U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Relatório de Projeto
JUMP!

Ricardo Neves (up201405868)
Sofia Alves (up201504570)

TURMA 6 – GRUPO 12

Relatório de Projeto realizado no âmbito do
Mestrado Integrado em Engenharia Informática e Computação
2º Ano – 2016/2017

Índice

1. JUMP!

1.1. Breve Descrição

O jogo consiste em evitar obstáculos que vão surgindo ao longo do mapa, a colisão com um destes obstáculos implica o fim do jogo. O objetivo é evitar a maior quantidade de obstáculos a fim de permanecer em jogo a maior quantidade de tempo possível, uma vez que a pontuação final depende do tempo de jogo.

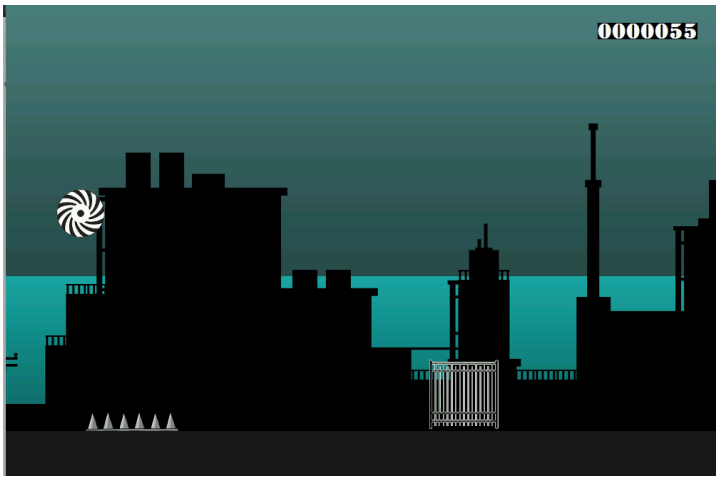
1.2. Instruções de Utilização

MENU PRINCIPAL

No menu principal o jogador tem 2 alternativas. Jogar, movendo o rato até à respetiva opção e selecionada com o botão esquerdo do mesmo ou sair do jogo usando o mesmo processo para selecionar a respetiva opção. Também é possível sair do jogo utilizando a tecla ESC.

MODO DE JOGO

No decorrer do jogo, a fim de evitar os obstáculos o jogador deverá utilizar a seta para cima ou a barra de espaço gerando um salto. O jogador poderá voltar ao menu principal a qualquer momento pressionando a tecla ESC. A pontuação encontra-se no campo superior direito e é atualizada no decorrer do jogo.



FIM DO JOGO

Quando o jogador não consegue evitar um obstáculo, perde o jogo. Tem agora duas opções que podem ser selecionadas através do rato, voltar a jogar ou sair do jogo. Também pode sair do jogo pressionando a tecla ESC.

2. Estado do Projeto

Dispositivo	Utilidade	Interrupções
Timer	Controlo da frame rate e geração de obstáculos	Sim
Teclado	Controlo da bola e interação com os menus	Sim
Rato	Navegação nos menus	Sim
RTC	Contagem do tempo de jogo	Sim
Placa Gráfica	Visualização dos menus, jogo e imagens	Não

A tabela acima apresenta resumidamente a funcionalidades usadas e implementação dos diferentes dispositivos de entrada e saída. Informações mais detalhadas acerca de cada dispositivo abaixo.

2.1. Timer 0

As interrupções do Timer, geradas a uma frequência de 60 Hz, permitem controlar a frame rate e gerir as animações. Na função "updateGame()" presente no modulo *Game* é realizado o processamento das interrupções do rato, o qual consiste no incremento de um contador e posterior atualização do buffer secundário.

2.2. Teclado

O teclado tem como função principal controlar a bola, fazendo com que ela salte quando determinadas teclas são pressionadas. Este é também utilizando como um atalho para sair de qualquer um dos 3 estados (tecla ESC). Estas implementações encontram-se nos módulos Menu, Play e Over nas funções de atualização respetivas (ex: updatePlay()).

2.3. Rato

Tanto a posição como os botões do rato são usados a fim de selecionar opções tanto no módulo Menu como no módulo Over. A sua implementação encontra-se nas funções de atualização (ex: updateMenu()) de ambos os módulos referenciados acima, mas sobretudo no módulo Mouse.

2.4. RTC

O RTC (Real Time Clock) é utilizado para obter a duração do jogo, desde o instante inicial até ao instante atual. A pontuação, quer ao longo do jogo, quer no fim do jogo é obtida recorredo ao tempo de jogo. Implementação disponível no módulo RTC e Game.

2.5. Placa Gráfica

A placa gráfica é usada para apresentar a informação gráfica do jogo no modo gráfico 0x117. A fim de evitar que problemas como flickering é usado um double buffer. Isto é, cria-se um buffer adicional que é copiado para o frame buffer apenas quando toda a informação nele for escrita. A única função VBE utilizada é vbe_get_mode_info(). A implementação das funcionalidades deste dispositivo encontra-se nos módulos Video Card e Bitmap.

3. Organização do Código

O código foi estruturado segundo módulos que se interligam entre si. Nesta secção será feita uma breve descrição do código de cada um dos módulos e serão descritas as principais estruturas de dados de cada módulo.

3.1. Main

Inclui o ficheiro “main.c”, este módulo pode ser considerado a “raiz” do jogo, é a partir dele que começa a execução do mesmo. Cria-se um objeto do tipo Game que será atualizado a cada iteração de um ciclo responsável também pelos movimentos do rato no ecrã (ou ausência deles), este ciclo termina apenas quando o Game criado chega ou fim.

Peso no Projeto: 5%

3.2. Game

Inclui os ficheiros “game.c” e “game.h”. Toda a gestão dos estados é feita neste módulo (atualização, eliminação, mudança de estado), para além disso o módulo é também responsável pela inicialização do modo gráfico e atualização da informação exibida no ecrã. As funções deste módulos são desempenhadas baseando-se nos dados fornecidos pelas interrupções dos diferentes dispositivos na função updateGame().

A estrutura Game é composta pelas posições na máscara de interrupção dos dispositivos usados, uma flag que indica se o jogo terminou, o último scancode gerado pelo teclado, um counter atualizado a cada interrupção do timer, a duração do jogo, informação relativa à manipulação do estado atual, e dois apontadores para objetos do tipo Time, sendo que o primeiro corresponde ao tempo de início do jogo e o segundo corresponde ao tempo atual.

Peso no Projeto: 15%

3.3. Menu

Inclui os ficheiros “menu.h” e “menu.c”. Neste módulo estão implementadas as funções que permitem a gestão do Menu, funções estas

responsáveis pela sua criação, atualização, eliminação e exibição no ecrã. A função responsável pela atualização verifica, a cada chamada, se o rato se encontra nalguma das opções do Menu ou se a tecla ESC foi pressionada, desencadeando, se necessário, a resposta correspondente.

A estrutura Menu é composta pela imagem de fundo, pelos botões de "play" e "exit" exibidos no ecrã e por uma flag que indica se haverá mudança de estado.

Peso no Projeto: 5%

3.4. Play

Inclui os ficheiros "paly.h" e "play.c". Neste módulo estão implementadas funções que permitem a gestão de um jogo, funções estas responsáveis pela sua criação, atualização, eliminação e exibição no ecrã. A função de atualização do jogo é responsável por:

- salto da bola em resposta a certas teclas serem pressionadas
- geração dos obstáculos e seu movimento
- movimento da imagem de funo
- detetar colisões entre a bola e os diferentes obstáculos
- atualizar a pontuação do jogador
- gerar um explosão no caso de colisão

A estrutura Play é composta pelas imagens necessárias à construção do fundo, uma flag que indica se o estado terminou, apontadores para objetos do tipo Obstacle, Ball e Explsoion, uma flag que indica se está a decorrer um explosão, uma flag que indica se houve colisão e um contador com a pontuação até ao momento.

Peso no Projeto: 15%

3.5. Over

Inclui os ficheiros "over.h" e "over.c". Neste módulo estão implementadas as funções que permitem a gestão do estado Game Over, funções estas responsáveis pela sua criação, atualização, eliminação e exibição no ecrã. A função responsável pela atualização verifica, a cada chamada, se o rato se encontra nalguma das opções exibidas no ecrã ou se a tecla ESC foi pressionada, desencadeando, se necessário, a resposta correspondente. Este módulo é muito semelhante ao módulo Menu (3.3) deferindo dele apenas por mostrar no ecrã a pontuação final do jogo.

A estrutura GameOver é composta pela imagem de fundo, pelos botões de "yes" e "no" exibidos no ecrã, por uma flag que indica se o estado terminou e pela pontuação obtida no jogo mais recente.

Peso no Projeto: 5%

3.6. Ball

Inclui ficheiros “ball.h” e “ball.c”. Este módulo contém as funções que permitem a gestão de objetos do tipo Ball, isto é a sua criação, atualização, eliminação e exibição no ecrã. A função responsável pela atualização muda a imagem que representa a bola a cada chamada criando o efeito de rotação. O efeito de salto é gerado pela função que está encarregue de desenhar a bola no ecrã.

A estrutura Ball é composta por um array de apontadores para as imagens que criam a animação, pelas coordenadas da posição no ecrã, por uma flag que indica se a bola está na a cair e por um índice que indica a posição atual no array de apontadores.

Peso no Projeto: 5%

3.7. Obstacle

Inclui os ficheiros “obstacle.h” e “obstacle.c”. Este módulo contém as funções que permitem a gestão de objetos do tipo Obstacle. A função responsável pela atualização de um objeto deste tipo é aquela que, alterando ligeiramente, a cada chamada, as coordenadas da imagem do objeto no ecrã, cria a animação dos obstáculos. Para além disso esta função encarrega-se de atualizar as coordenadas de colisão.

A estrutura Obstacle é composta por uma flag que indica se o obstáculo deve ou não ser desenhado no ecrã, um apontador para a imagem correspondente ao obstáculo pretendido, as coordenadas dessa imagem no ecrã e as coordenadas de colisão.

Peso no Projeto: 5%

3.8. Explosion

Inclui os ficheiros “explosion.h” e “explosion.c”. Este módulo contém as funções que permitem a gestão de objetos do tipo Explosion. A função responsável pela atualização de um objeto deste tipo muda a imagem que irá posteriormente ser desenhada no ecrã criando assim um efeito de explosão realista.

A estrutura Explosion é composta por um array de apontadores para as imagens que quando percorrido vai criar a animação, um índice que indica a posição atual no array de apontadores e por um flag que indica se já todas as diferentes imagens foram mostradas no ecrã (criando o efeito de uma explosão).

Peso no Projeto: 5%

3.9. Bitmap

Inclui os ficheiros “bitmap.h” e “bitmap.c”. Este módulo contém funções responsáveis pela gestão de Bitmaps, pelo seu carregamento, pela escrita da sua informação no buffer secundário e pela posterior libertação dos recursos usados.

Ao código inicialmente retirado do blog do Henrique Ferrolho, foram feitas algumas alterações. Retirou-se certas funcionalidades que eram inúteis para o projeto e acrescentou-se um algoritmo que permite considerar uma cor como transparente. Para além disso foram feitas pequenas alterações a fim de adaptar o código ao projeto.

URL: http://difusal.blogspot.pt/2014/09/minixtutorial-8-loading-bmp-images.html

Peso no Projeto: 10%

3.10. Video Card

Inclui os ficheiros “vídeo_gr.h” e “vídeo_gr.c”. Módulo responsável ela exibição do jogo e dos seus diferentes estados no ecrã. Composto por funções que permitem a entrada no modo gráfico, o mapeamento da VRAM, o retorno ao modo de texto do Minix e a cópia dos conteúdos do buffer secundário para o frame buffer.

Peso no Projeto: 5%

3.11. VBE

Inclui ficheiros “vbe.h” e “vbe.c”. Módulo que contém a função usada para obter informação acerca do atual modo de vídeo, tal como o endereço do frame buffer.

Peso no Projeto: 2.5%

3.12. Mouse

Inclui ficheiros “mouse.h” e mouse.c”. Este módulo contém funções que permitem ativar e subscrever as interrupções do rato, assim como as funções que são responsáveis por lidar com essas interrupções que leem os packets enviados pelo rato e atualizam a seu estado (posição, botões, etc)

quando necessário. Estas funções entre outras como criar, desenhar e libertar recursos estão encarregues da gestão do rato.

A estrutura Mouse destina-se a ter apenas uma instanciação, esta é composta por pelas coordenadas do rato, por um array de 3 elementos que correspondem aos 3 bytes de um packet, uma flag que indica se o packet está completo, 2 flags para informações acerca do botão esquerdo e um flag que indica se o rato se destina a ser ou não desenhado no ecrã.

Peso no Projeto: 5%

3.13. Timer

Inclui os ficheiros “timer.h” e “timer.c”. Este módulo contém funções que permitem subscrever as interrupções do timer, assim como aquelas que são responsáveis por fazer unsubscribe e lidar com as interrupções do timer, incrementando um counter.

Peso no Projeto: 5%

3.14. Keyboard

Inclui os ficheiros “keyboard.h” e “keyboard.c”. Este módulo contém funções que permitem subscrever as interrupções do teclado, assim como aquelas que são responsáveis por fazer unsubscribe e lidar com as interrupções lendo o scancode que se encontra no output buffer do teclado. Também está definida neste módulo a função que permite limpar o output buffer no final da execução.

Peso no Projeto: 5%

3.15. RTC

Inclui os ficheiros “rtc.h” e “rtc.c”. Este módulo contém o código que realiza a configuração, o subscribe e o unsubscribe do RTC. Para além disso é também nele que estão as funções de gestão dos objetos do tipo Time, nomeadamente a sua criação, atualização e libertação dos recursos.

A estrutura Time é composta por 3 valores que representam, os segundos, minutos e a hora atual. A função de atualização desta estrutura é também aquela responsável por lidar com interrupções que atualiza os membros dados da estrutura Time com os valores da hora atual. O módulo

inclui ainda uma função para calcular o tempo entre dois objetos do tipo Time diferentes.

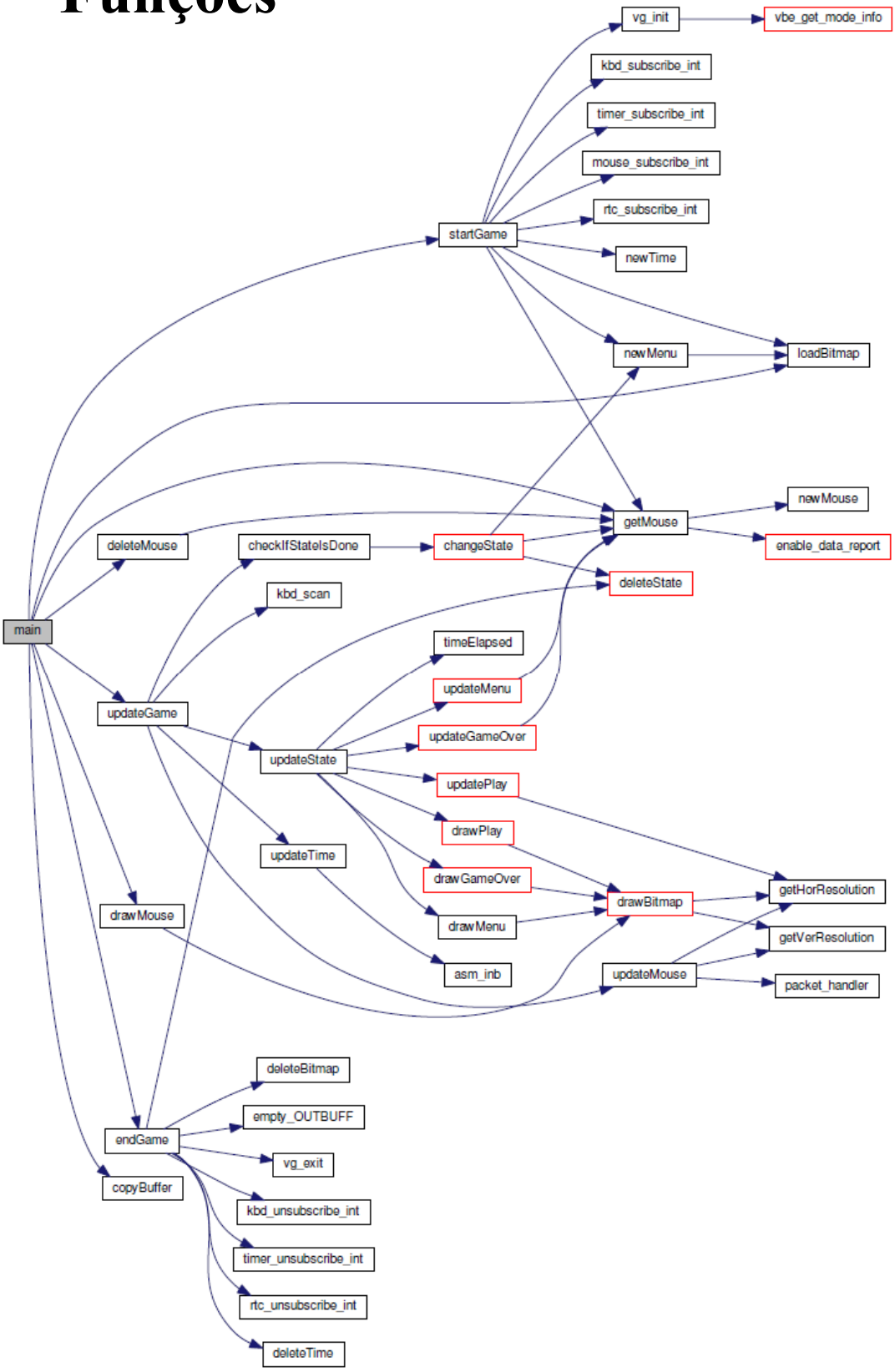
Peso no Projeto: 5%

3.16. RTC Assembly

Este módulo apenas inlcui o ficheiro “RTC_asm.S”, cuja função “asm_inb()” permite ler o conteúdo do output buffer, conteúdo este que ficará guardado numa variável global provada do módulo RTC.

Peso no Projeto: 2.5%

4. Diagrama de Chamada de Funções



5. Detalhes da Implementação

O jogo foi implementando usando uma máquina de estados. Existem 3 estados diferentes, Menu, Play e GameOver. Cada jogo pode estar apenas num estado de cada vez, a cada interrupção do timer o irá atualizar o estado em que se encontra e caso o utilizador tenha interagido com os dispositivos de forma a levar à transição de estado, os recursos ocupados pelo estado atual são libertados e efetua-se uma transição de estado.

No desenvolvimento do código recorreu-se a programação orientada a objetos, uma vez que em C não é possível criar classes para auxiliar este tipo de programação recorreu-se a structs. Cada struct representa uma classe com membros e criaram-se funções que permitem manipular esses membros.

A fim de evitar problemas nas animações exibidas no ecrã recorreu-se a uma técnica designada double buffering. Em vez de se escrever diretamente na memória vídeo, recorre-se a um segundo buffer para o qual é copiada toda a informação e só quando este processo é concluído o conteúdo do segundo buffer será copiado para a memória vídeo.

A implementação usada para a manipulação de bitmaps foi baseada na manipulação implementada pelo Henrique Ferrolho que se encontra disponível no seu blog. Foram feitas alterações a fim de remover funcionalidades que não eram uteis para o projeto e acrescentar outras que eram essenciais, tais como o uso de double buffering a criação de "transparência". A transparência das imagens é possível impedindo que uma determinada cor previamente escolhida seja desenhada.

O RTC foi implementado ativando as update interrupts, o que faz com que este dispositivo gere interrupções cada vez que acaba de atualizar os dados referentes à data, reduzindo assim a probabilidade de erro nos dados lidos deste dispositivo.

No projeto foram implementas funções de assembly que permitem a leitura e escrita em endereços de memória específicos.

6. Considerações Finais

ASPETOS A SALIENTAR

- Os prazos de avaliação prejudicam as turmas com aulas nos primeiros dias da semana, uma vez que grande parte das frequências são marcadas nesses dias e as restantes turmas podem depois tirar dúvidas com aqueles que tiveram de entregar nos primeiros dias
- O trabalho que a unidade curricular requer é, na nossa opinião, superior aos créditos que lhe são atribuídos
- Os dispositivos que não foram abordados nas aulas práticas deveriam ter menos impacto na nota final do projeto

7. Instruções de Instalação

A instalação e execução do jogo deve ser feitas com acesso root ao sistema, para que tal seja possível utilizar o comando "su". Assumindo que se encontra na pasta "proj" basta seguir a instruções abaixo indicadas.

Para instalar o jogo deve usar os seguintes scripts:

- sh install.sh
- sh compile.sh

Para executar o jogo deve usar o seguinte script:

- sh run.sh