

Laboratório de Introdução à Arquitetura de Computadores

IST - Taguspark

2016/2017

Interação do processador com memória e periféricos

Guião 4

17 a 21 de outubro de 2016

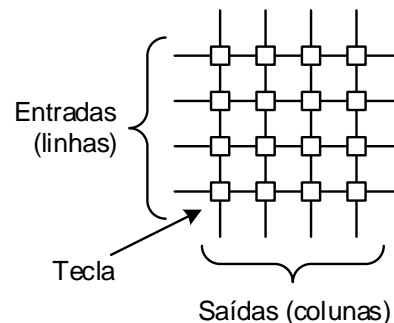
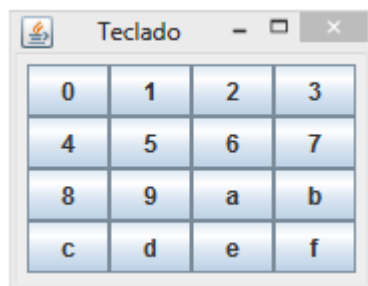
(Semana 5)

1 – Objetivos

Com este trabalho pretende-se que os alunos se iniciem na utilização do microprocessador PEPE (Processador Especial Para Ensino) para acesso a uma memória e a um periférico (um teclado).

2 – Funcionamento de um teclado

O teclado de 16 letras está organizado internamente como uma matriz de 4 linhas por 4 colunas. Neste caso as teclas disponíveis são os números 0 a 9 e ainda as letras de A a F. O que pretendemos fazer do ponto de vista do microprocessador é saber qual a tecla que foi premida.



O teclado não indica diretamente qual a tecla que foi premida. Para descobrir qual foi, o programa tem de testar sucessivamente as várias linhas do teclado, para ver se alguma tecla foi premida. Quando tal acontecer, a tecla faz um curto-circuito entre a linha e a coluna que a definem, fazendo com que o bit presente na linha (seja 0, seja 1) apareça na coluna (saída). Coluna a que nenhuma linha ligue vale 0 no bit de saída.

Aplica-se sucessivamente os valores “0001”, “0010”, “0100” e “1000” em formato binário nas entradas (linhas). Para cada um dos valores anteriores as saídas do teclado (colunas) são lidas. No caso de ser aplicado o valor 0001b nas entradas, e se se obtiver nas saídas o valor 0001b, quer dizer que a tecla que foi premida foi a tecla “0”.

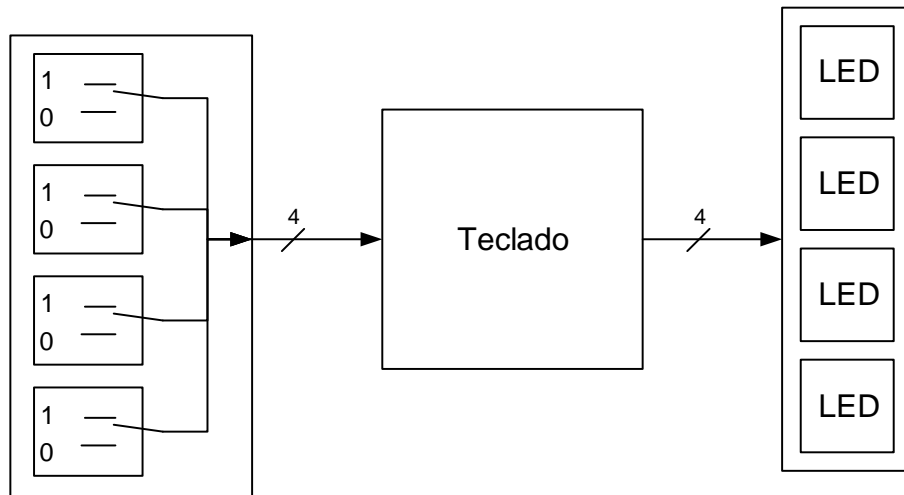
Se o valor for 0010b, quer dizer que foi a tecla “1” que foi premida. Se for 0100b ou 1000b, a tecla premida será “2” ou “3”, respetivamente. No caso de colocar na entrada o valor 0010b, para as 4 possíveis saídas obtêm-se as teclas “4”, “5”, “6” e “7”. Raciocínio idêntico será feito para as teclas da 3ª e 4ª linhas do teclado.

3 – Teste de um teclado

Primeiro vamos testar e verificar o funcionamento de um teclado com um circuito muito simples. O teclado é um periférico que tem o nome de Matriz de pressão.

No simulador, monte um circuito em que:

- Um interruptor de 4 bits liga à entrada da Matriz de Pressão (teclado);
- A saída da Matriz de Pressão liga a um led de 4 bits.

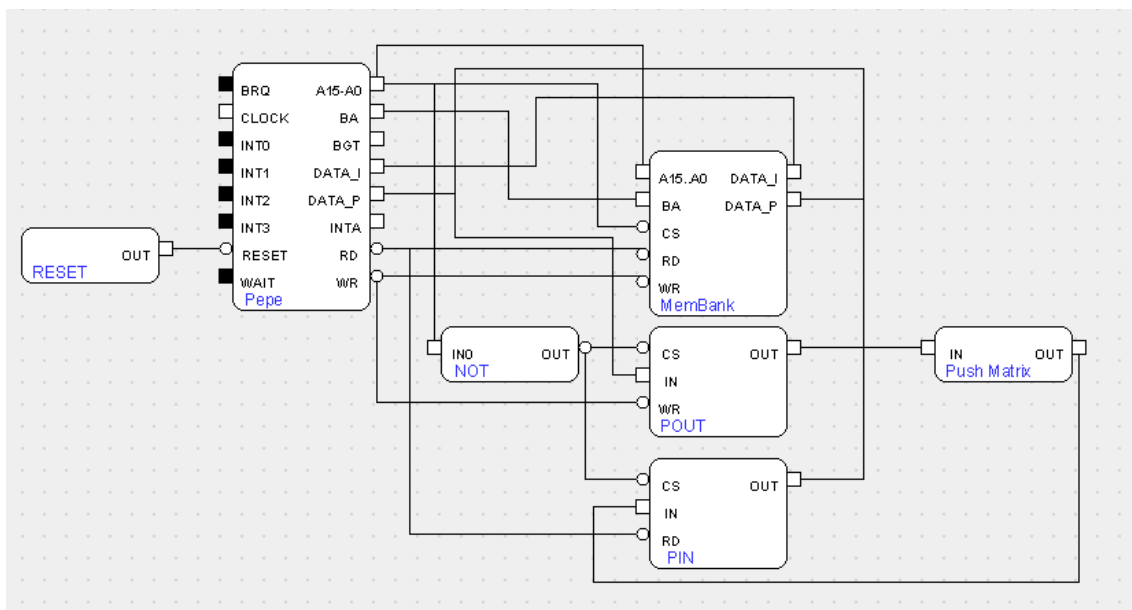


Não se esqueça que os interruptores e leds têm de ter 4 bits.

Experimente colocar o valor 0001b nos interruptores, clique nas várias teclas e verifique o que obtém na saída. Agora experimente sucessivamente com os valores 0010b, 0100b e 1000b, e verifique o que obtém na saída. Por fim experimente com o valor 1111b na entrada, clique nas várias teclas e conclua sobre o funcionamento do teclado. Consegue detetar (pelos leds) qual a tecla carregada se houver mais do que um bit a 1 na entrada?

4 – Leitura do teclado com o processador

O circuito para leitura do teclado com um programa é fornecido já montado (ficheiro **lab4.cmod**). Os ficheiros de ajuda sobre o PEPE estão disponíveis no site da cadeira.



São necessários os seguintes módulos:

- **PEPE** (cujo relógio é gerado internamente)
- **MemBank** (uma RAM de 16 bits com capacidade de endereçamento de byte)
- **PIN** (periférico de entrada)
- **POUT** (periférico de saída)
- **Push Matrix** (teclado de 4 linhas e 4 colunas)
- **Reset** (para inicializar o PEPE sempre que se faz reset no simulador)
- **NOT** (para distinguir os acessos à RAM dos acessos aos periféricos de entrada e saída, com base no A15. Na gama de endereços 0000H a 7FFFH endereça-se a RAM, enquanto em 8000H endereça-se quer o periférico de entrada quer o de saída. O que os distingue é o sinal de RD (leitura) e WR (escrita), que nunca estão ativos ao mesmo tempo.





NOTAS IMPORTANTES:

- Os acessos aos periféricos (quer em escrita quer em leitura) devem ser feitos com a instrução MOVB, pois estes periféricos são de apenas 8 bits. O endereço a usar é 8000H;
- Os acessos à memória podem ser feitos quer com MOV (16 bits) ou MOVB (8 bits).

5 – Execução do trabalho de laboratório

Abra o ficheiro **lab4.asm** (com um editor de texto do tipo NotePad++ para PC ou Brackets para Mac) e tente perceber o que faz, pelos comentários e pelo funcionamento descrito na secção 2.

De seguida, execute os seguintes passos:

1. Em modo de simulação, abra o módulo PEPE e faça Compile&Load () do ficheiro **lab4.asm** (NÃO carregue em START no simulador nem execute o programa do PEPE);
2. Abra o módulo MemBank (duplo clique) e observe as instruções máquina na RAM a partir do endereço 0000H;
3. Execute uma instrução de cada vez, ou de passo a passo, carregando no botão STEP () do PEPE. Para cada instrução executada veja o que ela devia ter alterado e confirme, quer nos registos do PEPE quer na memória (veja em que endereços, no programa, e se o acesso é em byte ou em palavra);
4. Quando o programa entra no “ciclo”, a leitura de uma tecla não pode ser executado em passo a passo, pois são usados os botões de pressão, que não podem ser actuados com o rato ao mesmo tempo que o botão STEP (). Para verificar se alguma tecla foi premida, coloque um ponto de paragem (*breakpoint*) na última instrução (JMP ciclo). Basta fazer clique na linha respectiva, que fica roxa. De seguida execute o programa em modo contínuo (), o que faz automaticamente START ao simulador, e carregue numa tecla

da linha 4 (a de baixo). A linha roxa passa a azul, significando que o simulador parou aí. Verifique que o valor do registo R3 tem o bit a 1 que corresponde à coluna da tecla. Verifique também que o byte na memória de endereço 0100H (BUFFER) tem o mesmo valor, devido à penúltima instrução (MOVB [R5], R3).

Substitua agora a penúltima instrução (MOVB [R5], R3) por MOV [R5], R3. Ou seja, elimine o “B”. O acesso à memória passou a ser de 16 bits.

Guarde o programa alterado, carregue-o no simulador (após fazer RESET no simulador) e repita os passos anteriores. No passo 4, note que o valor da tecla já não está no endereço 0100H mas sim no 0101H.

Isto deve-se ao facto de que a escrita no endereço 0100H (valor indicado pelo R5 na penúltima instrução – agora MOV [R5], R3) é feita em 16 bits e ocupa 2 bytes. O PEPE guarda o byte de maior peso do registo escrito (R3, neste caso) no endereço indicado (por R5, neste caso 0100H) e o byte de menor peso (que contém o valor da tecla) no endereço seguinte (0101H, neste caso). O objetivo é os dois bytes de R3 aparecerem na memória a partir do endereço 0100H pela mesma ordem com que se representam os números (byte de maior peso do lado esquerdo, de menor peso do lado direito).

Com estas experiências, pretende-se que observe:

- O funcionamento da interface do PEPE, em particular dos registos;
- O comportamento dos acessos à memória em byte e em 16 bits;
- A escrita e leitura de periféricos de entrada e saída, em que a atuação dos sinais de WR e RD destes módulos é feita automaticamente pelas instruções de escrita e leitura do PEPE;
- Se alguma tecla foi pressionada e qual.

6 – Objetivos a cumprir para o Projeto

O programa do ficheiro **lab4.asm** dá apenas para uma tecla na linha 4. Pretende-se detetar qualquer tecla em qualquer linha. O teclado é um componente essencial para a execução do projeto.

Na semana seguinte à deste guião, deve ser mostrar ao docente um programa que faça o varrimento do teclado e detete a tecla premida, obtendo o seu valor (0 a FH).

Para tal, deve fazer os passos seguintes:

- Faça um programa (pode usar o programa do ficheiro **lab4.asm** como base) que varre continuamente, em ciclo, as várias linhas do teclado, verificando se alguma das teclas foi premida e, quando tal acontecer, escreve na memória (a partir do endereço 0100H, por exemplo) o valor colocado na entrada do teclado (linhas) e o valor lido na saída (colunas);
- Com um *breakpoint* numa última instrução (ou pelo menos a seguir ao MOV que escreve a tecla na memória), verifique na memória se a informação detetada está correta;
- Mas, na realidade, o que quer obter é um valor entre 0 e FH, correspondente ao valor da tecla, em vez de apenas informação sobre a linha e coluna. Acrescente código para converter linha/coluna no valor entre 0 e FH. Sugestão: o valor da

tecla é igual a $4 * \text{linha} + \text{coluna}$, em que tanto linha como coluna são números entre 0 e 3. Logo, terá de converter 0001b, 0010b, 0100b e 1000b (valores usados no varrimento) em 0, 1, 2 e 3, respetivamente. Agora, o valor a escrever na memória deve ser o valor da tecla, e não linha e coluna;

- Mas, mesmo realmente, o que quer é detetar várias teclas, à medida que forem sendo premidas. Para tal, sempre que escrever o valor de uma tecla em memória, incremente um registo que serve de endereço a escrever. Quando escrever 16 valores (faça um teste ao valor deste registo para saber se já chegou ao fim), faça um JMP para uma instrução qualquer onde deve colocar um *breakpoint*. De seguida, execute o programa (🔍), vá carregando nas teclas e quando o programa terminar (a barra do *breakpoint* passar de violeta a azul) vá verificar se os valores na memória correspondem às teclas que carregou;
- Na eventualidade de o programa detetar continuamente a mesma tecla, tente perceber o que se está a passar. Normalmente, a máquina tem sempre razão...

A tabela A.9 do livro contém informação sobre cada instrução do PEPE.

O objetivo deste exercício é fazer o software básico de leitura do teclado, que constitui o coração do controlo do programa do projeto. Será apenas questão de, mais tarde, adaptar o que se faz com o valor da tecla lida.