Arquitetura de Computadores

3º e 4º Trabalho de Laboratório
Programação Assembly – Mastermind (Parte I e II)

Objetivo: Pretende-se no 3º e 4º trabalho de laboratório que os alunos compreendam a metodologia usada no desenvolvimento de programas em linguagem Assembly, incluindo o uso de periféricos e de rotinas de interrupção. A programação será desenvolvida para o processador P3, sendo utilizado um assemblador e um simulador disponibilizados para o efeito. O trabalho a desenvolver corresponderá aos 2 laboratórios e será organizado em 2 fases conforme descrito e detalhado neste enunciado. O trabalho deverá ser preparado fora do horário de laboratório, destinando-se as 3 horas de aula à resolução de eventuais dúvidas e demonstração do trabalho realizado.

1 INTRODUÇÃO

Pretende-se implementar no conjunto dos trabalhos 3 e 4 de laboratório uma versão do jogo Mastermind (https://en.wikipedia.org/wiki/Mastermind_(board_game) e http://mathworld.wolfram.com/Mastermind.html). O jogo a implementar deve permitir não só que a máquina gere uma sequência de cores e que avalie as tentativas do jogador (lab3) como também que a máquina seja capaz de adivinhar uma sequência definida pelo jogador (lab4). O trabalho a desenvolver utilizará os diversos periféricos associados ao simulador do processador P3, conforme indicado no enunciado.

Figura 1 – Tabuleiro tradicional de Mastermind.

A primeira fase do trabalho envolve: (1) perceber o funcionamento do jogo; (2) definir a estrutura de dados que permita representar o tabuleiro de jogo; (3) representar o tabuleiro na Janela de Texto do P3; (4) gerar uma sequência aleatória de cores (que no P3 deverá ser representado por um símbolo ASCII); (5) ler uma sequência de cores do teclado correspondente à jogada; (6) implementar a avaliação da jogada, guardar o resultado em memória e apresentar na Janela de Texto; (7) registar o tempo de jogo no Display de 7 Segmentos.

2 REPRESENTAR TABULEIRO, CRIAR SEQUÊNCIA ALEATÓRIA E AVALIAR JOGADA (1ª SEMANA)

Nesta fase deverá começar por compreender o funcionamento do jogo, definir a estrutura de dados que permita guardar a informação do jogo em curso, planear a interface com a Janela de Texto para: definir o écran inicial; escolher uma representação para o estado "actual" do jogo; receber jogada do teclado. **Sugestão**: analise o conjunto de exemplos fornecidos e realize testes para perceber o funcionamento das diferentes interfaces antes de escrever o código final.

2.1 Representar tabuleiro

Na representação do tabuleiro, sendo a Janela de Texto monocromática, deve escolher caracteres para representar as cores, por exemplo, (B – branco, P – preto, E – encarnado, V – verde, Z – azul, A – amarelo), e as cores para definir a resposta (P – acerto na cor e posição; B – cor certa mas posição errada). O tabuleiro deve ter 10 linhas correspondentes à representação das 10 tentativas de 4 cores (versão standard) e à representação da correspondente avaliação da jogada.

2.2 Sequência aleatória

Para gerar a sequência aleatória de 4 cores (versão standard) deverá implementar um gerador de números aleatórios, por exemplo, como descrito no Algoritmo I.

```
Mascara = 1001 1100 0001 0110b

    if (Ni0==0) /* Testa o bit de menor peso de Ni */
            Ni+1 = rotate_right(Ni);
    else
            Ni+1 = rotate_right (XOR (Ni, Mascara))
```

Algoritmo 1 – Geração de um número pseudoaleatório de 16 bits.

Este algoritmo baseia-se na simulação de um registo de deslocamento modificado com realimentação, que permite gerar uma sequência pseudoaleatória de números de 16 bits, com um passo de repetição longo e com uma distribuição uniforme (i.e., os números são equiprováveis). Em cada invocação desta função lê-se o valor anterior Ni e gera-se um novo valor pseudoaleatório, Ni+1. O valor de Ni deve ser inicializado com um valor diferente de zero (seed). Para a realização do trabalho, vai ser necessário obter um número pseudoaleatório entre zero e M-1 (para representar as cores, M=6 na versão standard). Para obter este número sugere-se que determine o valor do símbolo menos significativo do numero Ni quando representado na base M, i.e., divida Ni por M e obtenha o resto. Note que não deve modificar o valor de Ni, mas calcular um novo número, por exemplo, Zi = resto da divisão de Ni por M.

A sequência obtida deve estar armazenada em memória podendo o jogador visualizá-la no tabuleiro sempre que pretender (através da **interrupção 0**) levando assim ao fim do jogo. A sequência deve também ser mostrada no tabuleiro quando o jogador acerta ou quando atinge o limite das 10 jogadas.

2.3 Leitura e avaliação da jogada

Para permitir a execução do jogo, deve ser feita a leitura da jogada do teclado (sequência de 4 cores) com a respectiva validação e apresentação na Janela de Texto. De seguida, a jogada deve ser avaliada comparando com a sequência gerada aleatoriamente. O resultado da avaliação de cada jogada deve ser apresentado na Janela de Texto na respectiva linha.

3 ÉCRAN INICIAL, CRONÓMETRO E DEMO DA 1ª FASE (2ª SEMANA)

Na 2ª semana, os alunos devem concluir a primeira fase do trabalho (lab3), realizando a demonstração do trabalho no laboratório.

3.1 Écran Inicial

Deve ser criado um écran inicial de apresentação, na Janela de Texto, onde sejam indicadas as regras do jogo, os comandos do jogo e as opções relativas à versão implementada por cada grupo.

3.2 Cronómetro

Deve ser implementado um cronómetro para medir o tempo gasto pelo jogador no Display de 7 Segmentos. O funcionamento deve ser o seguinte: assim, que se inicia o jogo o cronómetro deve iniciar a contagem decrescente, nos **2 minutos**, e parar depois do jogador ter executado a jogada e voltar a reiniciar depois do programa dar a avaliação da jogada. O cronómetro deve parar quando o jogo acaba. Para implementar este cronómetro deve utilizar a interrupção 15 – Temporizador.

4 MULTI-JOGADOR, NÍVEIS DE JOGO E ESTATÍSTICAS (3ª SEMANA)

Na 2ª fase do trabalho pretende-se progressivamente enriquecer a solução implementada com novas funcionalidades.

4.1 Versão Multi-Jogador

A versão implementada deve agora permitir definir vários jogadores com o objectivo de competirem por acertar nas sequências de cores em menos jogadas. Cada jogador deve fazer um jogo completo e só depois passa a vez ao jogador seguinte.

4.2 Níveis de Jogo

O jogo deve agora ser melhorado introduzindo níveis de jogo que permitam alterar o número de cores da sequência e o tempo de jogo.

Assim, a solução implementada deve agora permitir ao jogador escolher o número de cores 4, 5 ou 6 e definir o tempo de jogo entre 1, 2 ou 4 minutos. A escolha das opções de jogo deve ser feita num écran que deve surgir após o écran inicial de apresentação.

Deixa-se como sugestão que ao pensar na estrutura de dados e na escrita do código o faça de forma a permitir facilmente a sua alteração para implementar os diferentes níveis de jogo.

4.3 Estatísticas

O LCD deve ser utilizado para apresentar as estatísticas do jogo como: o melhor tempo de jogo, a média dos tempos de jogo (arredondado ao segundo), a média do número de jogadas para acerto (arredondado às décimas). Estes valores devem ser actualizados no fim de cada jogo. As estatísticas do jogo devem ainda ser realizadas para cada jogador definido em 4.1.

5 DEMO E AVALIAÇÃO DA 2ª FASE (4ª SEMANA)

Na última semana serão realizadas as demos dos trabalhos onde o docente terá oportunidade de questionar os alunos sobre os vários aspectos do trabalho.

6 FASEAMENTO DO TRABALHO E RELATÓRIO

O relatório e o código comentado devem ser entregues via Fénix até à sexta-feira da 2ª semana do laboratório, ou seja até às 23h59 do dia 2 de Junho. No Fenix será ainda criado um link para submissões fora de prazo com penalização de 2^n em que n é o número de dias de atraso.

O relatório (não incluindo o código e a capa) não deve exceder as 10 páginas. O relatório deve ser fornecido no formato ".pdf" e o código em formato de texto, ".as" e em formato ".pdf", tudo num ficheiro zip submetido via Fénix. Caso sejam submetidos múltiplos ficheiros, apenas será tomado em consideração o último ficheiro submetido.

7 AVALIAÇÃO DO TRABALHO

A **avaliação** do trabalho será realizada, **ao longo das duas aulas de laboratório** de acordo com as cotações indicadas de seguida.

- Representação do Tabuleiro (2 val.)
- Sequência Aleatória (2 val.)
- Leitura da Jogada (1 val.)
- Avaliação da Jogada (1 val.)
- Cronómetro do Jogador no Display de 7 Segmentos (2 val.)
- Estatísticas de Jogo no LCD (2 val.)
- Níveis de Jogo (2 val.)
- Versão Multi-Jogador (2 val.)
- Extras Escolhidos pelo Grupo (2 val.)
- Código e Estrutura de Dados (2 val.)
- Comentários (2 val.)

Nota 1: a criatividade nas soluções apresentadas ao longo do trabalho é valorizada.
Nota 2: tudo o que não estiver especificado deve ser considerado como um grau de liberdade para a implementação de cada grupo de alunos.

8 IMPLEMENTAÇÃO DA VERSÃO JOGADOR VS MÁQUINA [OPCIONAL]

A título opcional e podendo apenas valorizar o trabalho deixa-se ao critério dos alunos implementar um algoritmo/estratégia de jogo para permitir a solução Jogador vs Máquina.

Neste caso a sugestão é implementar o algoritmo desenvolvido por Donald Knuth e que pode consultar em: https://en.wikipedia.org/wiki/Mastermind_%28board_game%29#Five-guess_algorithm ou outra alternativa à escolha.

A implementação deve ter em conta os recursos de memória disponíveis e o tempo de processamento. Em termos de visualização, a sequência gerada pelo jogador deve ficar visível na Janela de texto, o programa deve colocar na Janela de Texto a sua jogada e o jogador deve introduzir através do teclado a avaliação da jogada. O programa deve permitir observar no Display de 7 Segmentos o tempo de jogo do processador e no LCD as respectivas estatísticas.

Nota: Sugere-se que apenas os grupos com a parte obrigatória do trabalho concluída e validada considerem esta opção e procurem o acompanhamento dos docentes.

9 BIBLIOGRAFIA

[1] N. Horta, "Arquitecturas de Computadores", Aulas Teóricas, 2017.
[2] G. Arroz, J. Monteiro, A. Oliveira, "Arquitectura de Computadores: dos Sistemas Digitais aos Microprocessadores", IST Press, 2007.