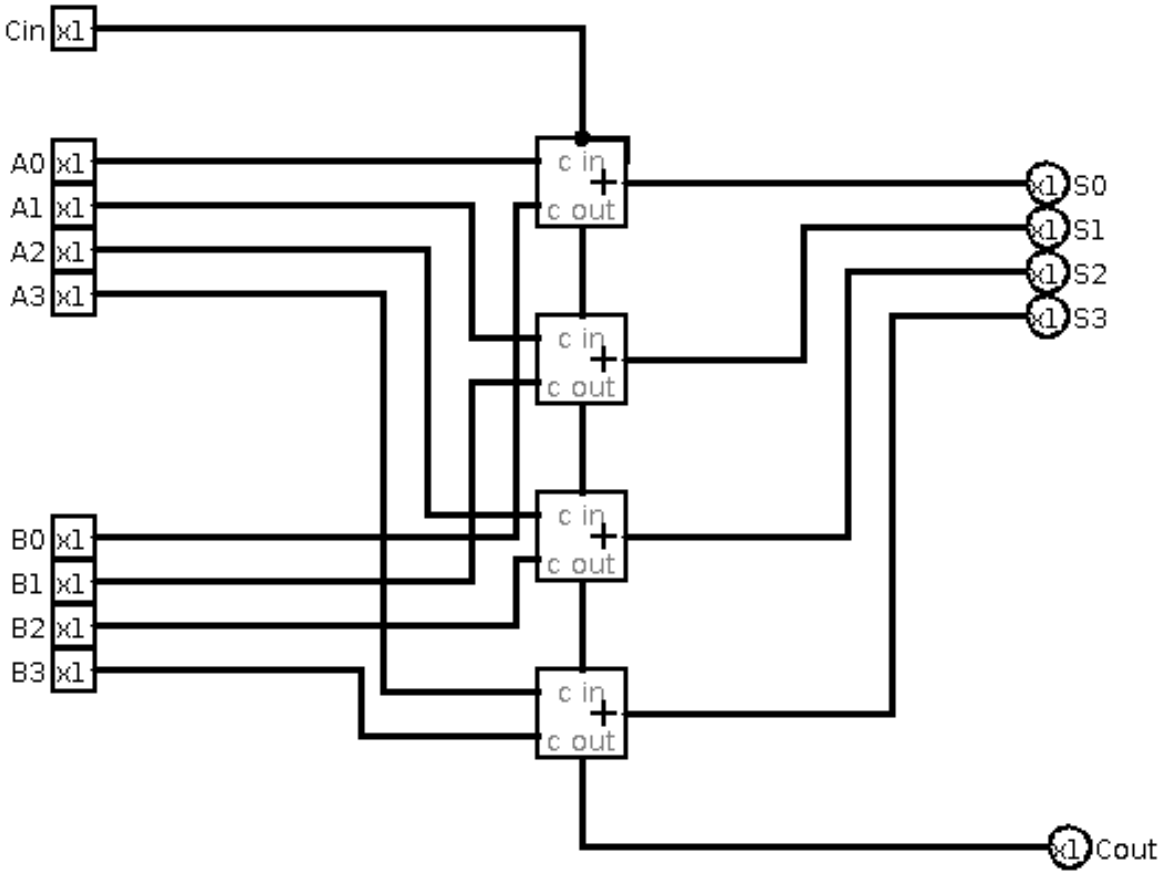
2ª Parte do Projecto

Objetivos

Avaliar a capacidade dos alunos de conceber e desenvolver um programa de uma nova instrução Assembly, utilizando os conceitos adquiridos nas aulas anteriores. Avaliar ainda a capacidade de concepção, desenvolvimento e teste de um programa complexo em linguagem Assembly, fazendo uso dessa nova instrução.

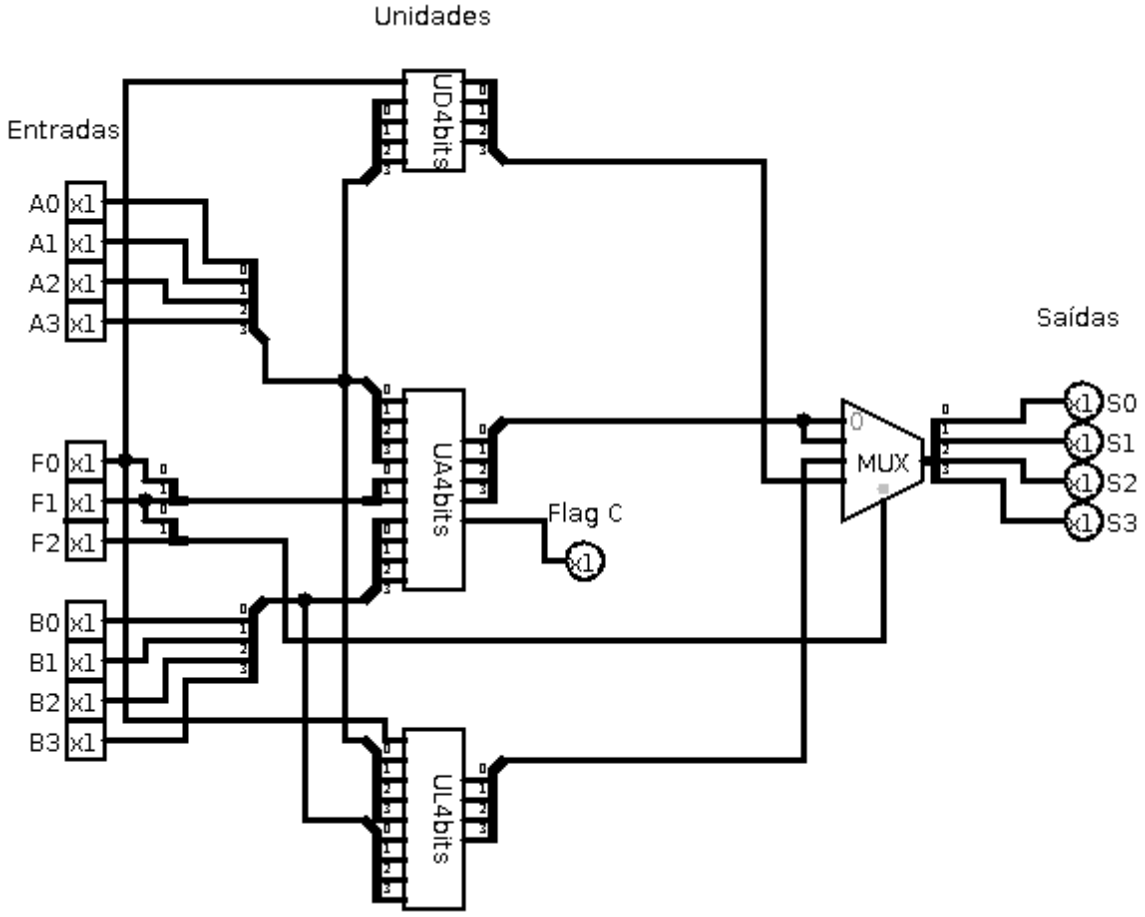
Anexo I – Circuitos e listagens dos programas

aula1.circ

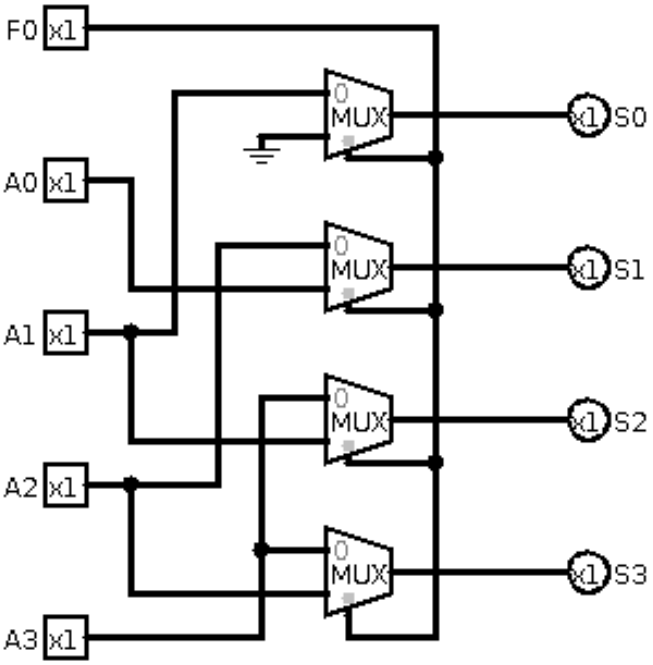


aula2.circ

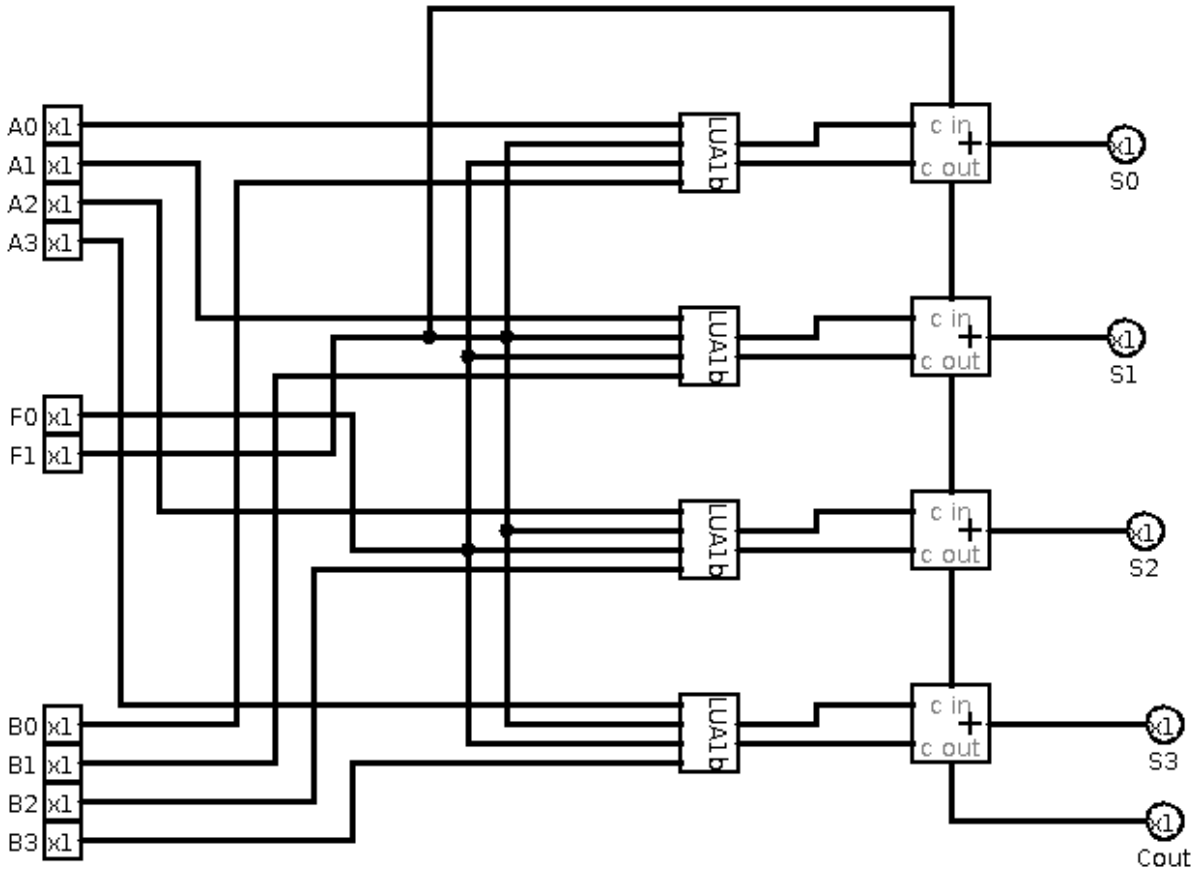
ULA



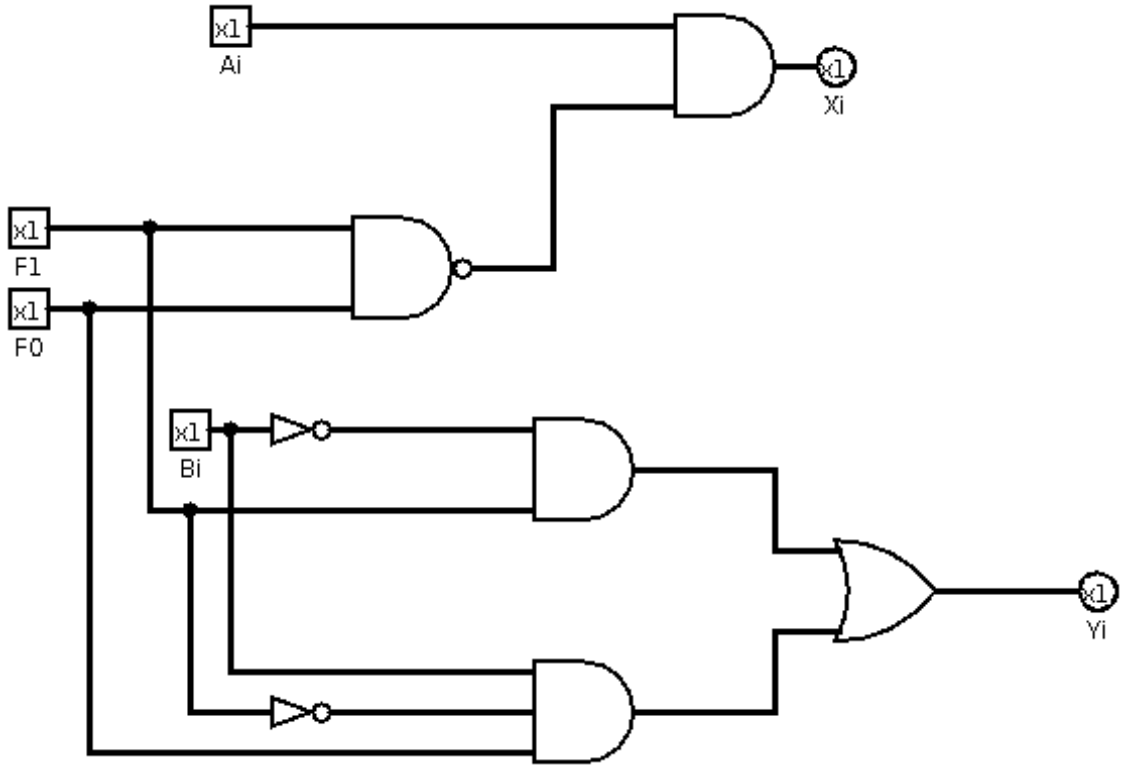
UD4bits



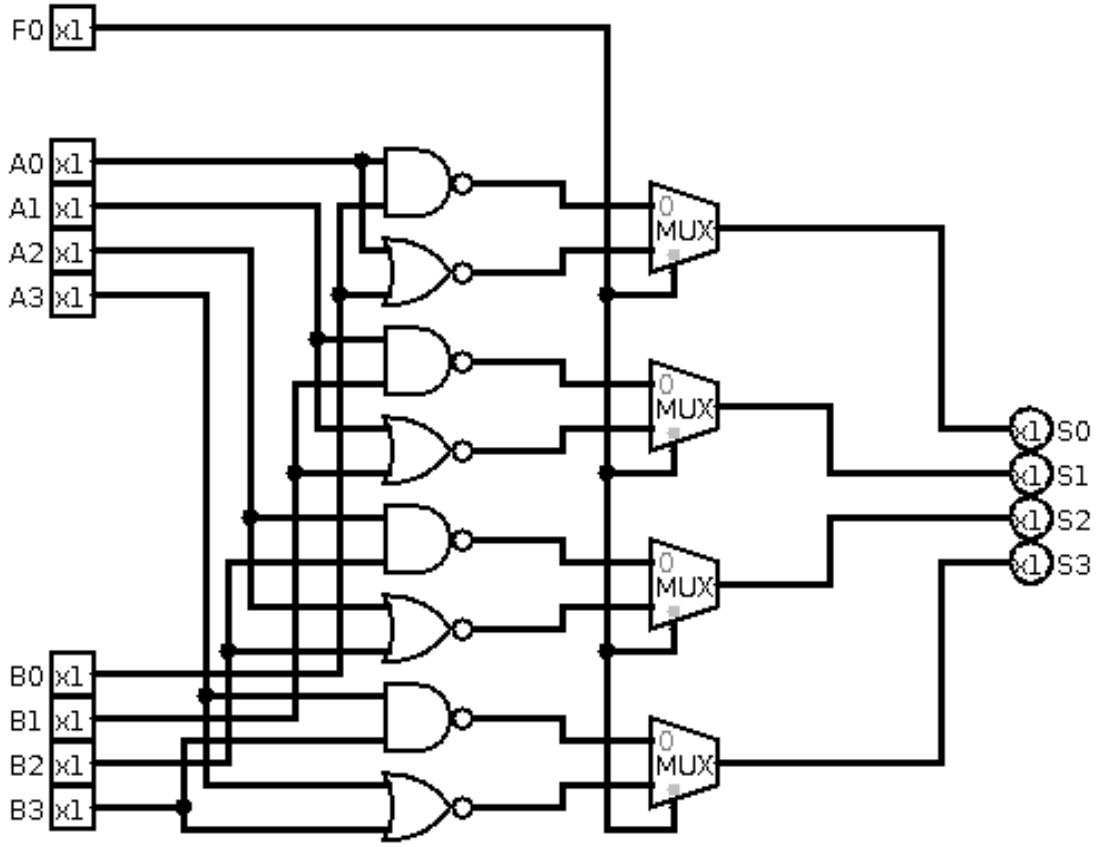
UA4bits



LUA1b



UL4bits



aula3.as

```
; Programa aula3.as
; NOTA: Este programa nao esta' comentado INTENCIONALMENTE !!

FIM_STR         EQU     '@'

                ORIG    8000h
VarStrOrigem    STR     'Arquitectura de Computadores @'
VarStrDestino   TAB     30

                ORIG    0000h
Inicio:         MOV     R1, VarStrDestino
                MOV     R2, VarStrOrigem
Ciclo:          MOV     R3, M[R2]
                CMP     R3, FIM_STR
                BR.Z    Meio
                MOV     M[R1], R0
                INC     R1
                INC     R2
                BR      Ciclo

Meio:           MOV     R3, M[R2]
                MOV     M[R1], R3
                DEC     R2
                DEC     R1
                CMP     R1, VarStrDestino
                BR.NN   Meio
Fim:            BR      Fim
```

aula4.as

```
; Programa aula4.as


; ZONA I: Definicao de constantes
;        Pseudo-instrucao : EQU
CR              EQU     0Ah
FIM_TEXTO       EQU     '@'
IO_READ         EQU     FFFFh
IO_WRITE        EQU     FFFEh
IO_STATUS       EQU     FFFDh
SP_INICIAL      EQU     FDFFh



; ZONA II: definicao de variaveis
;         Pseudo-instrucoes : WORD - palavra (16 bits)
;                             STR  - sequencia de caracteres.
;         Cada caracter ocupa 1 palavra
                ORIG    8000h
VarTexto1       STR     'Pressione uma tecla entre 0 e 9: ', FIM_TEXTO
VarTexto2       STR     'Numero de bits a 1 = ', FIM_TEXTO
VarErro1        STR     'ERRO: Tecla Invalida', FIM_TEXTO




; ZONA III: codigo
;        conjunto de instrucoes Assembly, ordenadas de forma a realizar
;        as funcoes pretendidas

                ORIG    0000h
                JMP     Inicio
```

```
; LeCar: Rotina que efectua a leitura de um caracter proveniente do teclado
;               Entradas: ---
;               Saidas: R1 - caracter lido
;               Efeitos: alteracao do registo R1

LeCar:          CMP     R0, M[IO_STATUS]
                BR.Z    LeCar
                MOV     R1, M[IO_READ]
                RET



; EscCar: Rotina que efectua a escrita de um caracter para o ecra.
;       O caracter pode ser visualizado na janela de texto.
;               Entradas: pilha - caracter a escrever
;               Saidas: ---
;                       Efeitos: alteracao do registo R1
;                       alteracao da posicao de memoria M[IO]

EscCar:         PUSH    R1
                MOV     R1, M[SP+3]
                MOV     M[IO_WRITE], R1
                POP     R1
                RETN    1



; MudaLinha: Rotina que efectua a escrita de um caracter mudanca de linha.
;               Entradas: ---
;               Saidas: ---
;               Efeitos: ---
```

```
MudaLinha:      PUSH    R1
                PUSH    CR
                CALL    EscCar
                POP     R1
                RET



; EscString: Rotina que efectua a escrita de uma cadeia de caracter,
; terminada pelo caracter FIM_TEXTO. Pode-se definir como terminador
; qualquer caracter ASCII.
;           Entradas: R2 - apontador para o inicio da cadeia de caracteres
;           Saidas: ---
;           Efeitos: ---

EscString:      PUSH    R1
                PUSH    R2
Ciclo:          MOV     R1, M[R2]
                CMP     R1, FIM_TEXTO
                BR.Z    FimEsc
                PUSH    R1
                CALL    EscCar
                INC     R2
                BR      Ciclo
FimEsc:         POP     R2
                POP     R1
                RET


; EscNum: Rotina que efectua a escrita de um algarismo, fazendo a conversao
;         necessaria para ASCII.
;               Entradas: R1 - numero a escrever
```

```
;               Saidas: ---
;               Efeitos: ---

EscNum:         PUSH    R1
                ADD     R1,'0'
                PUSH    R1
                CALL    EscCar
                POP     R1
                RET



; PrintBits: Rotina que efectua a escrita da representacao em binario no
;            ecra de uma palavra de 16 bits.
;               Entradas: R1 palavra a imprimir
;               Saidas: ---
;               Efeitos: ---
PrintBits:      PUSH    R1
                PUSH    R2
                PUSH    R3
                PUSH    ' '
                CALL    EscCar
                PUSH    '('
                CALL    EscCar
                MOV     R2, 15
procbit:        SHL     R1, 1
                MOV     R3, '0'
                ADDC    R3, R0
                PUSH    R3
                CALL    EscCar
                DEC     R2
                BR.NN   procbit
```

```
                PUSH    ')'
                CALL    EscCar
                POP     R3
                POP     R2
                POP     R1
                RET


; CountBits: Rotina de calculo do numero de bits a '1' numa palavra de 16
bits.
;               Entradas: R1 - palavra a processar
;               Saidas: R1 - resultado
;               Efeitos: altera R2


CountBits:      MOV     R2, R0
continua:       SHR     R1, 1
                ADDC    R2, R0
                CMP     R1, R0
                BR.NZ   continua
                MOV     R1, R2
                RET



; Programa Principal: programa que recebe um algarismo do teclado,
;       retornando o numero de bits a '1' da correspondente representacao
;       em binario. Caso receba um caracter invalido, retorna uma
;       mensagem de erro.


Inicio:         MOV     R7, SP_INICIAL
                MOV     SP, R7


ProcWord:       CALL    MudaLinha
```

```
                MOV     R2, VarTexto1
                CALL    EscString
                CALL    LeCar ; Devolve em R1 caracter lido (simbolo ASCII)
                SUB     R1,'0'          ; Conversao do simbolo ASCII

                CMP     R1,0            ; Deteccao de condicao de erro
                BR.N    Erro
                CMP     R1,9
                BR.P    Erro

                CALL    EscNum          ; Escrita do numero lido (R1)

                CALL    PrintBits     ; Escrita da representacao em binario
                                      ; do numero lido
                CALL    MudaLinha

                CALL    CountBits       ; Contagem do numero de bits a '1'

                MOV     R2, VarTexto2   ;Escrita do valor calculado no ecra
                CALL    EscString
                CALL    EscNum

                CALL    MudaLinha
                JMP     ProcWord

Erro:           CALL    MudaLinha
                MOV     R2, VarErro1
                CALL    EscString
                JMP     ProcWord

Fim:            BR      Fim
```

aula5.as

```
;     Programa aula5.as


SP_INICIAL      EQU     FDFFh
INT_MASK_ADDR   EQU     FFFAh
INT_MASK        EQU     0000000000000001b
IO_DISPLAY      EQU     FFF0h
DELAY_COUNT     EQU     0200h
NIBBLE_MASK     EQU     000fh
NUM_NIBBLES     EQU     4
BITS_PER_NIBBLE EQU     4

; Palavra de memoria que contem a variavel de contagem
                ORIG    8000h
Contador        WORD    0000h



; Tabela de interrupcoes
                ORIG    FE00h
INT0            WORD    ResetCont

; Codigo
                ORIG    0000h
                JMP     Inicio

; ResetCont: Rotina que faz o reset do contador
;       Entradas: ---
;       Saidas: ---
;       Efeitos: alteracao do conteudo da posicao de memoria M[Contador]
ResetCont:      MOV     M[Contador], R0
```

```
                RTI

; ContHex: Rotina que incrementa o contador
;       Entradas: M[Contador] - contador
;       Saidas: ---
;       Efeitos: alteracao do conteudo da posicao de memoria  M[Contador]
ContHex:        INC     M[Contador]
                RET

; EscCont: Rotina que efectua a escrita do contador
;       Entradas: ---
;       Saidas: ---
;       Efeitos: ---
EscCont:        PUSH    R1
                PUSH    R2
                PUSH    R3
                DSI
                MOV     R2, NUM_NIBBLES
                MOV     R3, IO_DISPLAY
Ciclo:          MOV     R1, M[Contador]
                AND     R1, NIBBLE_MASK
                MOV     M[R3], R1
                ROR     M[Contador], BITS_PER_NIBBLE
                INC     R3
                DEC     R2
                BR.NZ   Ciclo
                ENI
                POP     R3
                POP     R2
                POP     R1
```

```
                RET

; Delay: Rotina que provoca um atraso
;       Entradas: ---
;       Saidas: ----
;       Efeitos: ---
Delay:          PUSH    R1
                MOV     R1, DELAY_COUNT
Delay_L1:       DEC     R1
                BR.NZ   Delay_L1
                POP     R1
                RET


Inicio:         MOV     R7, SP_INICIAL
                MOV     SP, R7
                MOV     R7, INT_MASK
                MOV     M[INT_MASK_ADDR], R7
                ENI
CicloCont:      CALL    EscCont
                CALL    Delay
                CALL    ContHex
                BR      CicloCont
```

aula6.as

```
;     Programa aula6.as - estudo da microprogramacao

            ORIG    8000h
Valor       WORD    0fffh

            ORIG    0000h
Inicio:     MOV     R1, 1000h
            INC     R1
            INC     M[Valor]

Fim:        BR      Fim
```