U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

U.PORTO
FC FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Space Invaders – Relatório de Projeto

Licenciatura em Engenharia Informática e Computação

Laboratório de Computadores, 2022/2023

Turma 5 – Grupo 3

Guilherme de Sousa Ribeiro up202108731@up.pt
Mariana Solange Monteiro Rocha up202004656@up.pt
Pedro Simão Januário Vieira up202108768@up.pt

Índice

1. Instruções de uso 4
   Execução do Projeto 4
   Introdução 4
   Menu Inicial 4
   Jogo 5
      Nave 5
      Vidas 5
      Score 6
      Aliens 6
      Barreiras 7
      Ranking 7
2. Estado do Projeto 8
   Funcionalidades 8
   Uso de dispositivos de I/O 8
      Timer 8
      KBD 9
      Rato 9
      Placa de Vídeo 9
      Real Time Clock (RTC) 9
3. Organização e Estrutura do Código 10
   Timer – 5% 10
   Keyboard – 10% 10
   Mouse – 10% 10
   Vídeo – 10% 10
   RTC – 3% 11
   Utils – 1% 11
   Space Invaders – 20% 11
   Shot – 5% 11
   Shield – 5% 12
   Player – 8% 12
   Alien – 8% 12
   Main – 15% 12
   Conversão de números em BCD para binário 13
   Leitura do Top 10 de pontuações 13

Escrita do Top 10 de pontuações 13
Grafo de Chamada de Funções 14
4. Detalhes da Implementação 15
Placa de Vídeo 15
Real Time Clock (RTC) 15
5. Conclusões 16
Apêndice: Instruções de Instalação 17

1. Instruções de uso

Execução do Projeto

Para executar este projeto é necessário entrar numa máquina virtual com o sistema operativo MINIX-LCOM e efetuar *login*. Depois disso, deve correr os seguintes comandos:

```
>> make clean
>> make depend && make && lcom_run proj
```

Introdução

O nosso projeto inspira-se no icónico videojogo *Space Invaders*. Neste, o jogador assume o controlo de uma nave e o seu papel é defender a Terra de uma invasão alienígena, aniquilando o maior número de aliens que consiga sem ser morto por estes, que também o atacam.

Menu Inicial

O menu inicial é a primeira tela a aparecer ao entrar no programa. Este contém um logótipo do jogo acompanhado por dois botões: *Play* e *Ranking*.
O botão *Play* permite ao jogador ingressar numa partida (pode fazê-lo também premindo a tecla *space bar*), enquanto o botão *Ranking* permite visualizar as melhores pontuações até o momento.
Ambos os botões são clicáveis, mas o jogador poderá também começar um jogo pressionando a tecla *space bar*.
Podemos terminar a aplicação premindo a tecla de *escape*.

Figura 1 - Menu Inicial do Jogo

Jogo

Ao entrar no jogo, o utilizador depara-se com diversos elementos visuais os quais vão ser descritos abaixo.

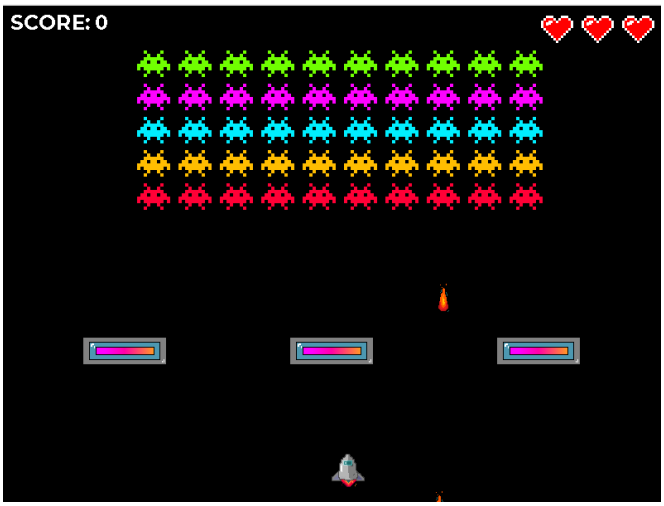

Figura 2 - Aparência do Jogo

Nave

A nave é controlada pelo jogador, o seu movimento é apenas na horizontal e é controlado pelas setas do teclado e pelo movimento horizontal do rato.
Esta pode disparar com as teclas de *space bar*, de seta para cima ou com um clique no botão esquerdo do rato e assim atacar os oponentes – os aliens – aniquilando-os ou protegendo-se contra os ataques destes.
O objetivo do jogador é o de matar o maior número de aliens sem perder o jogo – atingindo uma maior pontuação.

Figura 3 - Nave no seu estado normal (esquerda) e a disparar (à direita)

Vidas

Cada partida começa com três corações, cada um com duas vidas. Cada vez que a nave é atingida por um ataque alienígena perde-se uma vida.

Figura 4 – Representação de seis, cinco e quatro vidas, da esquerda para a direita, respetivamente

Assim que todas as vidas são perdidas, a partida termina e é apresentada uma tela de *Game Over* com um botão *Replay* no qual o jogador poderá escolher começar uma nova partida (também o poderá fazer pressionando a tecla *space bar*) e outro botão que redireciona ao menu, o qual também pode ser acedido pressionando a tecla *escape* nesta tela.

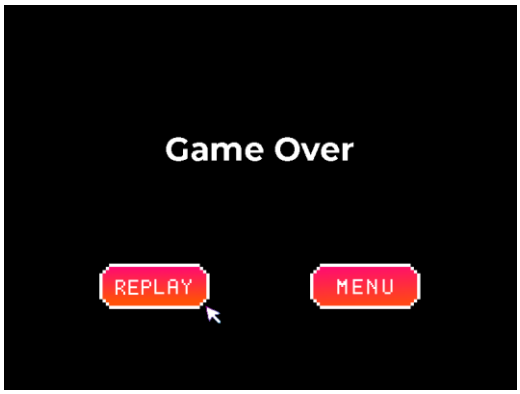

Figura 5 - Tela "Game Over"

Score

O jogador ganha pontos ao atingir os aliens com os seus tiros, matando-os.
Há cinco filas de aliens e a pontuação obtida difere de acordo com a fila em que o alien atingido se encontra. Acertando num que esteja na fila mais abaixo de todas (aliens vermelhos) recebe 1 ponto, ao passo que se acertar em um que se encontre na fila mais acima de todas (aliens verdes) recebe 5 pontos.
A pontuação ganha evolui de forma linear conforme a distância do inimigo. Quando o jogador elimina todos os aliens em cena, aparece um novo grupo.
O score aparece no canto superior esquerdo do ecrã, tal como pode ser visualizado na Figura 2.

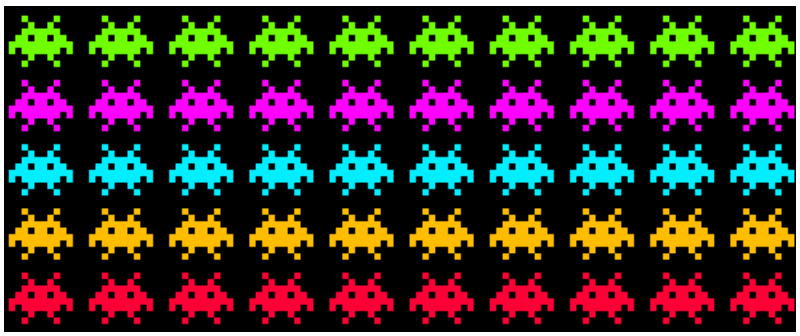

Figura 6 - Apresentação dos Alienígenas no jogo

Aliens

Os aliens são os inimigos do jogador. O ataque deles é também um disparo e cada vez que este atinge a nave o jogador perde uma vida.
Os aliens movimentam-se todos em grupo, quer na vertical quer na horizontal.

Figura 7 – Aliens

Barreiras

As barreiras são elementos estáticos do jogo que servem de proteção tanto para os aliens quanto para a nave, o seu tamanho diminui conforme esta é atingida pelos disparos, desaparecendo ao 3º tiro que a alveje.

Figura 8 - Estado inicial da barreira; barreira após um e dois ataques, respetivamente, da esquerda para a direita

Ranking

As melhores 10 pontuações conquistadas em jogos são guardadas e mostradas na página *Ranking*. As pontuações são guardadas no ficheiro highscores.csv. Ao premir a tecla *escape* volta-se novamente para a tela do menu.

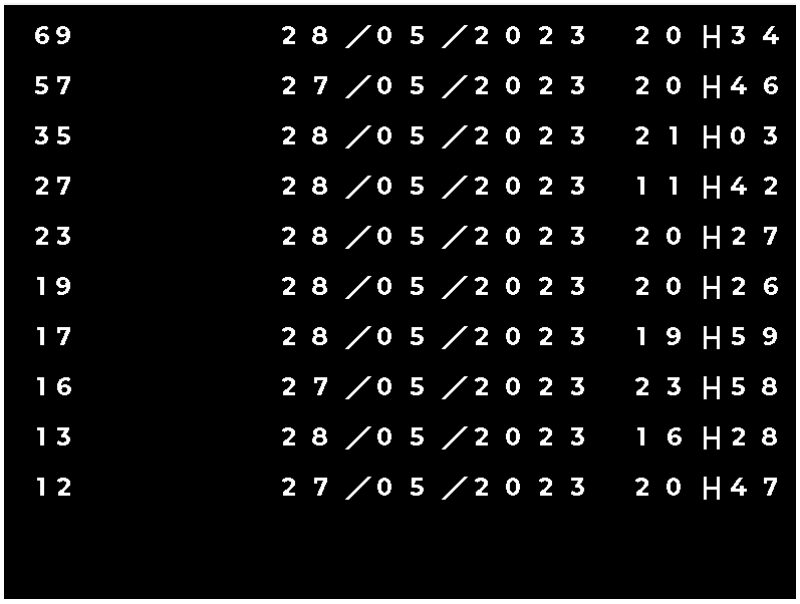

Figura 9 - Tela de classificações

2. Estado do Projeto

Funcionalidades

Foram implementadas com êxito todas as funcionalidades mencionadas na secção acima.

Funcionalidade	Dispositivos Associados	Estado
Mudar de página	Rato, Teclado e Placa de Vídeo	Concluído
Disparar	Rato, Teclado, Placa de Vídeo e Timer	Concluído
Desenhar os elementos do jogo/botões	Placa de Vídeo	Concluído
Atualizar Score	Placa de Vídeo	Concluído
Mover a Nave	Rato, Teclado, Placa de Vídeo, Timer	Concluído
Atualizar Ranking	Placa de Vídeo, RTC	Concluído

Uso de dispositivos de I/O

Dispositivo	Propósito	*Interrupts*
Timer	Controlar taxa de atualização do ecrã e temporizar ações do jogo e alterações nos seus elementos.	Sim
KBD	Controlar a nave do jogo e navegar pelo menu.	Sim
Rato	Controlar a nave do jogo e navegar pelo menu.	Sim
Placa de Vídeo	Menus e elementos visuais do jogo.	Não
RTC	Obter a data e hora do momento em que um jogo com pontuação elegível para o Top-10 acaba.	Não

Nota: Para cada dispositivo, as funções referidas estão sempre declaradas no ficheiro .h correspondente à parte da *framework* do dispositivo.

Timer

Foram processadas as interrupções geradas pelo Timer 0 tal como configurado pelo MINIX (60 por segundo) (`timer_interrupt_handler()`). Por ser do nosso interesse usar o Timer conforme pré-configurado, apenas foram subscritas as suas interrupções no início da execução do programa (`subscribe_timer_int()`), sem qualquer configuração prévia do dispositivo, e anulada a subscrição antes de o programa ser terminado (`unsubscribe_timer_int()`).
Ao atender as interrupções, é incrementado um contador. Através desse valor, os elementos do jogo são redesenhados e a sua posição e estado atualizados duas vezes por segundo.

KBD

Foram processadas as interrupções geradas pelo KBD tal como configurado pelo MINIX (`kbc_subscribe_int()`, `kbc_unsubscribe_int()`, `kbc_ih()`). O teclado é usado para controlar a navegação pelo menu e os movimentos e disparos da nave do jogo, pelo que os códigos (*make*/*break codes*) transmitidos no processamento das interrupções são interpretados de modo a constituírem mapeamento para as teclas correspondentes (`kbd_get_key()`), para posterior processamento e consequente conversão em ações concretas (ou nenhuma ação) na aplicação. As teclas mapeadas foram as de seta, a barra de espaços e a tecla de *escape*.

Rato

Foram processadas as interrupções geradas pelo Rato (através do KBC) tal como configurado pelo MINIX (`subscribe_mouse_int()`, `mouse_interrupt_handler()`, `unsubscribe_mouse_int()`). O rato é usado para controlar a navegação pelo menu e os movimentos e disparos da nave do jogo, pelo que os dados transmitidos pelos pacotes lidos no atendimento das interrupções são interpretados de modo a constituírem mapeamento para movimentos ou cliques, para posterior processamento e consequente conversão em ações concretas (ou nenhuma ação) na aplicação.

- Uso de posição
    - Controlo do cursor no menu e da posição da nave no jogo. (`update_mouse()`)
- Botões
    - O botão esquerdo é usado para clicar nas opções do menu e para desencadear um disparo no jogo. (`left_click()`)

Placa de Vídeo

- Modo de vídeo: 0x115 (`video_init()`)
    - Resolução: 800x600 (px, largura * altura)
    - Modo de cor: Direta
    - Construção dos componentes da cor (R:G:B): 24 bits (8:8:8) – 255*255*255 = 16 581 375 cores possíveis.
- Uso de *double buffering*
    - Os píxeis pretendidos são editados num *buffer* (memória “sombra” da VRAM), cujo conteúdo é copiado, no final de cada atualização do ecrã, para a VRAM.

Todos os elementos visuais da aplicação são desenhados com recurso a XPMs (`video_draw_xpm()`), criadas com recurso ao GIMP e adaptadas à especificação da função de desenho de XPM desenvolvida no âmbito do Lab 5: concretamente, foi adicionado o modificador `const` a cada *array* representativo de um XPM.

Real Time Clock (RTC)

- Usado para leitura de data (dia, mês e ano) e hora (hora e minuto) a pedido (`get_time()`).

3. Organização e Estrutura do Código

O código está organizado, primariamente, em dois subdiretórios: *framework* e *logic*. O primeiro contém o código encarregue de configurar, manipular e estabelecer contacto com os dispositivos usados. Já o segundo contém o código cujo propósito é o de constituir os elementos que asseguram a lógica e funcionamento da aplicação e do jogo.

Timer – 5%

Esta parte da *framework* contém algumas das funções desenvolvidas no Lab2 que puderam ser adaptadas de modo a ser utilizadas no projeto. Alguns exemplos das funções são a de subscrição de interrupções do timer e *interrupt handler.*

Contribuidores:

- Pedro Januário

Keyboard – 10%

Esta parte da *framework* contém algumas das funções desenvolvidas no Lab3 que puderam ser adaptadas de modo a ser utilizadas no projeto. Alguns exemplos das funções são a de subscrição de interrupções do *keyboard* e tradução do *break code* de várias teclas relevantes.

Contribuidores:

- Guilherme Ribeiro

Mouse – 10%

Esta parte da *framework* contém algumas das funções desenvolvidas no Lab4 que puderam ser adaptadas de modo a ser utilizadas no projeto. Alguns exemplos das funções são a de subscrição de interrupções do *mouse* e leitura dos *packets*.

Contribuidores:

- Mariana Rocha

Vídeo – 10%

Esta parte da *framework* contém algumas das funções desenvolvidas no Lab5 que puderam ser adaptadas de modo a ser utilizadas no projeto. Alguns exemplos das funções são a de inicialização do modo de vídeo e desenho de um pixel com uma cor específica no ecrã.

Contribuidores:

- Pedro Januário

RTC – 3%

Esta parte da *framework* contém algumas funções úteis para o desenvolvimento do projeto, nomeadamente para registar a hora a que um determinado resultado apto a integrar o top-10 foi atingido por um jogador.

Contribuidores:

- Pedro Januário

Utils – 1%

Esta parte da *framework* contém algumas das funções desenvolvidas principalmente no Lab2, e usadas nos outros Labs também, que puderam ser utilizadas no projeto. Alguns exemplos das funções são retirar os 8 bits menos e mais significativos de um valor de 16 bits.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Space Invaders – 20%

Esta parte contém a lógica mais geral possível, ou seja, tudo o que não seja respetivo a nenhum elemento do jogo específico, mas sim ao jogo como um todo. Alguns exemplos das funções são dar atualizar de todas as posições dos elementos, detetar colisões e o que fazer aquando dessas colisões, bem como garantir o desenho de todos os ecrãs diferentes do jogo.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Shot – 5%

Esta parte contém a lógica associada aos tiros do jogo, tanto dos aliens como da nave. Alguns exemplos das funções são a de inicialização das dimensões do tiro, deteção da origem do tiro e o movimento do mesmo.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Shield – 5%

Esta parte contém a lógica associada às 3 barreiras que servem de proteção à nave dos tiros dos aliens e às respetivas 3 camadas. Alguns exemplos das funções são a de inicialização das dimensões da barreira e redução das camadas do mesmo aquando atingido por algum tiro, seja oriundo de um alien ou da nave.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Player – 8%

Esta parte contém a lógica associada ao jogador (nave) e a todos os seus possíveis movimentos, ações ou atributos. Alguns exemplos das funções são a de inicialização das suas dimensões, o seu movimento, deteção de quando e onde disparar, aumento da sua pontuação e diminuição das suas vidas aquando atingido por um tiro e desenho da nave espacial, correspondente ao jogador.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Alien – 8%

Esta parte contém a lógica associada aos aliens e ao conjunto dos aliens e ao seu movimento e ações. Alguns exemplos das funções são a de inicialização do conjunto de 50 aliens, cada um na sua posição respetiva, o movimento de cada alien, a morte de cada alien atingido por um tiro do player e o disparo de um tiro aleatoriamente de um alien que esteja vivo.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Main – 15%

Esta parte contém o *loop* principal do projeto que contém a inicialização das variáveis necessárias e o início do ciclo *driver receive*. Depois, este *loop* principal chama, dependendo do estado atual do jogo, vários *loops* diferentes, correspondentes a cada estado possível do jogo (`highscores_loop()`, `game_loop()`, `main_menu_loop()`, `game_over_menu_loop()`), que por sua vez detetam as interrupções de cada dispositivo e agem de acordo com o pretendido.

Contribuidores:

- Pedro Januário
- Guilherme Ribeiro
- Mariana Rocha

Conversão de números em BCD para binário

O corpo da função `bcd_to_bin()`, declarada em *framework/utils/utils.h*, foi retirada de: https://github.com/Fabio-A-Sa/Y2S2-LabComputadores/tree/main/Labs/lab6#bcd-vs-bin%C3%A1rio.

Leitura do Top 10 de pontuações

Na função `load_scores()`, declarada em *logic/spaceinvaders.h* a parte de leitura de valores de ficheiros foi adaptada de: https://www.geeksforgeeks.org/relational-database-from-csv-files-in-c/.

Escrita do Top 10 de pontuações

Na função `store_scores()`, declarada em *logic/spaceinvaders.h*, a parte de escrita de valores em ficheiros foi adaptada de: https://www.programiz.com/c-programming/c-file-input-output.

Grafo de Chamada de Funções

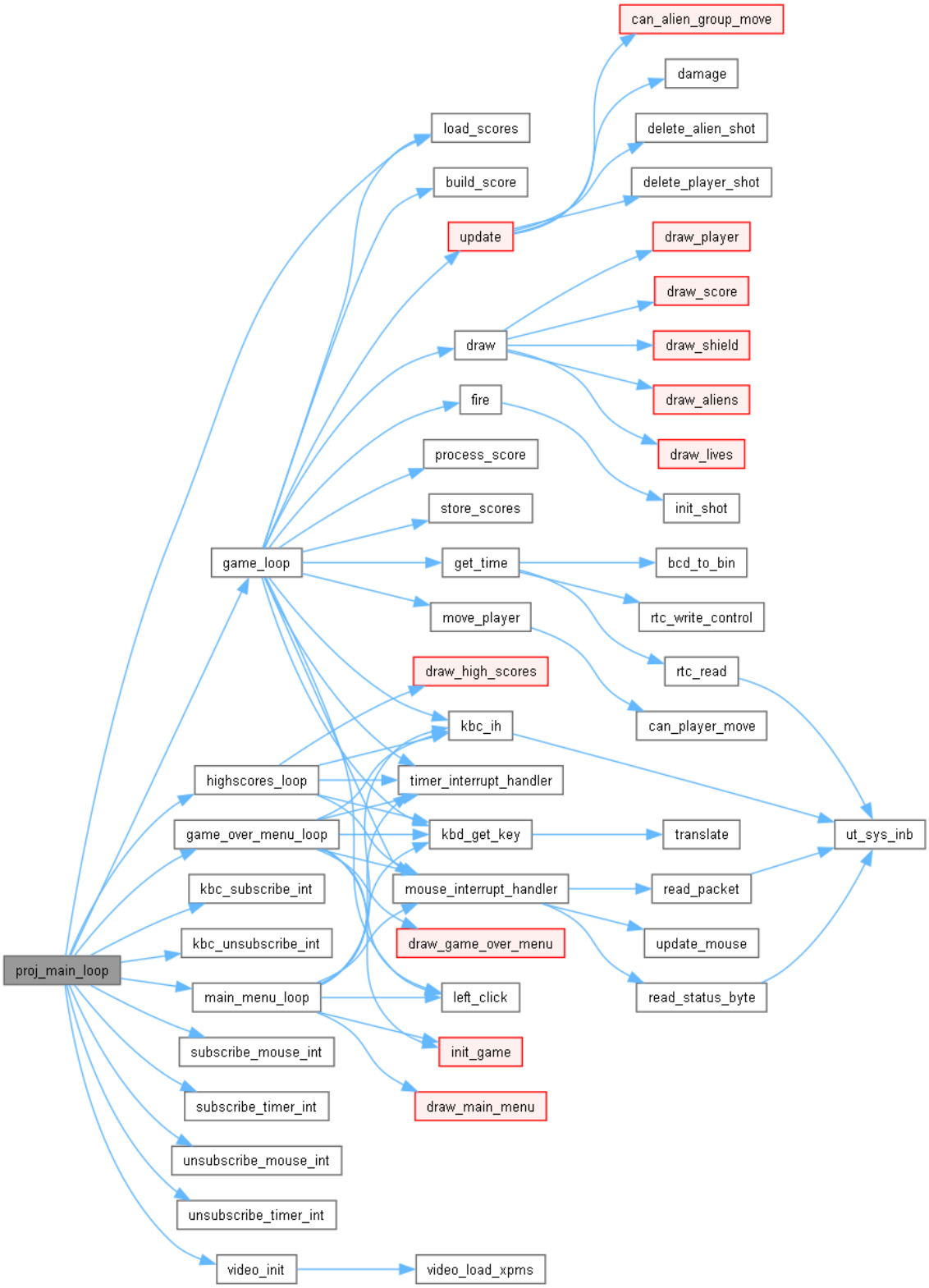

Figura 10 - Grafo de chamada de funções

4. Detalhes da Implementação

Placa de Vídeo

Inicialmente, de cada vez que uma imagem XPM era desenhada, a função da LCF que carrega um XPM era chamada. No entanto, acabámos por nos aperceber de que esta abordagem tinha um custo enorme e acabava por levar a erros.

Assim, optámos por seguir a estratégia de carregar todas as imagens XPM ao inicializar a placa de vídeo, apenas uma vez, no início da execução da aplicação. A função `video_load_xpms()`, declarada no ficheiro *framework/video/video.h*, encarrega-se do tal carregamento de todas XPMs usadas, cujos mapas e *arrays* são guardados em variáveis globais declaradas no mesmo último ficheiro .h referido.

Real Time Clock (RTC)

Na implementação da parte da *framework* relativa ao RTC, reparámos que os dados de data e hora recebidos na leitura dos registos do RTC se encontravam no sistema BCD e não no binário.

Assim, recorremos a uma função (`bcd_to_bin()`, declarada em *framework/utils/utils.h*) que efetua a conversão de números no formato BCD para números no formato binário.

Tal como já exposto, o corpo da função foi retirado de github.com/Fabio-A-Sa/Y2S2-LabComputadores/tree/main/Labs/lab6#bcd-vs-bin%C3%A1rio.

5. Conclusões

Concluímos com tudo o proposto na proposta inicial, com exceção da opção de *multiplayer*, visto que não foi atingida a implementação da porta-série. No entanto, tal não afeta a usabilidade do nosso projeto.

No futuro consideramos que a implementação da porta-série seria uma mais-valia para a valorização do projeto possibilitando não só vários jogadores ao mesmo tempo, mas também chats dentro do jogo.

Consideramos que este projeto é um meio interessante de cativar os estudantes aos conteúdos da disciplina, dando-lhes a possibilidade de implementar um programa à escolha.

Sentimos que a realização deste projeto nos concedeu uma melhor compreensão do funcionamento dos dispositivos bem como da interação entre eles.

Apêndice: Instruções de Instalação

A única particularidade na instalação nossa aplicação é o facto de, devido a particularidades do MINIX3 o caminho para o ficheiro *highscores.csv* ter de ser absoluto nas funções `load_scores()` e `store_scores()`, definidas no ficheiro *logic/spaceinvaders.h*. Assim, é importante que o utilizador verifique e, se necessário, corrija o caminho absoluto para o ficheiro nestas funções. Neste momento, o código está preparado para que o caminho absoluto seja */labs/proj/src/highscores.csv* ou */labs/proj/g3/src/highscores.csv*.