

Laboratório de Introdução à Arquitetura de Computadores

IST - Taguspark

2020/2021

Descodificação de endereços

Guião 7

30 de novembro a 4 de dezembro 2020

(Semana 9)

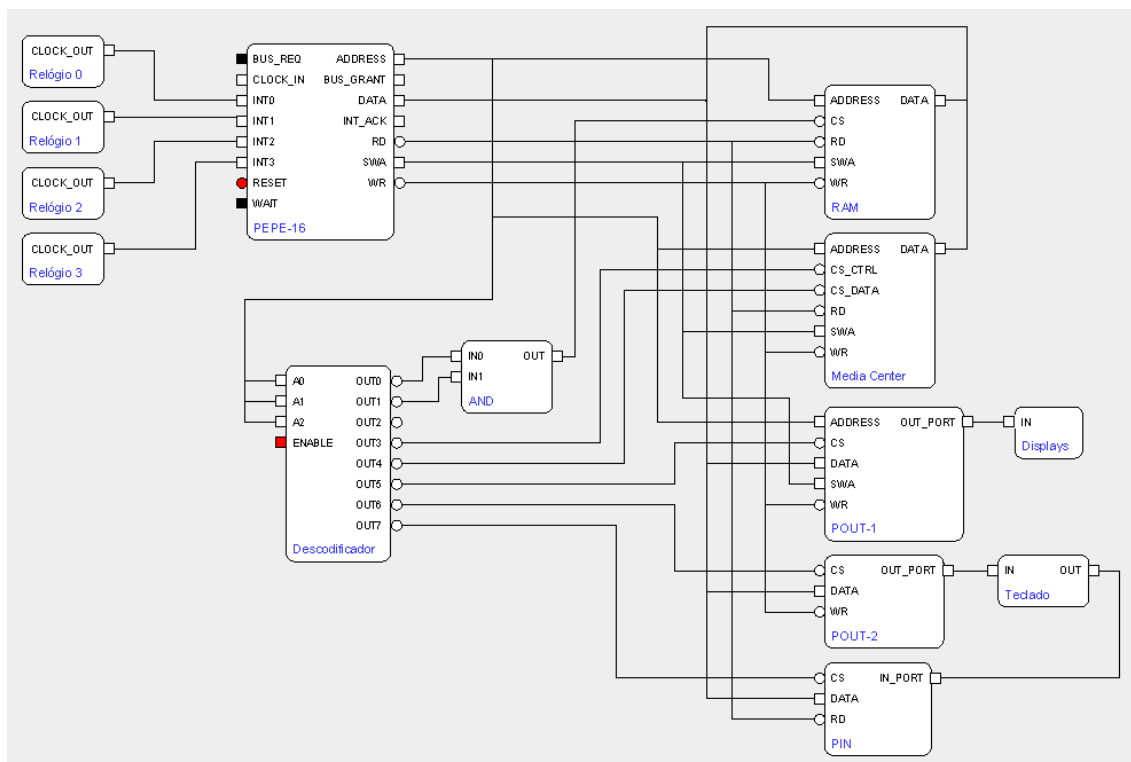
1 – Objetivos

Com este trabalho pretende-se exercitar e demonstrar a descodificação de endereços.

2 – O circuito de simulação

Nos Guiões de laboratório 5 e 6 utilizou-se o seguinte circuito, constituído por:

- Uma memória (RAM);
- Um ecrã (MediaCenter);
- Dois periféricos de saída (POUT-1, ligado a dois displays de 7 segmentos, e POUT-2, ligado às linhas do teclado);
- Um periférico de entrada (PIN, ligado às colunas do teclado).



Neste guião utiliza-se um circuito quase igual, contido no ficheiro **lab7.cir**. A diferença está no periférico POUT-1, que agora é de 16 bits (para ilustrar periféricos de 16 bits).

O mapa de endereços (em que os dispositivos podem ser acedidos pelo PEPE) é o seguinte (já vem do Guião de laboratório 4, com a diferença de que o POUT-1 tem agora dois endereços):

Dispositivo	Endereços
RAM	0000H a 3FFFH
MediaCenter (acesso aos comandos)	6000H a 6063H (ver guião 4)
MediaCenter (acesso à sua memória)	8000H a 8FFFH
POUT-1 (periférico de saída de 16 bits)	0A000H a 0A001H
POUT-2 (periférico de saída de 8 bits)	0C000H
PIN (periférico de entrada de 8 bits)	0E000H

3 – Execução do trabalho de laboratório

3.1 – Verificação do mapa de endereços

O Guião de laboratório 6 terminou com o programa contido no ficheiro **lab6-processos-estados.asm**, aqui também incluído para facilidade de referência.

Se quiser, releia a secção 3.5 do Guião de laboratório 6 e relembre a funcionalidade deste programa.

Vamos agora usá-lo na perspetiva da descodificação de endereços.

Carregue o programa *assembly* **lab6-processos-estados.asm**, carregando em **Load asm** (📁) ou *drag & drop*.

Abra o ecrã, os displays, o teclado e os painéis de todos os relógios.

Execute o programa, carregando no botão **Start** (▶) do PEPE.

Verifique que tudo funciona, e que os displays vão aumentando ou diminuindo o seu valor, consoante se carregue na tecla C ou D, respetivamente.

No entanto, há uma diferença: agora há 4 displays, e os que contam são os da esquerda!

Tal deve-se ao facto de o PEPE ser um processador Big-endian, e como tal o endereço 0A000H, usado para atualizar os displays, refere-se ao byte de maior peso dos registos. Se mudar o **MOVB** na rotina **altera_displays** para **MOV**, pode verificar que os 4 displays já contam nos displays do lado direito. Isto ilustra também a diferença de comportamento destas duas instruções.

Com o editor de texto, verifique no programa do ficheiro **lab6-processos-estados.asm**:

- Quais são as linhas de programa em que se definem os endereços em que os dispositivos estão acessíveis;
- Quais as constantes usadas para os definir;
- Que os endereços definidos são os do mapa acima.

Verifique agora no circuito que o mapa de endereços implementado é o indicado acima.

Com o simulador em modo “Design”, faça duplo clique no descodificador. Deverá abrir-se a janela ilustrada pela figura seguinte.

O bit A0 que se observa é o de menor peso dos três bits que especificam qual das saídas do decodificador está ativa em cada instante. Estes três bits ligam a bits do bus de endereços do PEPE (ver circuito na página 1). Note-se que as designações A0 a A2 do decodificador são internas e não ligam ao A0 a A2 do bus de endereços do PEPE.

A linha “Bit 0 of pin A0 connects to Connection at bit 13” indica a qual o bit do bus de endereços do PEPE (neste caso o bit 13) a que o A0 do decodificador liga. Faça clique no A1 e no A2 do Decodificador e verifique que o A0 do decodificador liga ao A13 do bus de endereços, o A1 ao bit A14 e o A2 ao bit A15:

Bit interno do decodificador	Bit do bus de endereços do PEPE
A0	A13
A1	A14
A2	A15

Decodificador configuration

Module

Name: Decodificador

Delay: 5 (simulation time units)

Number of address inputs: 3

Pins

A0
A1
A2
ENABLE
OUT0
OUT1
OUT2
OUT3
OUT4
OUT5

No. of bits of pin A0: 1 Fixed to: -

Bit 0 of pin A0 connects to Connection at bit: 13

Bit 0 of Connection connects to pin A0 at bit: 0

Close

A fatia de endereços em que cada saída do decodificador (que liga ao *Chip Select* de um dispositivo) se mantém ativa é de 2000H (ou 8 Ki) endereços, o que corresponde aos 13 bits (A0 a A12) que podem ligar a cada dispositivo ($2^{13} = 2000H$, ou 8 Ki), tal como indicado na tabela seguinte:

Saída do decodificador	Endereço inicial	Endereço final
OUT0	0000H	1FFFH
OUT1	2000H	3FFFH
OUT2	4000H	5FFFH
OUT3	6000H	7FFFH
OUT4	8000H	9FFFH
OUT5	A000H	BFFFH
OUT6	C000H	DFFFH
OUT7	E000H	FFFFH

A generalidade dos periféricos requer apenas uma fração de cada fatia de endereços. Por exemplo, o MediaCenter não precisa de todos na parte dos comandos, cada periférico de 16 bits usa dois endereços e cada periférico de 8 bits requer apenas um endereço.

Menos do que a fatia não tem problema, mas no caso da RAM quer-se uma capacidade que é o dobro (4000H) do tamanho da fatia, e é por isso que se usa um AND das primeiras duas saídas. Como estas são ativas a 0, basta uma delas estar ativa para ativar o *Chip Select* da RAM (também ativo a 0).

O mapa real de endereços é então (ver de novo o circuito na página 1):

Saída do decodificador	Dispositivo	Endereço inicial	Endereço final
OUT0	RAM	0000H	1FFFH
OUT1	RAM	2000H	3FFFH
OUT2	Não usado	4000H	5FFFH
OUT3	MediaCenter (comandos)	6000H	6063H (ver guião 4)
OUT4	MediaCenter (ecrã)	8000H	8FFFH
OUT5	POUT-1	A000H	A001H
OUT6	POUT-2	C000H	C000H
OUT7	PIN	E000H	E000H

O POUT-1 é um periférico de saída de 16 bits, pelo que gasta dois endereços. O POUT-2 e o PIN são periféricos de apenas 8 bits, pelo que só gastam um endereço.


Quando um dispositivo requer menos endereços do que a fatia do decodificador, os endereços em que o dispositivo está ativo repetem-se (espelhos).

Por exemplo, o periférico POUT-1 está também disponível de A002H a A003H, de A004H a A005H, etc. De 2 em 2, repete-se, ao longo de toda a gama de endereços da respetiva fatia.

Como os periféricos POUT-2 e PIN são periféricos de 8 bits, só usam os endereços pares, ou seja, só usam metade do barramento de dados. Por isso, os espelhos são sempre em endereços pares. Assim, o POUT-2 está acessível nos endereços C000H, C002H, C004H, etc., enquanto o PIN está acessível nos endereços E000H, E002H, E004H, etc.

3.2 – Alteração do mapa de endereços





Pretende-se agora alterar o mapa de endereços do circuito e verificar que basta alterar também os endereços no software para que tudo continue a funcionar. Assim:

- **Faça uma cópia** do ficheiro **lab7.cir** e carregue-o no simulador (por **Load** ) ou por *drag & drop*);
- **Faça as alterações necessárias** no circuito para que o novo mapa de endereços do circuito seja o seguinte:

Dispositivo	Endereços
RAM	0000H a 1FFFFH
MediaCenter (acesso aos comandos)	2000H a 2063H
MediaCenter (acesso à sua memória)	3000H a 3FFFFH
POUT-1 (porto de saída de 16 bits)	4000 a 4001H
POUT-2 (porto de saída de 8 bits)	5000H
PIN (porto de entrada de 8 bits)	6000H

SUGESTÃO – Veja qual o tamanho da fatia de endereços em que cada *Chip Select* (CS) deve estar ativo;

NOTAS:

- É preciso alterar diversas coisas, incluindo refazer ligações. Peça ajuda ao docente, se necessitar;
- A RAM é agora acessível apenas em 2000H endereços, em vez dos 4000H iniciais. Não é necessário reduzir a capacidade da RAM (as segundas 2000H células ficam desaproveitadas), mas pode experimentar reduzir a capacidade desta para metade (menos um bit de endereço) no seu painel de configuração;
- Guarde o novo circuito num ficheiro, para o caso precise de o usar novamente (carregando no botão  na toolbar da janela principal do simulador);
- Faça uma cópia do ficheiro lab6-processos-estados.asm;
- Nessa cópia, altere os endereços necessários para que o programa bata certo com este mapa de endereços. Deverá apenas alterar as constantes que definem os endereços dos periféricos;
- Carregue no PEPE o ficheiro com o programa alterado, com **Load asm**  ou *drag & drop* na janela de programa do PEP;
- Abra o ecrã, os dois displays, o teclado e os painéis de todos os relógios;
- Execute o programa, carregando no botão **Start** () do PEPE;
- Verifique que tudo funciona como dantes, e que os displays vão aumentando ou diminuindo o seu valor, consoante se carregue na tecla C ou D, respetivamente;
- Termine o programa, carregando no botão **Stop** () do PEPE.

Desta forma, é possível alterar o mapa de endereços do hardware e manter a funcionalidade do software, desde que se altere de forma correspondente no programa as constantes que definem os endereços.