# Guião de Laboratório
# de
# Arquitectura de Computadores (2ª edição)

## Simulação 7.7 – Exclusão mútua

### 1 – Objectivos

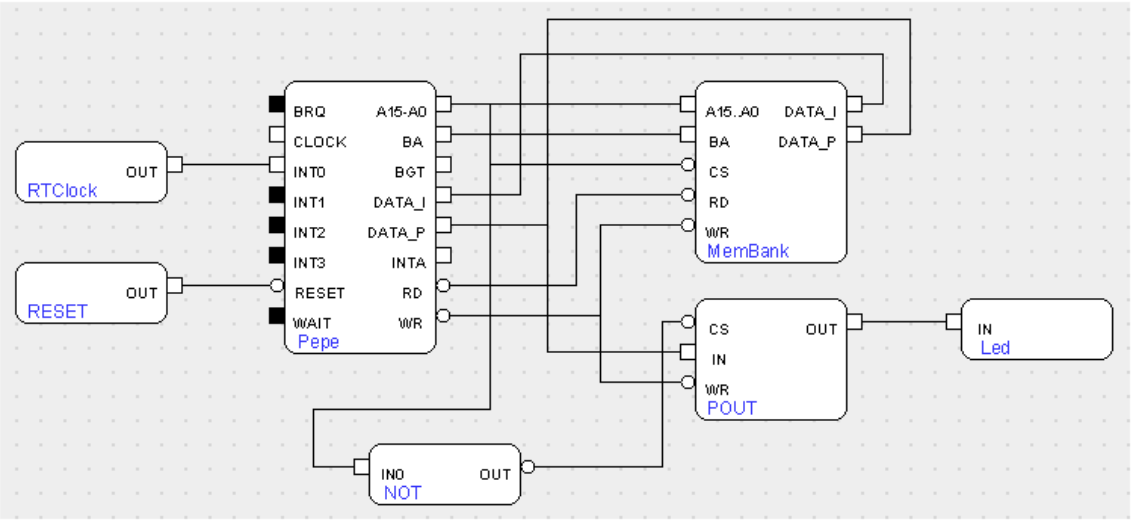
Esta simulação toma como base o Programa 7.3 e exemplifica o funcionamento do mecanismo de exclusão mútua no PEPE. São focados em particular os seguintes aspectos:

- Efeito da inclusão do tempo de espera antes da actualização da imagem dos LEDs;
- Atomicidade da instrução SWAP;
- Funcionamento das rotinas de acesso ao trinco lógico.

### 2 – Circuito

O ficheiro "multiprogramacao.cmod" implementa o circuito da figura seguinte, que consiste apenas no PEPE, no banco de memória, um periférico ligado a oito leds e um relógio de tempo real, ligado à interrupção 0.

Este circuito é igual ao já usado na simulação 7.6.




Carregue este circuito no simulador e passe para Simulação.

### 3 – Simulação do programa 7.2 modificado

Abra o painel do PEPE e compile e carregue (  ) o ficheiro "programa7-2-modificado.asm", que foi obtido a partir do programa 7.2 (já usado na simulação 7.6) por troca dos passos 3 e 5 da Fig. 7.33, tal como referido na secção 7.7.3.1.

Esta alteração destina-se a aumentar a probabilidade de ocorrência do problema ilustrado pela Fig. 7.33, de alteração da imagem dos leds na memória pelo processo Alterna no meio do tratamento a esses leds por parte do processo Pisca. Este problema está descrito em mais detalhe na secção 7.7.3.1.

Abra o painel dos leds e do relógio, onde deve garantir que o período está na ordem de 50 a 100 (para garantir uma troca rápida entre os processos). Execute o programa em contínuo (botão  no painel do PEPE) e observe o piscar irregular (às vezes) dos leds (nota-se mais no alternar dos leds 2 e 3).

### 4 – Simulação do programa 7.3

Abra o painel do PEPE e compile e carregue (  ) o ficheiro "programa7-3.asm", que foi obtido a partir do programa 7.2 (já usado na simulação 7.6) pela inclusão da chamada às rotinas Entrar e Sair antes e depois, respectivamente, da leitura e modificação da estrutura de dados partilhada (neste caso, a imagem dos leds em memória), tanto no processo Pisca como no processo Alterna.

Estas rotinas (Entrar e Sair) delimitam zonas de código designadas secções críticas, e usam uma instrução SWAP (entre um registo e uma memória, designada Trinco), para garantir que apenas um dos processos entra na secção crítica. O outro, se o tentar, fica em ciclo à espera que o outro termine e liberte a secção.

Desta forma, garante-se que uma vez que um processo leia a imagem dos leds, pode alterá-la com a garantia que nenhum outro processo alterará essa imagem até que ele tenha terminado de processar essa imagem dos leds.

A este mecanismo chama-se exclusão mútua e é fundamental em sistemas operativos.

Olhe fixamente para os leds 2 e 3 e veja que o seu piscar se mantém regular.

O funcionamento da instrução SWAP é simples. De forma atómica (é uma só instrução, e por isso não pode ser interrompida por uma interrupção), faz uma leitura e uma escrita a uma célula de memória, trocando o seu valor com um registo.

A rotina Entrar coloca 1 no registo R1 e executa o SWAP. A célula de memória (Trinco) fica com esse 1 e o R1 fica com o valor que estava na célula de memória. Se lá estava 1, o trinco estava ocupado e tem de esperar (espera activa, em ciclo. É uma espera desperdiçada, mas no máximo dura o tempo que o processador dedica a esse processo). Se lá estava 0, é porque o trinco estava livre, e entretanto já foi ocuopado. Logo, se for o caso, este processo pode prosseguir, pois todos os outros que tentem fazer o mesmo (executar o SWAP) já lêem o valor 1. No fim das alterações, é só invocar a rotina Sair, que apenas coloca 0 na variável Trinco (sem preocupações, pois é só uma escrita). A partir daqui, o próximo processo que tentar executar o SWAP pode avançar.