



Mestrado Integrado em Engenharia Informática e Computação

05 de janeiro de 2020

Projeto Laboratório de Computadores

Turma 8 | Grupo 11

Composição do grupo:

Francisco José Paiva Gonçalves (up201704790)
Miguel Delgado Pinto (up201706156)

Indice

- Manual do Utilizador

-	Menu	2
-	Start Game (Single Player)	3
-	Multiplayer	4
-	Highscores	5
-	Rules	6

- Estado do projeto

-	Placa Gráfica	7
-	Teclado	8
-	Rato	8
-	Timer	8
-	RTC	8
-	Porta série	9

- Estrutura do código

-	Asteroid	10
-	Highscore	10
-	Keyboard	10
-	KBC	10
-	Mouse	10
-	Proj	11
-	RTC	11
-	Serial Port	11
-	Space Lander	11
-	Spacecraft	12
-	Sprite	12
-	Timer	12
-	UART Protocol	13
-	Utils	13
-	Video Card	13

- Function call graph (Doxygen)

14

- Detalhes da implementação

15

- Detalhes da implementação

16

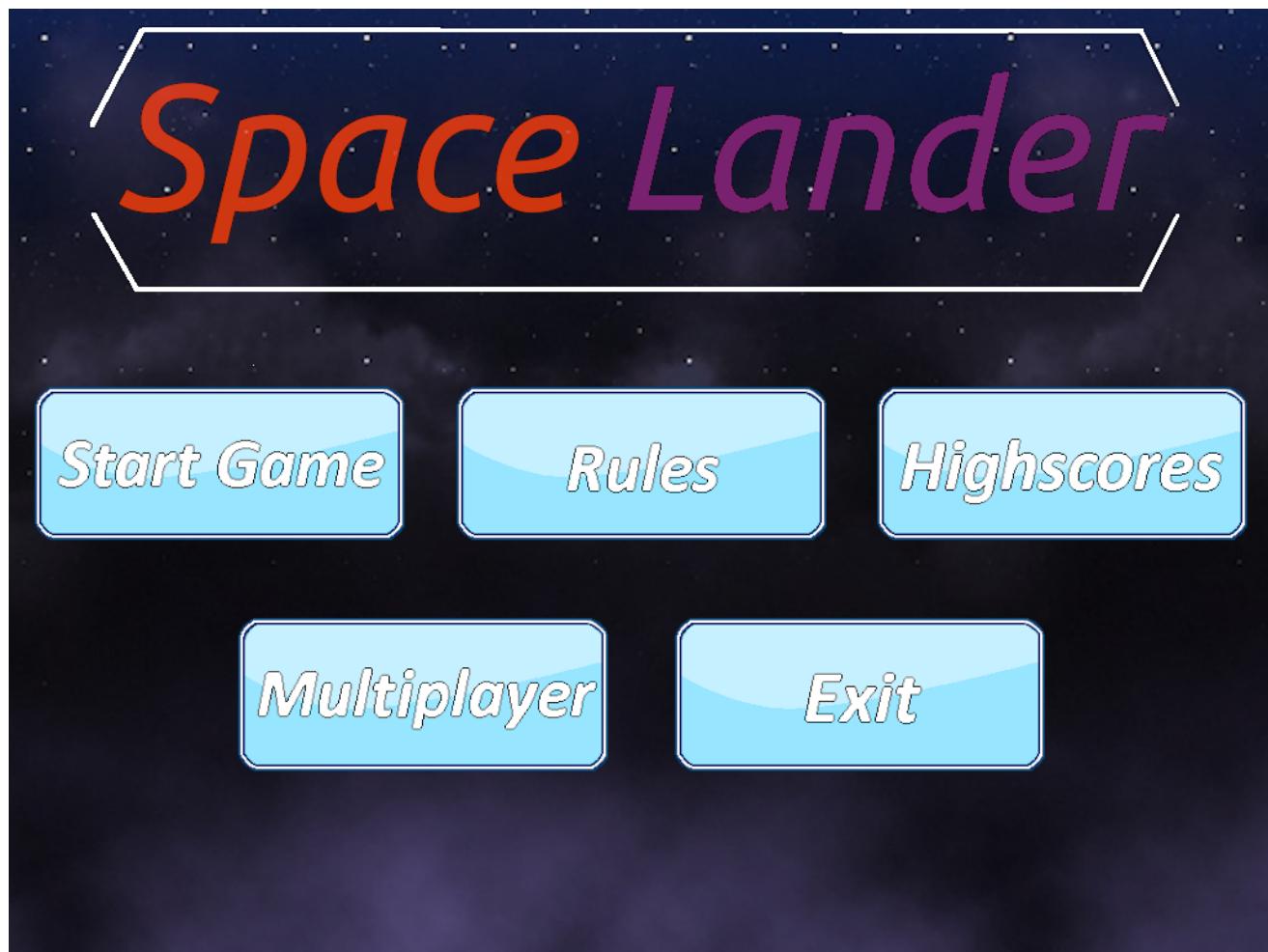
Manual do Utilizador

Menu

Após compilar e correr o programa, o utilizador será enviado para o menu principal. Neste momento tem 5 opções:

- **Start Game**, que inicia o jogo para apenas 1 jogador;
- **Multiplayer**, que inicia o jogo para 2 jogadores (usando Serial Port);
- **Rules**, que apresenta um *sprite* com as instruções, regras e objetivo do jogo;
- **Highscores**, que utiliza *fonts* para apresentar os melhores resultados para cada nível do jogo. Note-se que um resultado é o tempo, em segundos, que um nível demora a ser completado;
- **Exit**, que termina o programa, voltando a apresentar o terminal do Minix.

Cada opção deve ser selecionada com o rato, clicando no botão do lado esquerdo na área delimitada por cada botão.



Start Game

Nesta opção, o utilizador será levado para outro menu, no qual poderá escolher entre 3 níveis diferentes. O primeiro nível situa-se na Lua, o segundo nível situa-se em Marte e terceiro nível situa-se no *The End*, inspirado no famoso videojogo “Minecraft”. O *sprite* de seleção de níveis é o seguinte:



Após selecionar um nível, o jogo entra no modo de jogo. A sua nave começa numa ponta do planeta e o objetivo é chegar à zona de aterragem, desviando-se dos asteróides que caem constantemente do céu. Estes asteróides surgem periodicamente no espaço de jogo. Tem ao seu dispôr um lança-foguetes, que permite destruir os asteróides, apontando com o rato e clicando na direção que pretende disparar.

Para controlar a nave, as teclas ‘A’ e ‘D’ rodam a nave, respetivamente, no sentido contrário aos ponteiros do relógio e no sentido dos ponteiros do relógio. Para impulsionar a nave, a tecla ‘Espaço’ deve ser premida. A nave encontra-se sempre sob o efeito da gravidade do planeta. Cabe ao jogador impedir que se despenha.

O jogo termina, com sucesso, quando o jogador aterrará na zona destinada, que está sempre colorida a verde. Se o jogador se despenhar ou for atingido por um asteróide, o jogo termina também, mas sem sucesso.

A aparência dos três níveis pode ver vistas nas imagens seguintes:



Multiplayer

Esta opção é semelhante à anterior, mas permite jogar juntamente com outro utilizador. Neste modo, os dois jogadores começam ao mesmo tempo, no mesmo nível, jogando um contra o outro. O objetivo é ser o primeiro a aterrinar na zona pretendida. Se um jogador despenhar a sua nave ou for atingido por um asteróide, então o jogador vence.

Ao selecionar esta opção, deverá escolher o nível, de um modo igual ao *Start Game*. Se o outro jogador não tiver selecionado também a opção Multiplayer, fica numa a “sala de espera” a aguardar que tal aconteça:

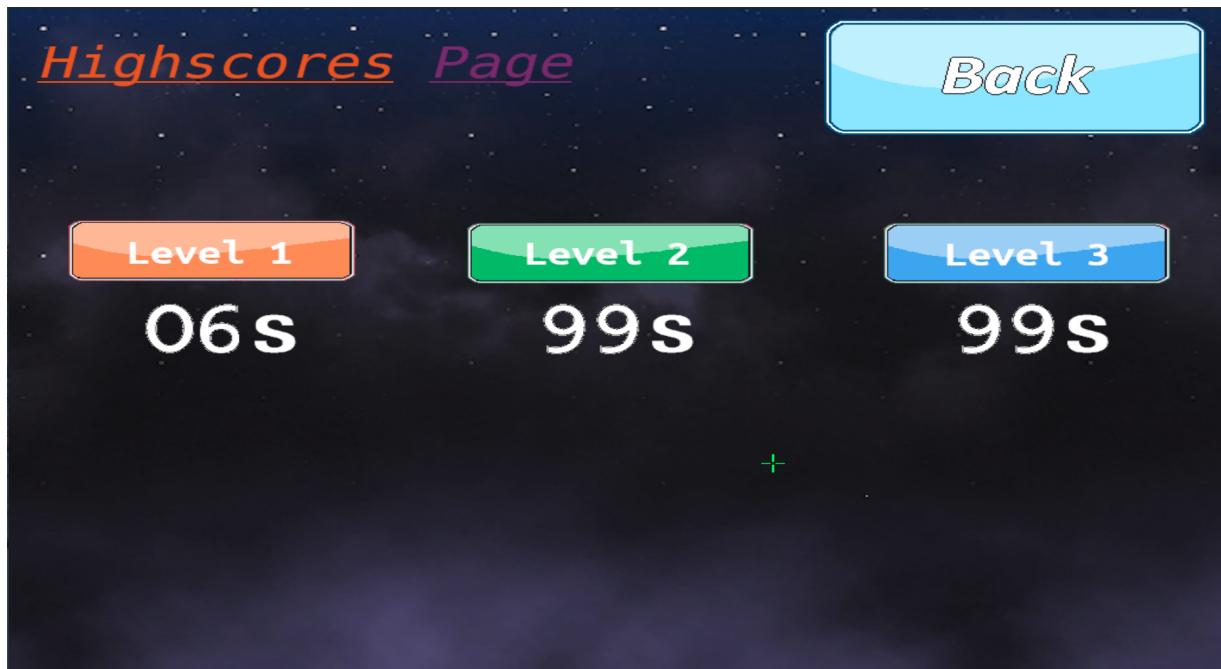


Após terminar um nível, tanto no modo *Singleplayer* como no modo *Multiplayer* é imprimida uma sprite de sucesso ou insucesso no ecrã. Neste ecrã, carregando na tecla ‘Esc’, o jogador volta ao menu principal. A imagem à esquerda é mostrada quando o jogador perde um nível e da direita quando o jogador vence:



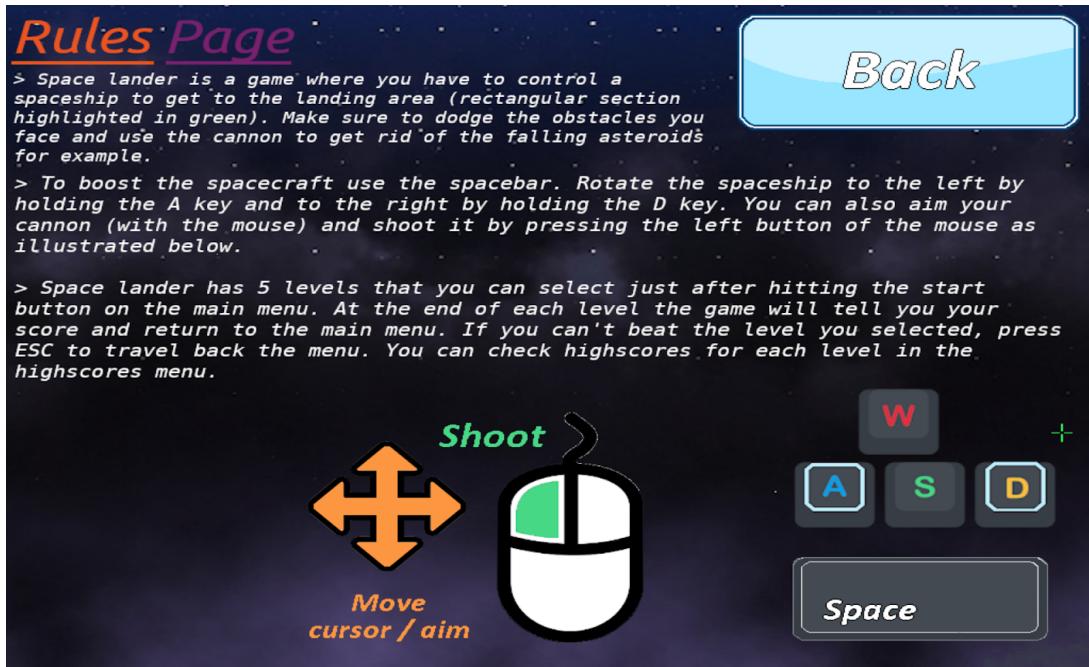
Highscores

Nesta opção, é imprimida no ecrã a informação relativamente aos melhores tempos de finalização de cada nível, em segundos. Esta é informação é atualizada cada vez que um jogador termina, com sucesso, um nível, passando a haver um novo highscore no caso do score ser inferior (menor tempo) ao anterior.



Rules

Nesta opção, é imprimida no ecrã informação relativamente aos controlos e funcionamento do jogo, com imagens ilustrativas:



Estado do projeto

Um resumo dos dispositivos utilizados e das suas funcionalidades encontra-se na seguinte tabela. Nas subsecções seguintes, estes serão abordados em maior detalhe.

Dispositivo	Propósito	Interrupts
Timer	Controlar a <i>frame rate</i> , determinar <i>highscores</i> , atualizar <i>sprites</i> e dar origem a asteróides	S
Placa gráfica	Desenhar <i>sprites</i> , com rotação e sem rotação	N
Teclado	Controlar o movimento da nave: impulsão e rotação	S
Rato	Controlar os mísseis da nave e seleção de opções nos menus	S
RTC	Seleção de modo diurno ou modo noturno para o fundo de um nível	N
Porta Série	Modo de corrida 1vs1	S

Placa gráfica

A placa gráfica é utilizada para visualizar o jogo e todos os seus componentes, bem como os menus. É responsável pelas colisões de *sprites*, tanto por cor como por *hitboxes*, e pelo *display* destas no ecrã, tanto com rotação (muito útil para a nave), como sem rotação. Implementou-se também *double buffering*, de modo a que a visualização de *sprites* fosse mais suave. Finalmente, utilizaram-se *fonts* para o *display* dos *highscores*.

Utilizou-se um modo de cor direto, 0x118, com uma resolução de 1024x768 e com certa de 16.8 milhões de cores possíveis, dado que utiliza RGBA 8:8:8.

Teclado

O teclado é utilizado para controlar o movimento da nave durante os níveis: permite rodar a nave e impulsioná-la pela atmosfera. Permite também voltar atrás para o menu, durante um jogo ou no final do jogo, carregando no 'Esc'.

Rato

O rato é utilizado para, durante o jogo, controlar a direção dos mísseis. Carregar no lado esquerdo do rato numa zona do ecrã faz com que o míssel siga a trajetória da nave até esse ponto. O movimento do rato permite escolher a zona do ecrã. Está sempre acompanhado de *crosshairs*, tanto no menu como nos níveis. Permite também selecionar as opções de navegação de menus, com cliques no lado esquerdo.

Timer

O timer é um dispositivo de grande importância para o projeto, dada o grande número de funções que desempenha. Tem como funções principais controlar a *frame rate* do jogo, atualizar as *sprites* quando se movimentam ou desaparecem e, periodicamente, dar origem aos asteróides. É também importante na determinação dos *highscores* de um jogador (tempo que demora, em segundos, a terminar um nível). Para além disso, é utilizado para limitar o número de mísseis que são possíveis disparar, ao longo de um período de tempo.

RTC

O RTC (*Real-Time Clock*) é utilizado para selecionar as imagens de fundo de cada nível, dependendo da altura do dia em que estes são jogados. Se for jogado entre as 07:00h e as 19:00h, então os planos de fundos apresentam cores claras. Caso contrário, apresentam cores escuras.

Porta Série

A porta série é utilizada no modo multijogador, 1 contra 1, para ser possível fazer “corridas” entre jogadores. Primeiramente, é enviada uma mensagem de conexão inicial, por parte de ambos os utilizadores. Uma vez conectados, o jogo inicia ao mesmo tempo. Quando um jogador colide com um asteróide/chão do planeta ou aterra com sucesso, uma mensagem é enviada para o outro jogador, que termina o jogo e determina o vencedor.

Estrutura do Código

asteroid

Este módulo é responsável pela criação, atualização e destruição de asteroides. Serve como uma abordagem de programação orientada a objetos, para este componente.

Peso para o projeto: 3%

Peso dos participantes: Francisco Gonçalves (60%) e Miguel Pinto (40%)

highscore

Este módulo é responsável pela gestão dos highscores: a sua atualização, carregamento e *display*.

Peso para o projeto: 3%

Peso dos participantes: Francisco Gonçalves (100%)

kbc

Este módulo é responsável por interagir com o controlador KBC, utilizado tanto no rato como no teclado, tendo sido importado do LAB4.

Peso para o projeto: 5%

Peso dos participantes: Francisco Gonçalves (40%) e Miguel Pinto (60%)

keyboard

Este módulo é responsável por garantir a interação do programa com o teclado, tendo sido importado do LAB3.

Peso para o projeto: 8%

Peso dos participantes: Francisco Gonçalves (40%) e Miguel Pinto (60%)

mouse

Este módulo é responsável por garantir a interação do programa com o rato, tendo sido importado do LAB4.

Peso para o projeto: 8%

Peso dos participantes: Francisco Gonçalves (50%) e Miguel Pinto (50%)

Proj

Neste módulo encontra-se a função fornecida pelos professores, *proj_main_loop*, responsável por subscrever a interrupções dos dispositivos que as utilizam, iniciar o modo de vídeo. No final do projeto, é responsável por cancelar as subscrições e libertar a memória alocada pelos dispositivos.

Peso para o projeto: 5%

Peso dos participantes: Francisco Gonçalves (25%) e Miguel Pinto (75%)

rtc

Neste módulo encontra-se assente o funcionamento do RTC (*Real-Time Clock*), com todas as funções necessárias para extrair a data e as horas atuais.

Peso para o projeto: 2%

Peso dos participantes: Francisco Gonçalves (100%)

serial_port

Neste módulo encontram-se definidas as funções necessárias para configurar e transmitir informação utilizando a porta série. Note-se que não contém funções específicas para o jogo, apenas constitui uma API para a utilização da porta série.

Peso para o projeto: 3%

Peso dos participantes: Miguel Pinto (100%)

space_lander

Este é o módulo de maior importância do projeto. É responsável pela gestão de todos os eventos gerados por qualquer dispositivo. Possui também a máquina de estados jogos, que é alterada de acordo com os eventos. Possui também as chamadas de todos os *interrupt handlers* dos dispositivos. É responsável pela inicialização, alocando a memória necessária para as suas componentes, e pela sua destruição, desalocando a memória usada, de modo a garantir a integridade dos espaço de memória.

Peso para o projeto: 20%

Peso dos participantes: Francisco Gonçalves (35%) e Miguel Pinto (75%)

spacecraft

O módulo *spacecraft* é também de grande relevância. Trata da gestão da nave espacial, atualizando-a e simulando a ação da gravidade, a propulsão do seu motor, o atrito face ao deslocamento e a sua rotação. Também contém funções para a sua visualização e para a apagar do ecrã, de modo eficiente (apagando apenas a zona onde foi desenhada).

Peso para o projeto: 12%

Peso dos participantes: Francisco Gonçalves (30%) e Miguel Pinto (70%)

sprite

Trata-se do módulo principal para a gestão de todas as sprites e tudo o que é imprimido no ecrã. Possui métodos de *display* de sprites com rotação, o que se revelou um grande desafio, no entanto o grupo conseguiu superá-lo com sucesso. Também contém métodos de *display* de sprites com rotação, métodos para atualizar a posições das *sprites* dependendo das suas velocidades e métodos para “apagar” *sprites* do ecrã, deixando as suas posições preenchidas pela zona do *background* correspondente (lida também com rotações, se necessário). Por fim, mas não menos importante, é responsável pela deteção de colisões, por dois métodos diferentes: colisão por *hitboxes* e colisão por cor (contemplando, também, rotações, se necessário).

Peso para o projeto: 12%

Peso dos participantes: Francisco Gonçalves (20%) e Miguel Pinto (80%)

timer

Este módulo é responsável por garantir a interação do programa com o timer, tendo sido importado do LAB2.

Peso para o projeto: 9%

Peso dos participantes: Francisco Gonçalves (50%) e Miguel Pinto (50%)

uart protocol

Este módulo é responsável por definir um protocolo de comunicação por *serial port* específico para o Space Lander. É possível enviar mensagens de conexão, de aviso de destruição da nave e de aviso de aterragem bem sucedida, todas usadas no modo multijogador.

Peso para o projeto: 3%

Peso dos participantes: Miguel Pinto (100%)

utils

Possui funções úteis para o uso de *sys_inb* e *sys_outb* de uma forma segura. Permite também extrair o *MSB* e *LSB* de um número. Importado do LAB2.

Peso para o projeto: 1%

Peso dos participantes: Francisco Gonçalves (50%) e Miguel Pinto (50%)

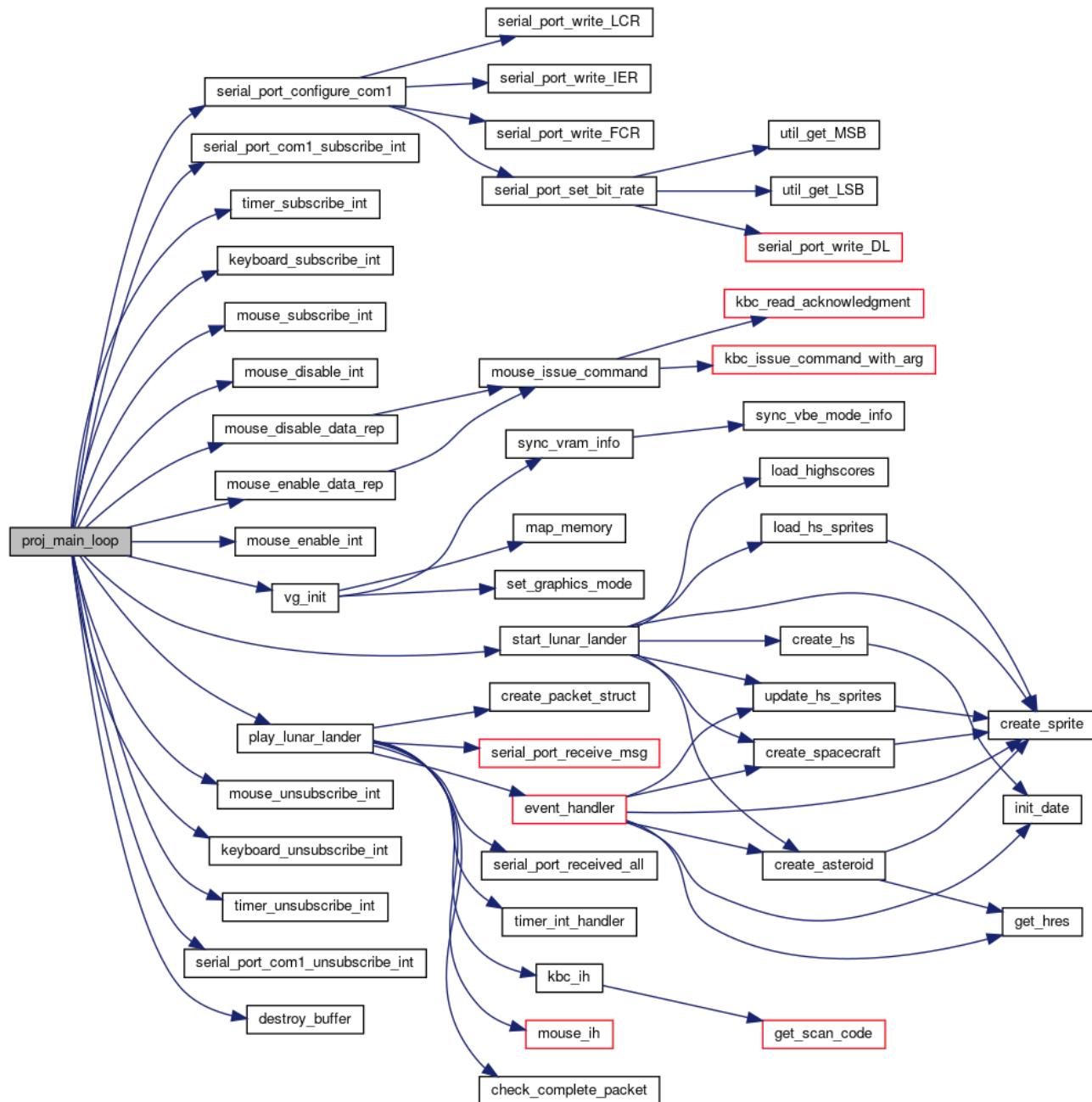
video card

Este módulo é responsável por garantir a interação do programa com o timer, tendo sido importado do LAB5.

Peso para o projeto: 6%

Peso dos participantes: Francisco Gonçalves (30%) e Miguel Pinto (70%)

Function call graph



Detalhes de implementação

De modo a manter o código modular e organizado, decidiu-se seguir uma abordagem orientada a eventos, para gerir toda a informação recebida pelos dispositivos utilizados. Cada *interrupt* gerado por um dispositivo cria um evento, que é posteriormente gerido de acordo com o estado da máquina de estados do jogo (GameStates). É importante de notar, que devido ao número elevado de páginas do menu, também foi implementada uma máquina de estados para este (MenuStates). Para controlar o modo de multijogador, sabendo que jogador é que esperaria pela entrada do outro utilizador, cada jogador possui também um estados (PlayerStates), podendo ser FIRST caso fique à espera de outro jogador, SECOND caso já tenha alguém à espera ou SINGLE, se estiver a jogar por si próprio apenas.

Quanto às entidades principais, decidiu-se modelá-las segundo uma abordagem orientada a objetos. Definiram-se structs tanto para a nave espacial como para os asteróides, permitindo atualizá-los de modo organizado e conciso. Revelou-se uma boa estratégia, devido à simplificação dos mecanismos complexos destes componentes.

Quanto ao RTC, como se trata de um dispositivo simples, este é apenas usado para a alteração do plano de fundo, consoante a hora do dia. Esta interação é feita por *polling*, apenas quando é necessário retirar informação dos seus registo.

Relativamente à porta série, definiu-se não só uma API para a sua interação e configuração, como também um protocolo simples, totalmente orientado ao funcionamento do jogo. Isto permitiu obter um maior nível de abstração e modularidade face a um dispositivo relativamente complexo.

É importante também mencionar o mecanismo de rotação de sprites desenvolvido. Permitiu criar e “limpar” rotações de maneira eficiente, algo que era essencial para o deslocamento da nave. Para o atingir, foram aplicados conhecimentos de Computação Gráfica, como por exemplo: matrizes de rotação e matrizes de translação. Quanto ao mecanismo de colisões, as que funcionam por hitboxes têm em conta as dimensões dos componentes, enquanto que as por cores detectam a presença de cores de componentes em determinadas zonas do plano de fundo, útil para detetar a colisão da nave com o terreno planetário.

Conclusão

Nesta unidade curricular tivemos a oportunidade de aprender muitos conceitos pertinentes relacionados com periféricos e desenvolver capacidade a nível de programação e conceção de algoritmos. Consideramos os conceitos abordados interessantes e essenciais para criar boas bases sobre o conhecimento informático. Pensamos que atingimos os objetivos pretendidos e apesar da exigência elevada desta cadeira, terminamos com maior maturidade e capacidade de resolver problemas propostos.

A disponibilidade dos docentes foi muito útil, especialmente porque os fóruns utilizados no moodle permitem ver dúvidas de outros que podem esclarecer problemas nossos e as respostas foram sempre postadas rapidamente.

No entanto, consideramos que os guiões podiam por vezes ser mais organizados, já que facilitariam a fase inicial de cada lab. Também consideramos que transitar do lab0 para o lab2 pode ser complicado e podia haver um passo intermédio ou um lab0 que desse uma *overview* maior sobre a cadeira.