Introdução à Arquitectura de Computadores
Relatório do projecto
Pássaro Bamboleante

Luisa Santo 79758
Pedro Miguel Orvalho 81151
Gonçalo Simões 81031

Instituto Superior Técnico
Universidade de Lisboa

2 de Dezembro de 2015
Quarta-Feira LAB 10

Conteúdo

Lista de Tabelas

1 Introdução

Neste trabalho foi projetado e implementado um jogo. A sua implementação baseou-se num pássaro cujo objectivo é percorrer a maior distância possível sem chocar contra obstáculos de acordo com as leis da física. O jogo arranca quando o botões de pressão I1 é premido. A velocidade com que os objetos se deslocam é constante e as suas eventuais colisões podem ser evitadas premindo o botões de pressão I0, que faz o pássaro saltar. A velocidade do passáro depende do nível de dificuldade do jogo – o jogo inicia no nível 1, com o LED mais à esquerda aceso e, o nível aumenta pressionando o botão de pressão I2, até ser atingido o nível 16. Se existir sucesso na passagem pelo obstáculo, a pontuação é incrementada. Caso contrário, o jogo acaba e reinicia quando I1 é premido ou termina quando qualquer outra interrupção é pressionada.

Esta solução foi implementada em linguagem Assembly para ser executado no P3, tendo-se simulado extensivamente. Por fim, em laboratório, implementou-se o projecto na placa com a versão *hardware* do P3.

Apresentamos de seguida a forma como procedemos ao desenvolvimento do projeto, abordando os aspetos e rotinas mais importantes.

2 Conceção e Implementação

2.1 Estrutura Geral

De um modo geral, este projecto recorre a uma fase de inicialização seguida de um ciclo principal que é composto pela chamada de várias funções. Pode-se facilmente observar a estrutura do projecto através do fluxograma na secção em Anexo.

A fase de inicialização consiste na desclaração de múltiplas variáveis e constantes de jogo de apoio ao programa e na chamada de rotinas de inicialização. Esta fase está encarregue de escrever uma mensagem na janela de texto e esperar que I1 seja pressionado. De seguida corremos o restante código de inicialização que trata de inicializar a pontuação, os *displays* e o nível.

A fase seguinte corresponde ao ciclo principal – *CicloJogo* – cujo objectivo é atualizar vários parâmetros que permitem a execução do jogo. Esses parâmetros correspondem, entre outros, à posição dos obstáculos e do pássaro, à pontuação e ao nível atual. De notar que este ciclo se divide em várias rotinas que por sua vez se subdividem em várias rotinas de apoio. Um exemplo desse caso é a rotina *Update* que utiliza outras rotinas: *UpdateBird* e *GravityFX*. Estas últimas desempenham funções simples que podem ser repetidas ao longo do programa e funcionam como uma forma de abstrair os procedimentos e evitar a repetição de código.

Para uma melhor compreensão do jogo, ver em Anexo o manual do jogo, na secção 4.1, e o manual de teclas, na secção 4.2.

2.1.1 Mapa de endereçamento

As diferentes partes do programa em cima descrito estão localizadas em partes diferentes da memória do P3. Por definição, e atendendo também a questões de eficiência, reservámos os endereços inicias de memória para os dados; as primeiras posições de memória a partir do endereço 8000h para declarações de variáveis; e as últimas posições para a tabela de interrupções e portos para aceder aos vários dispositivos de entrada/saída disponíveis, tais como o display de 7 segmentos e um conjunto de LEDs e a pilha. Ver tabela 3 em Anexo.

2.1.2 Rotinas

O projecto foi implementado com auxilio de várias rotinas, de modo a que este fique mais eficaz e melhor organizado. Entre as várias rotinas, as que verificam um papel mais importante na abordagem do programa são as seguintes:

- **Rotina Escrita:** recebe a posição onde vai escrever a *String* na Janela de Texto. Esta rotina, *EscString* suportada pela rotina *EscCar*, é usada de forma a evitar repetição de código quanto à escrita de string e o seu posicionamento. A rotina é assim usada pela função de escrita das frases de início de jogo e de fim de jogo assim como na escrita do pássaro e o seu update no decorrer do programa.

- **Rotina de Limpeza:** recebe a posição que se quer limpar no caso das rotinas *LimpaVector*, *LimpaPista* e *ApagaColunas* ou reinicia os *dis-*

plays no caso da rotina *ResetStats*. Esta abordagem facilita a leitura do código dado que estas rotinas são chamadas várias vezes, tornando o programa menos denso e menos alongado.

- **Rotina de Interrupção:** incrementa as variáveis auxiliares. Esta abordagem apresenta vantagens, entre elas:
  - torna o código mais rápido por serem pequenas;
  - interrompe o código quando é necessário, nao sendo necessário a utilização de quase nenhuns ENIs e DSIs, , através da avaliação das variáveis auxiliares;
  - activa a maior parte das interrupções usando apenas a mesma máscara (exemplo: *INT_MASK*).
- **Rotinas de Espera:** pára o programa até que determinada condição seja cumprida. Um exemplo dessa rotina é a rotina *CicloPausa*;
- **Rotina Random:** muda o valor da variável *NumAleatorio* para um número aleatório, de modo a que possa ser acedido noutras rotinas. A rotina *AlturaAleatoria* usa este número para a escrita de um novo obstáculo – necessário para saber em que linha o espaço entre as colunas irá ficar.
- **Rotina *Update*:** cria as condições necessárias para ser possível a continuação do jogo no *CicloJogo*, tais como a definição da velocidade do pássaro e a atualização da posição do mesmo, simulando também os efeitos gravitacionais.

2.1.3 Funcionalidades Extra

No projecto foram implementadas, como é observável no fluxograma (ver em Anexo), três funcionalidades extras. Estas novas implemetações são as seguintes:

- ***Restart:*** quando o jogo acaba, quer pela colisão em qualquer obstáculo, quer pela ultrapassagem dos limites da Janela de Texto, o jogador tem a possibilidade de reiniciar o programa se pressionar o botões de pressão I1;
- ***Stop:*** durante a a execução do programa, o jogador é livre de colocar o jogo em modo *Pausa*, pressionando o botões de pressão I3, e sair do mesmo pressionando o botões de pressão I0;
- ***GravityShift:*** quando I1 é premida, o efeito gravitacional fica com o valor simétrico do anterior, permitindo ao jogador saltar de forma inversa.
- **Número de obstáculos:** quando o jogo acaba, será possível ver no LCD, não só a distância, mas como o *record*.

3 Conclusão

Neste trabalho efectou-se a simlulação de um programa em Assembly, bem como o seu projecto e implementação na placa de *hardware* do processador P3.

Na primeira parte, a simulação foi feita com o *software* P3 e observou-se um funcionamento previsto do programa. Para evitar a disputa de recursos, foi necessário recorrer à passagem de parâmetros para a pilha de modo a salvaguardar valores entre rotinas. Para uma melhor compreensão do programa, foi criada várias rotinas auxiliares à rotina principal.

Na segunda parte, concebeu-se o programa testado na placa *hardware* P3 na qual verificámos alterações no espaço do jogo: no desenho de texto *hardware*, caracteres "I" apareciam ao longo do programa. Estas anomalias são justificadas pelo desenho da placa e na discrepância entre o simulador e a versão *hardware* do P3.

Este projecto foi realizado seguindo o enunciado com maior rigor, e, não divergindo do mesmo. Não só foram implementadas as funcionalidades básicas obrigatórias como também as rotinas que permitem o funcionamento das funções de *Restart, Stop* e *Inverter*.

Dado que a placa apresenta um funcionamento diferente do simulador, foi necessário alterar código mostrando ser a principal dificuldade durante a codificação. Posteriormente o programa foi testado em *hardware* verificando-se que o seu funcionamento correspondeu ao planeado.

4 Anexo

4.1 Manual do Jogo

- O nível mais básico é o nível 1, com o pássaro centrado no primeiro terço da pista;
- Os obstáculos (colunas) surgem na coluna mais à direita da Janela de Texto e o tamanho delas é gerados aleatóriamente;
- A velocidade de cada obstáculo é constante mas dependente da velocidade do pássaro;
- A velocidade do pássaro depende dos nível em que o jogador se encontra;
- Cada nível pertence a um Super Nível. Cada Super Nível é um conjunto de 4 níveis que partilha características comuns, como a velocidade e o espaçamento entre colunas. A velocidade vai gradualmente aumentando em cada Super Nível e o espaçamento entre colunas vai diminuindo dentro de cada Super Nível. A tabela1 descreve cada um destes aspetos;

Tabela 1: Descrição de Níveis

	Super Níveis															
	Super Nível 1				Super Nível 2				Super Nível 3				Super Nível 4			
Níveis	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Velocidade[1]	2m/s				3m/s				4m/s				5m/s			
Espaçamento entre colunas[2]	8	7	6	5	8	7	6	5	8	7	6	5	8	7	6	5

- O display de 7 segmentos mostra o número de obstáculos passados;
- O LCD mostra a distância atual percorrida;
- Os LEDs aceso correspondem ao nível em que o jogador se encontra.

4.2 Manual de Teclas

Durante as várias fase de jogo: a fase *Inicialização* – correspondente à fase depois da escrita da mensagem de Início de Jogo e antes da fase *Jogo* –, a fase *Jogo* e fase *Pausa*; o jogador pode optar pelos seguintes comandos:

Tabela 2: Manual de Teclas

	Fases de Jogo		
Botões de pressão	**Inicialização**	**Jogo**	**Pausa**
I0	-	Pássaro salta	Fase Jogo
I1	Começa o Jogo	Decrementa nível	-
I2	-	Incrementa nível	-
I3	-	Fase pausa	-

[1] O aumento da velocidade é proporcional ao número de ciclos executados. Por exemplo, se a velocidade depender de dois ciclos, como cada ciclo demora 100ms a executar-se (tempo de temporizador), 200ms são demorados por metro (coluna), que corresponde a 5m/s.

[2] As unidades do espaçamento entre colunas é número de colunas.

4.3 Mapa de endereçamento

Tabela 3: Mapa de endereçamento

Tipo	Endereço Inicial	Endereço Final	Total de Posições
Dados	0000h	—	X
Declarações de variáveis	8000h	—	Y
Tabela de interrupções	FE00h	FE0Fh	16
Portos para dispositivos de E/S	FFF0h	FFFFh	16
Pilha	FDFFh	—	Z