Guião de Laboratório
de
Arquitectura de Computadores

Simulação 3.4 – PEPE-8: programação em assembly

1 – Objectivos

Esta simulação ilustra o funcionamento do PEPE-8 (circuito da Fig. 3.18) com programação em linguagem assembly, usando o Programa 3.4 e a informação da Tabela 3.11.

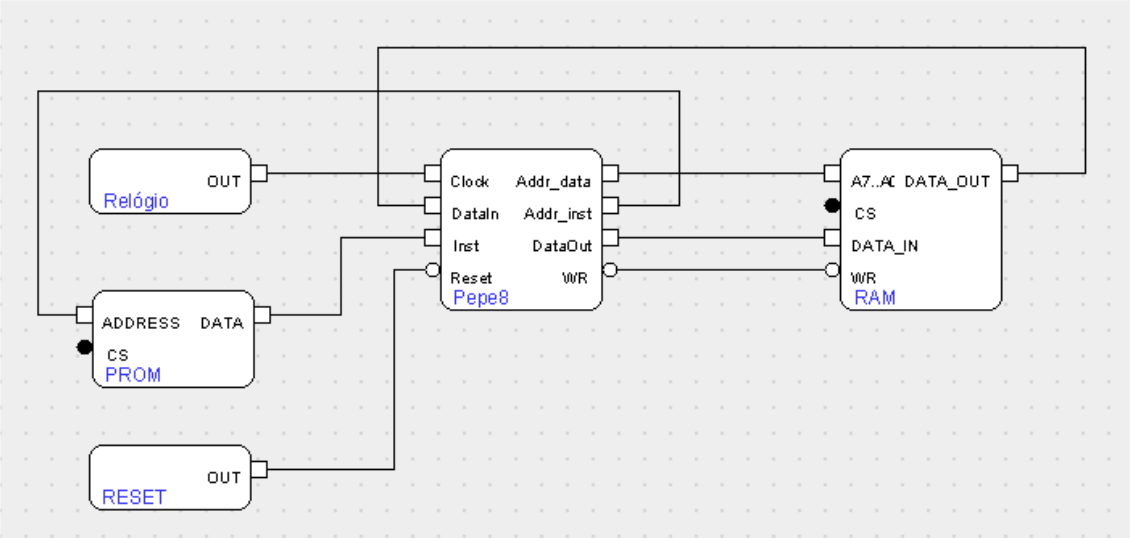
Os aspectos cobertos incluem os seguintes:

- Montagem do circuito com um módulo que simula o PEPE-8 ao nível das instruções;
- Funcionamento individual das várias instruções do PEPE-8, em termos dos recursos de hardware e sinais de controlo que envolvem;
- Execução do programa em ambiente de desenvolvimento:
  - Execução das instruções passo a passo (single-step), verificando a evolução dos registos A e PC;
  - Utilização de pontos de paragem (breakpoints);
  - Modificação manual do valor dos registos durante a execução do programa (após uma paragem motivada por execução passo a passo ou ponto de paragem).

2 – Circuito

O ficheiro "pepe-8.cmod" implementa o circuito da Fig. 3.18, mas em que o PEPE-8 está implementado num módulo, simulado funcionalmente e não por blocos de hardware constituintes.

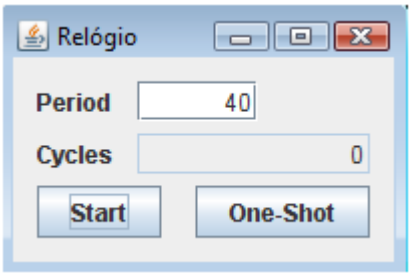
O módulo Reset dá um impulso na entrada Reset do PEPE-8 quando se faz START na simulação, para inicializar o processador.



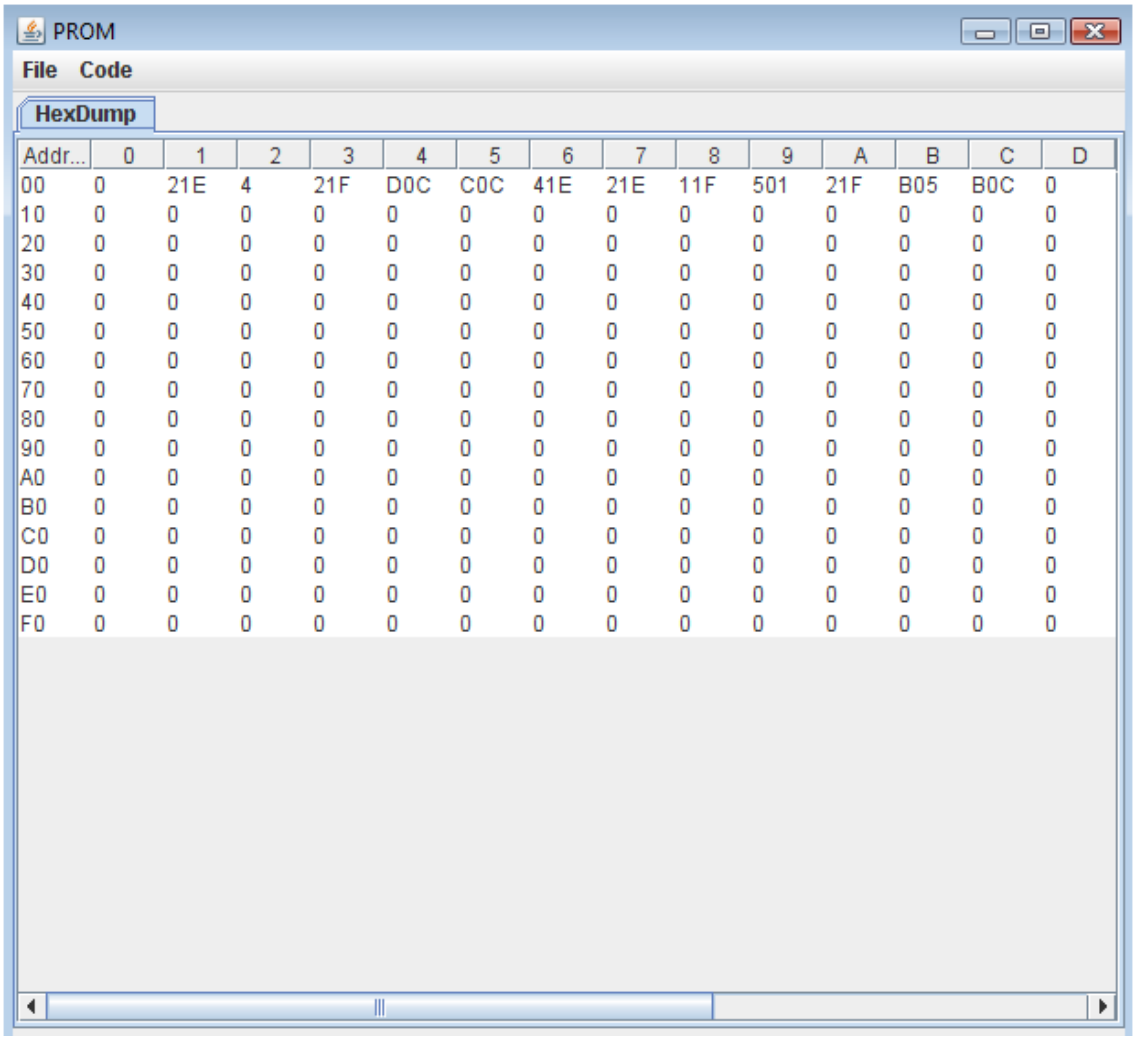
3 – Compilação e carregamento dos programas em assembly

O PEPE-8 assume que tem uma RAM na memória de dados e uma PROM na memória de instruções. Para compilar, carregar e executar um programa de assembly, tem de se fazer os passos seguintes:

1. Carregar a arquitectura "pepe-8.cmod"
2. Passar para Simulação
3. Abrir o painel de controlo do Relógio com duplo clique neste módulo e carregar em START nessa janela



4. Abrir o painel de controlo do PEPE-8 com duplo clique neste módulo
5. Carregar em Compile e escolher o ficheiro com o programa em assembly (nesta simulação, o ficheiro "programa3-4.asm"). Aparentemente o PEPE-8 não faz nada, mas gera o ficheiro "programa3-4.cod" no mesmo directório, que contém as instruções em binário para o PEPE-8 executar
6. Abrir o painel de controlo da PROM com duplo clique neste módulo
7. No menu desta janela, use o comando Code -> Load Binary (note que NÃO é File -> Load) e especifique o ficheiro "programa3-4.cod". Neste momento, já aparece na PROM as instruções em binário a executar pelo PEPE-8



Addr...	0	1	2	3	4	5	6	7	8	9	A	B	C	D
00	0	21E	4	21F	D0C	C0C	41E	21E	11F	501	21F	B05	B0C	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

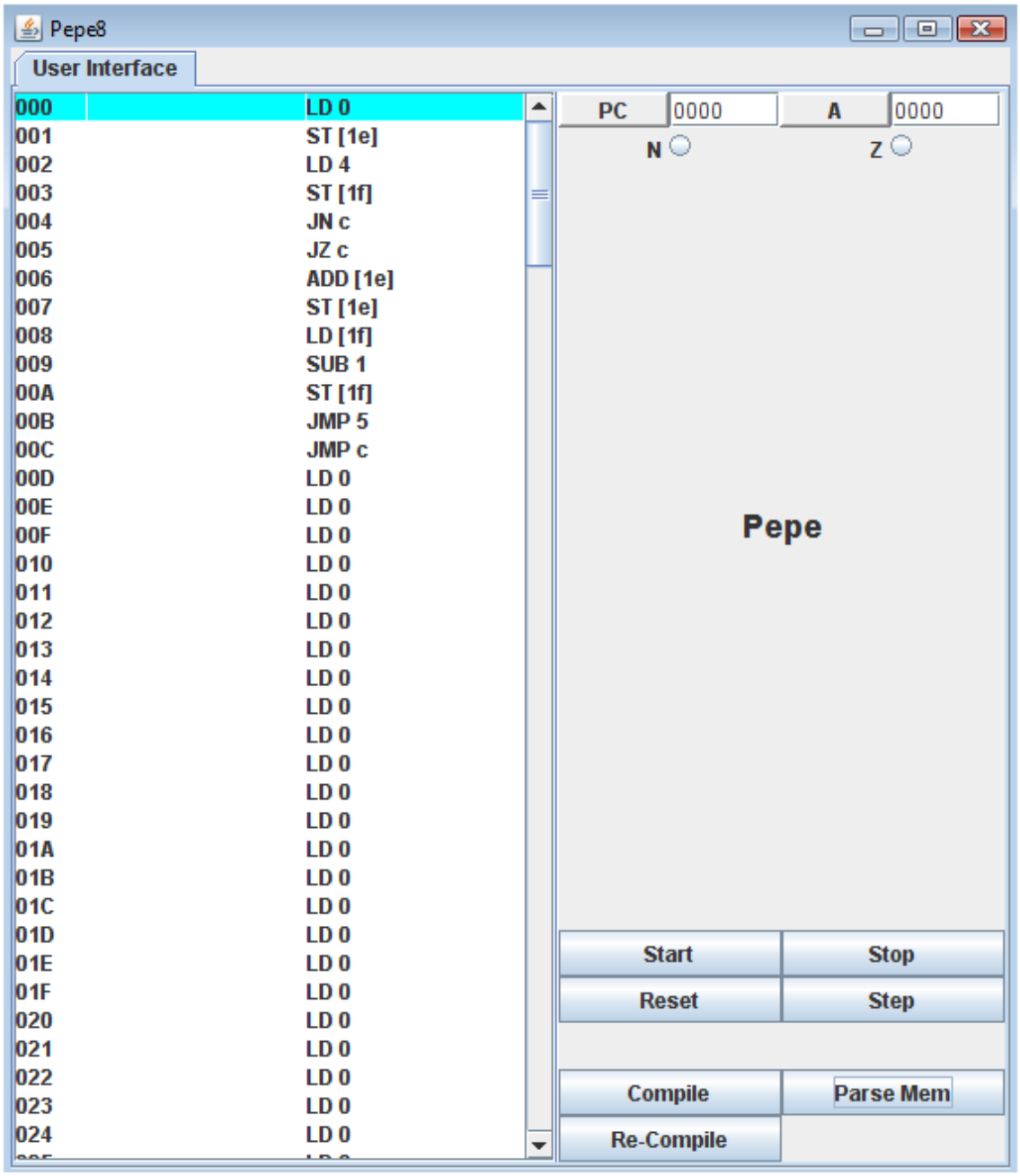
8. No painel do PEPE-8, carregue no botão “Parse Mem”, o que faz o PEPE-8 ler a PROM e actualizar a sua janela com as instruções, que apresenta em linguagem assembly por conversão a partir das instruções em binário. No entanto, note que já não tem informação sobre as etiquetas e símbolos definidos com EQU no programa fonte

Neste momento, o programa está pronto a ser executado, o que pode ser feito de três formas:

- Passo a passo (single-step), ou instrução a instrução
- Execução contínua até um ponto de paragem (breakpoint)
- Execução contínua

O painel do PEPE-8 mostra o conteúdo do PC e do registo A e indica ainda se o valor do A é zero (Z) ou negativo (N).

O conteúdo da memória de dados pode ser verificado com duplo clique na RAM.



4 – Execução passo a passo

Carregue no botão STEP do painel do PEPE-8. De cada vez que o faz, avança uma única instrução. A próxima instrução a ser executada aparece com uma barra azul.

Vá carregando em STEP e verifique o seguinte (se necessário, execute a simulação várias vezes para ver os vários aspectos):

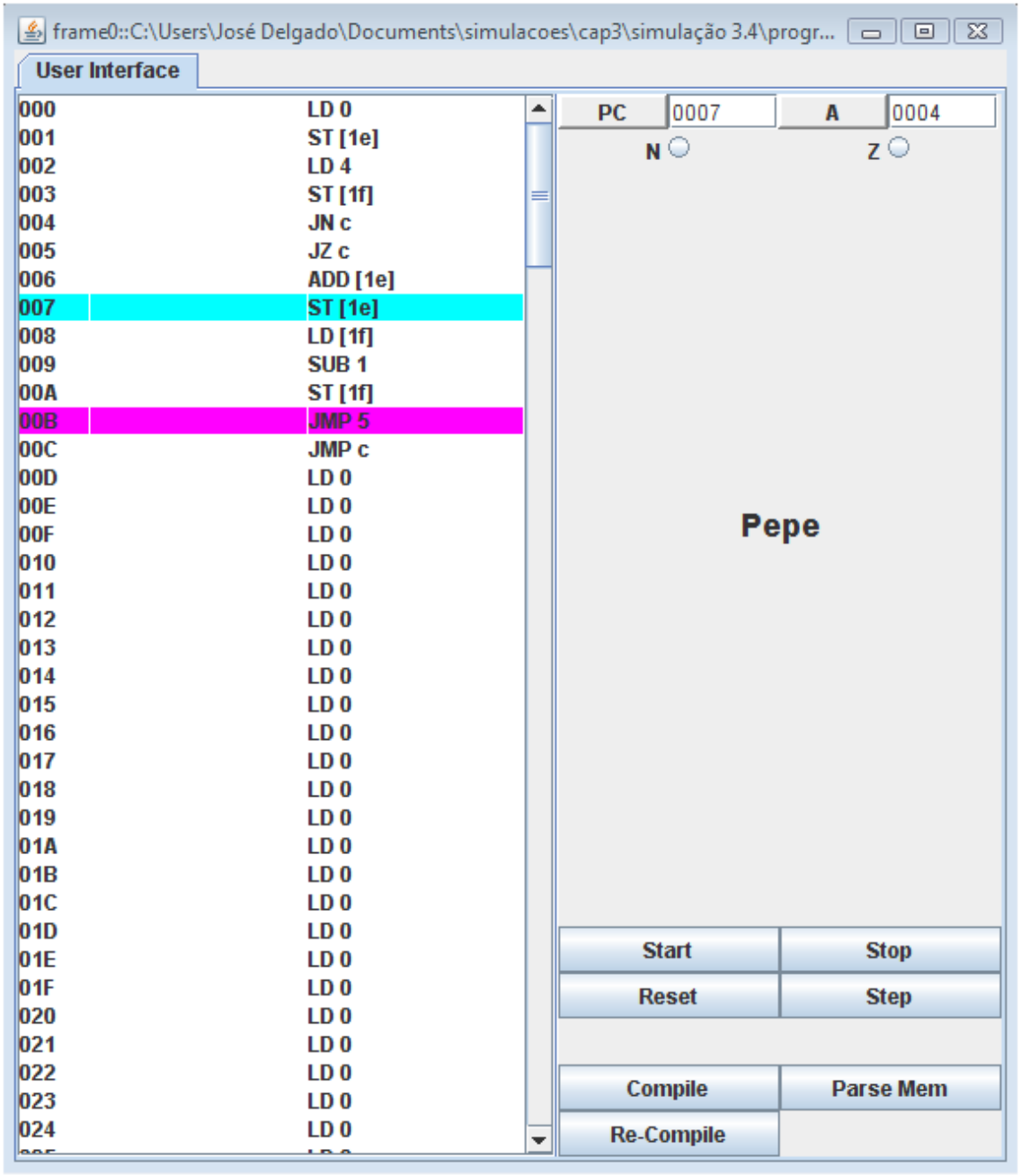
- No painel do PEPE-8, o PC vai variando de forma consistente com a barra azul;
- No painel do PEPE-8, o registo A vai indicando o seu valor;
- No painel da RAM, as instruções ST alteram o conteúdo da RAM nos endereços indicados (1EH e 1FH, que correspondem às variáveis soma e temp);

- No circuito, os sinais que ligam ao PEPE-8 (cujo valor pode ser visto colando o cursor em cima deles) indicam o que se está a passar em termos das memórias, em particular os buses de endereço e de dados das duas memórias (conferir com as Fig. 3.19 a 3.25).

Note que em qualquer altura pode mudar o conteúdo dos registos (até o PC, mas tal pode implicar um funcionamento incorrecto do programa) e da memória de dados.

5 – Execução com pontos de paragem

Os pontos de paragem podem ser definidos fazendo clique em cima da instrução em que se pretende que a execução pare, fazendo aparecer uma barra roxa.



Fazendo START no painel do PEPE-8, este começa a executar as instruções à sua velocidade máxima (mesmo que antes tivesse estado em passo a passo. Neste caso, a execução começa a partir do ponto em que estava).

Se e quando uma instrução com um breakpoint é atingida, o PEPE-8 pára a sua execução imediatamente antes de a executar. Nessa altura, podem inspeccionar-se o registo ou a memória, fazer-se STEP, etc.

Durante a execução com START, os valores no painel do PEPE-8 não são actualizados (só quando parar).

Para eliminar um ponto de paragem, basta carregar na barra roxa em causa.

6 – Execução sem pontos de paragem

Neste caso o processador não pára. No entanto, pode carregar-se em STOP no painel do PEPE-8 e ele pára na instrução que estiver a executar nesse momento.

Nessa altura pode inspeccionar ou alterar os registos e a memória.

A partir daí pode fazer STEP, START ou definir breakpoints e fazer START.