Laboratório de Introdução à Arquitetura de Computadores

IST - Taguspark

2017/2018

Descodificação de endereços

Guião 8

20 a 24 de novembro 2017

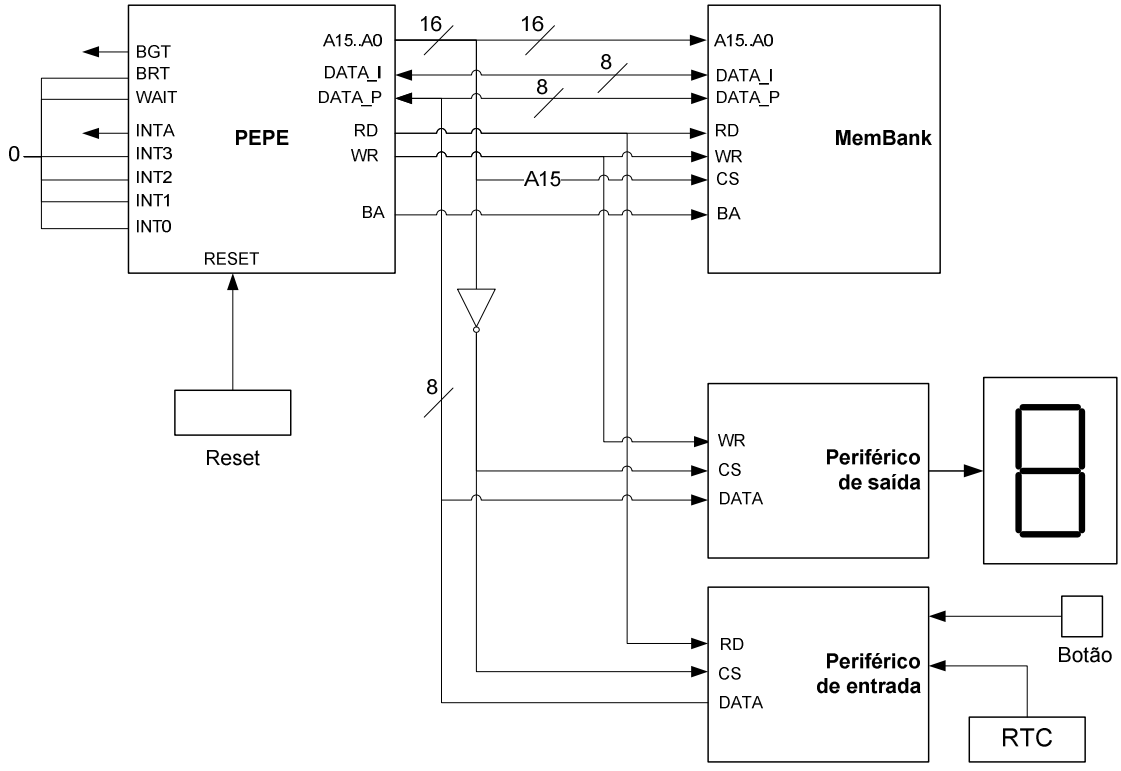
(Semana 10)

1 – Objetivos

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

2 – O problema

No guião de laboratório 6 utilizou-se o seguinte circuito, constituído por uma memória e por dois periféricos, um de saída e outro de entrada.



A descodificação de endereços é rudimentar e baseia-se simplesmente no bit A15 e na negação que a ele liga. Apesar de tão simples, este circuito funciona porque:

- A RAM é de 16 bits de largura e já suporta endereçamento de byte;
- O espaço de endereçamento é dividido ao meio, metade para RAM e metade para periféricos (se quisermos outro mapa de endereços já não pode ser assim);

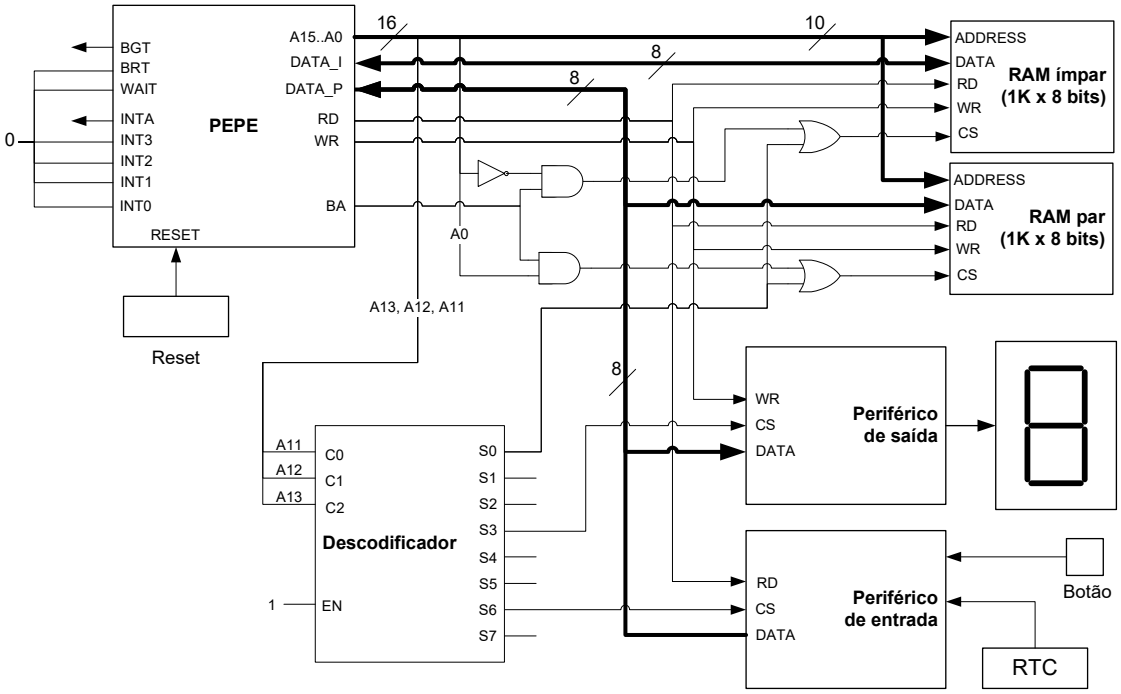
- A distinção entre periféricos de entrada e de saída é feita não por endereço mas por função (ou seja, estão os dois nos mesmo endereços, mas um só lê e o outro só escreve);
- Na segunda metade do espaço de endereçamento, os periféricos estão acessíveis repetidamente em todos os endereços pares.

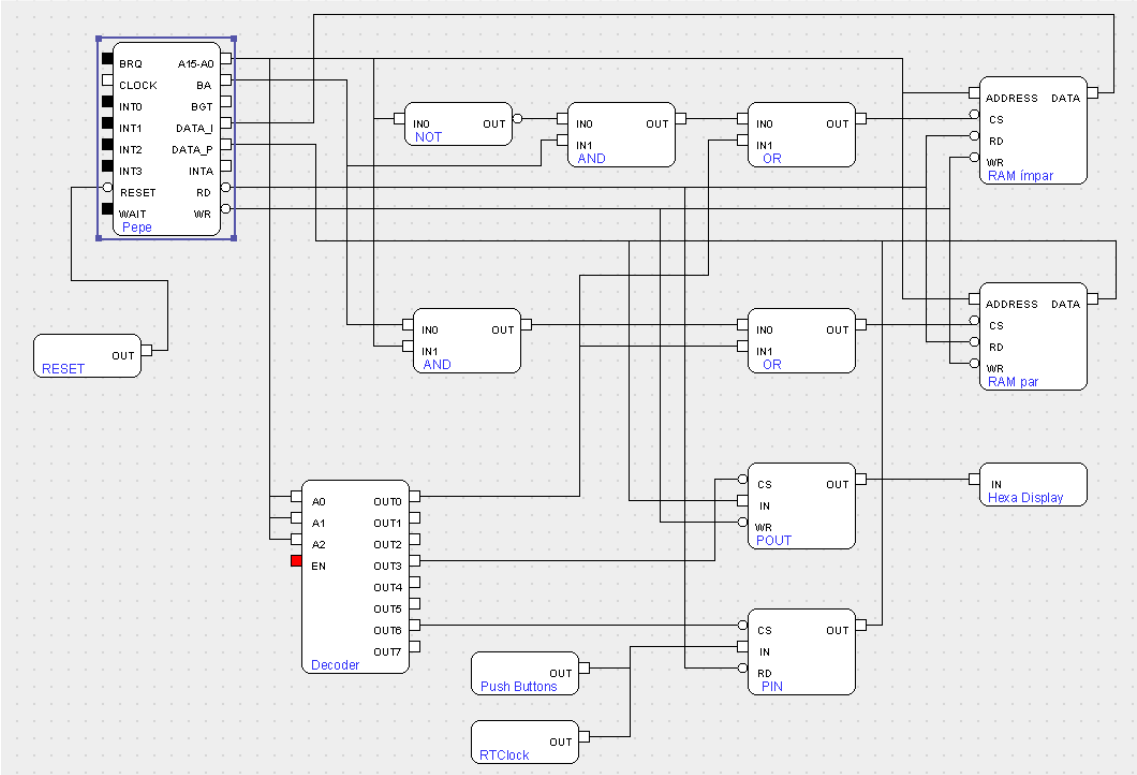
Note-se que os periféricos são apenas de 8 bits e ligam à metade par do barramento de dados (isto é, os 8 bits deste barramento que o processador usa para aceder aos bytes com endereço par).

Na vida real a situação não é tão simples e tem de se montar um circuito descodificador de endereços que produza de forma adequada os sinais CS (Chip Select) para as RAMs e periféricos.

3 – Circuito de descodificação de endereços

Pretende-se montar um circuito funcionalmente equivalente ao anterior, mas com um sistema de descodificação de endereços. Este circuito já é fornecido, no ficheiro **lab8a.cmod**.





Note os bits de endereço a que o descodificador de 3 bits (8 saídas) liga e o esquema de geração dos CS das RAM par e ímpar (o banco de memória é substituído, na prática, pelas duas metades da RAM e pelo circuito que gera os seus sinais CS).

Responda às seguintes perguntas sobre este circuito:

a) Que bits do barramento de endereços ligam a ADDRESS em cada uma das RAMs?

b) Faça o mapa de endereços, indicando a gama de endereços (em hexadecimal) em que cada uma das saídas do descodificador está ativa (a 0);

c) Em que gama de endereços há RAM? Quantos bytes de RAM há no sistema?

d) Em que gama de endereço está acessível cada um dos periféricos? Note que os periféricos são apenas de 8 bits e ligam apenas a metade do barramento de dados;

e) Se o processador aceder ao endereço 8000H, está a aceder a quê?

f) Se quisesse que o periférico de entrada estivesse disponível no endereço 4001H (ímpar), que alterações tinha que fazer ao circuito?

g) Qual seria o efeito no mapa de endereços se em vez de se ligar A13..A11 ao descodificador se ligasse A14..A12?

h) Descreva o funcionamento do sinal BA e a sua utilidade para o acesso à memória nas instruções MOVB.

4 – Software

Com o circuito **lab8a.cmod** fornecido, e que implementa a figura anterior, execute o programa **lab8.asm**, quase idêntico ao que já foi usado no guião de laboratório 6 (**lab6.asm**) e que também é fornecido aqui para facilidade de referência, mas em que os dispositivos têm de ser mapeados nos endereços decorrentes da figura (e que já determinou ao responder às perguntas anteriores).

Assim, o programa **lab8.asm** é idêntico ao **lab6.asm** com exceção de:

- Endereços dos periféricos (INPUT e OUTPUT), que no **lab6.asm** eram 8000H e agora têm de ser alterados de acordo com esta figura;
- Endereço do PLACE dos dados, que no **lab6.asm** era 1000H, mas que agora tem de ser alterado porque a RAM aqui é mais pequena.

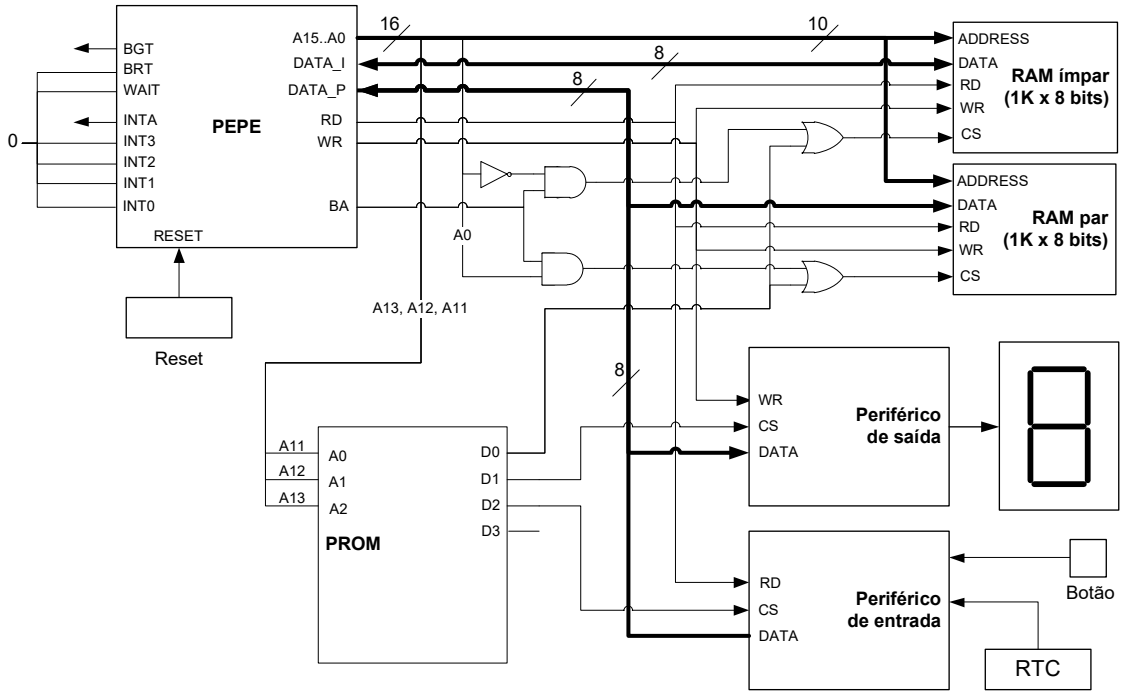
Antes de executar o programa, coloque os valores corretos no **lab8.asm**.

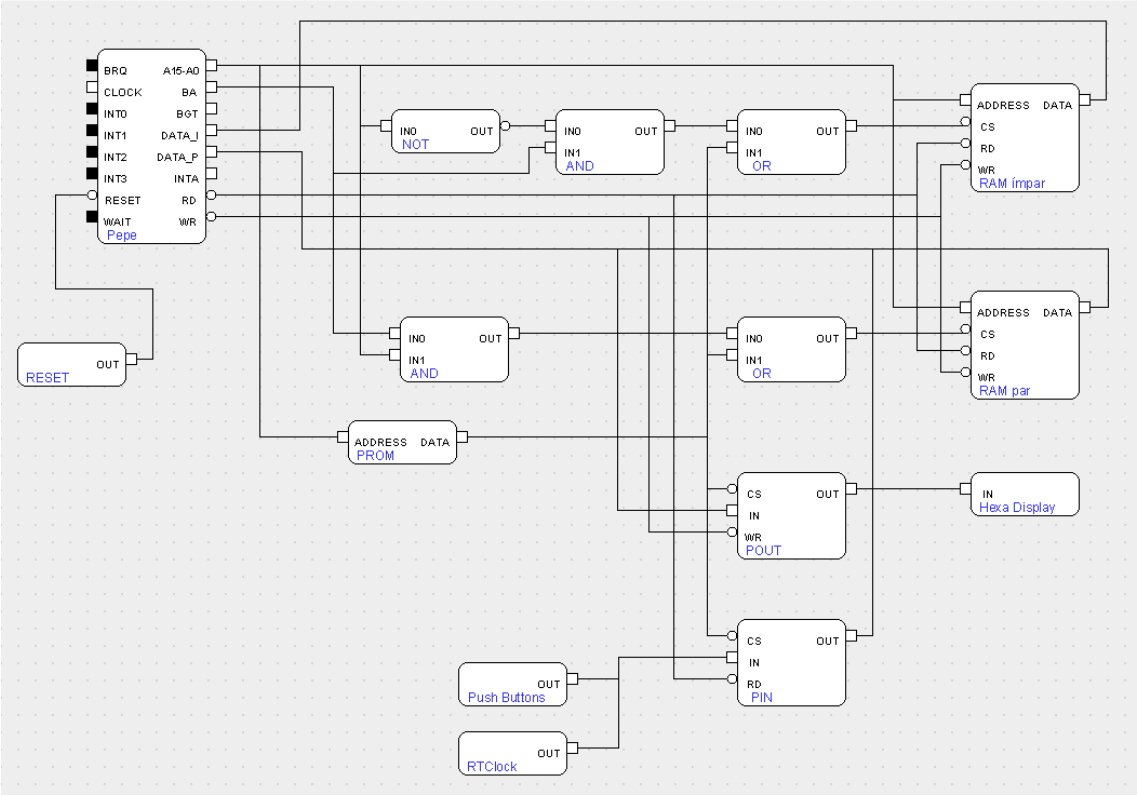
Verifique que a sua funcionalidade se mantém com os novos endereços (o mostrador de 7 segmentos é incrementado enquanto se carrega no botão, ao ritmo do RTC – Real Time Clock). Não se esqueça de fazer START no RTC.

5 – Descodificação de endereços com PROM

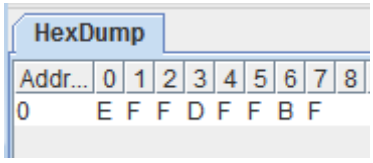
O descodificador pode ser substituído por uma PROM (Programmable ROM), que dá mais liberdade e permite mudar o mapa de endereços sem alterar o hardware.

O circuito seguinte, contido no ficheiro **lab8b.cmod**, usa uma PROM com 3 bits de endereço mas apenas 4 saídas, pois apenas são necessários 3 sinais de CS. Será portanto uma ROM de 8 células, cada uma com 4 bits de largura.





Carregue este circuito no simulador e passe para simulação. Faça duplo clique na PROM e verifique que o seu conteúdo é:



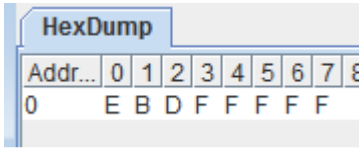
Cada palavra da PROM corresponde a uma combinação dos bits A13..A11 do processador, tal como no descodificador. Nas saídas (D3..D0), apenas um bit deve estar a 0, para ativar um CS.

O mapa de endereços conseguido com esta PROM é igual ao do descodificador. Note que todas as palavras são F (os 4 bits a 1), exceto as palavras 0, 3 e 6, que são as saídas usadas no descodificador. Nestas, note que o único bit a 0 liga a um dado CS, o mesmo a que o descodificador já ligava:

- E (1110) – Bit a 0 na posição 0, CS das RAMs
- D (1101) – Bit a 0 na posição 1, CS do periférico de saída
- B (1011) – Bit a 0 na posição 2, CS do periférico de entrada

Pode experimentar executar o programa **lab8.asm** e verificar que continua a funcionar.

No entanto, e ao contrário do descodificador, para mudar o mapa de endereços basta mudar o conteúdo da PROM. Experimente por exemplo mudar para:



Note que mudou os endereços dos periféricos (os valores B e D). Quais os endereços em que ficaram agora?

Altere também estes endereços no programa **lab8.asm** e execute-o, verificando que ainda funciona (mas nos novos endereços).

Se quiser guardar estes novos endereços, guarde o conteúdo da PROM num ficheiro. Este será carregado automaticamente da próxima vez que carregar este circuito.

O ficheiro **prom.dat** contém o conteúdo já fornecido com o guião.