

Faculdade de Engenharia da Universidade do  
Porto



## Laboratório de Computadores

Turma: 15

Grupo: 3

**AquaBoy & LavaGirl**

David Gonçalves (up202208795)

Tomás Marques (up202206667)

João Santos (up201707427)

Luís Freitas (up201905767)

<b>Introdução.....</b>	<b>2</b>
<b>Instruções para o utilizador.....</b>	<b>3</b>
Menu inicial.....	3
Play.....	4
Leaderboard.....	7
Menu de vitória.....	8
Menu de derrota.....	9
<b>Estado do Projeto.....</b>	<b>10</b>
Tabela de funcionalidades.....	10
Tabela de dispositivos e usos.....	10
<b>Dispositivos usados e as suas funcionalidades.....</b>	<b>11</b>
Timer.....	11
Teclado.....	12
RTC (Real Time Clock).....	12
Rato.....	13
Video Graphics.....	14
<b>Estrutura/organização do código.....</b>	<b>16</b>
<b>Grafo da chamada da função.....</b>	<b>19</b>
<b>Detalhes de Implementação.....</b>	<b>20</b>
<b>Conclusão.....</b>	<b>21</b>
<b>Recursos utilizados.....</b>	<b>21</b>

# Introdução

Este projeto foi desenvolvido tendo por base o jogo *Fireboy & Watergirl*, cujo objetivo é ultrapassar os obstáculos existentes nos diversos mapas, de modo a alcançar a porta final. Cada mapa tem uma dificuldade diferente e esta vai progredindo conforme os 2 personagens chegam com sucesso ao destino, passando então para um outro mapa. Para movimentação das personagens é usado o “W”, “A” e “D” no caso do Aquaboy, e as setas de direção no caso da Lavagirl. A tecla “ESC” permite ao jogador abandonar o jogo a qualquer momento, voltando ao menu inicial.

# Instruções para o utilizador

## Menu inicial



No menu inicial é possível selecionar 3 opções: “Play”, “Leaderboard” e “Exit”.

- Caso “Play” seja selecionado, dá-se início ao jogo e aparece o mapa do primeiro nível.
- “Leaderboard” permite aceder a uma tabela com os registos dos 5 melhores resultados em função do tempo.
- “Exit” quando acedido neste ecrã fecha o jogo por completo.

# Play

Após a seleção de “Play” damos sim início ao jogo propriamente dito. Começamos no nível 1 e vamos progredindo, se não pertermos ;), até ao nível 3. Passando este vemos um ecrã “You Won” a felicitar-nos a partir do qual podemos repetir o jogo ou voltar ao menu inicial.

Os níveis em si também fazem uso do rato, visto que para alcançarmos as portas para o nível seguinte precisamos de “modificar” o nível carregando na alavanca presente no mapa.

## Nível 1:



## Nível 1 modificado:



## Nível 2:



## Nível 2 modificado:



## Nível 3:



### Nível 3 modificado:



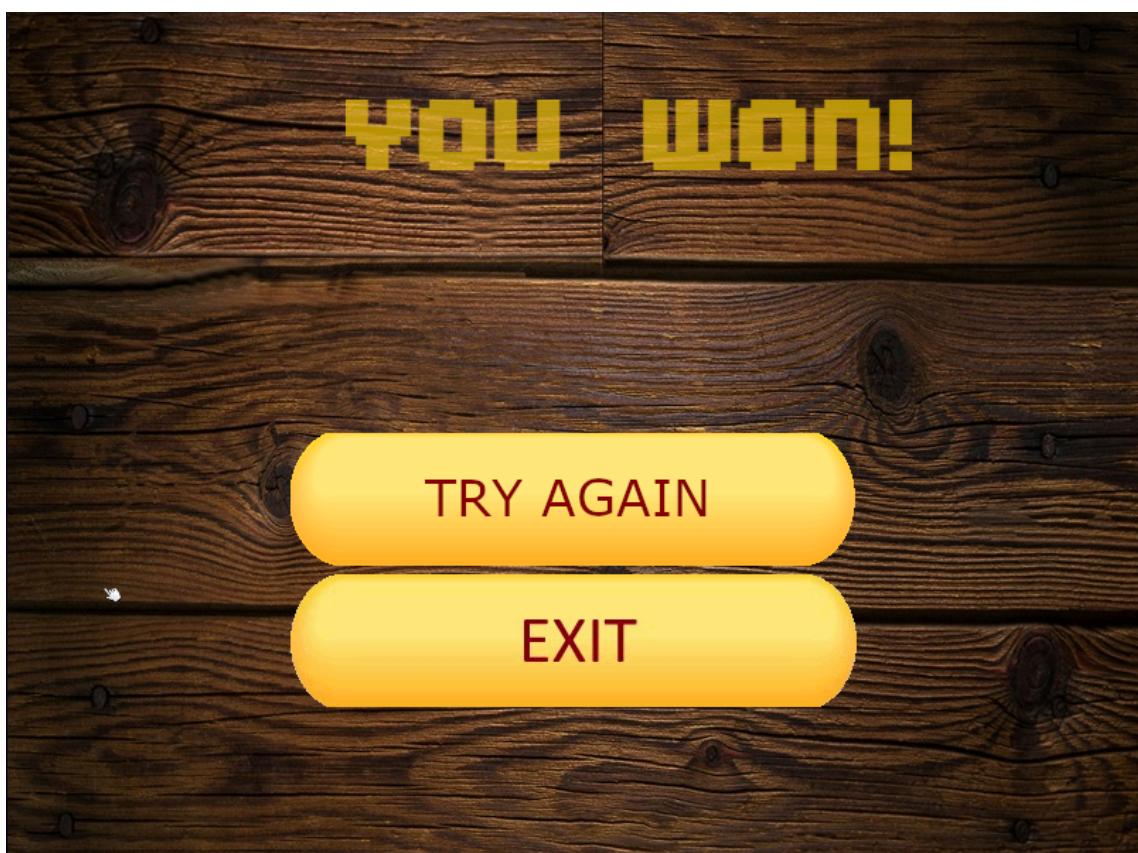
## Leaderboard

O menu “Leaderboard” apresenta uma lista “Top 5” das melhores pontuações do jogo, em termos de tempo (em segundos), tal como a data em que cada recorde foi estabelecido. Neste ecrã podemos utilizar o rato para voltar ao menu inicial, ou então utilizar a tecla “ESC”.

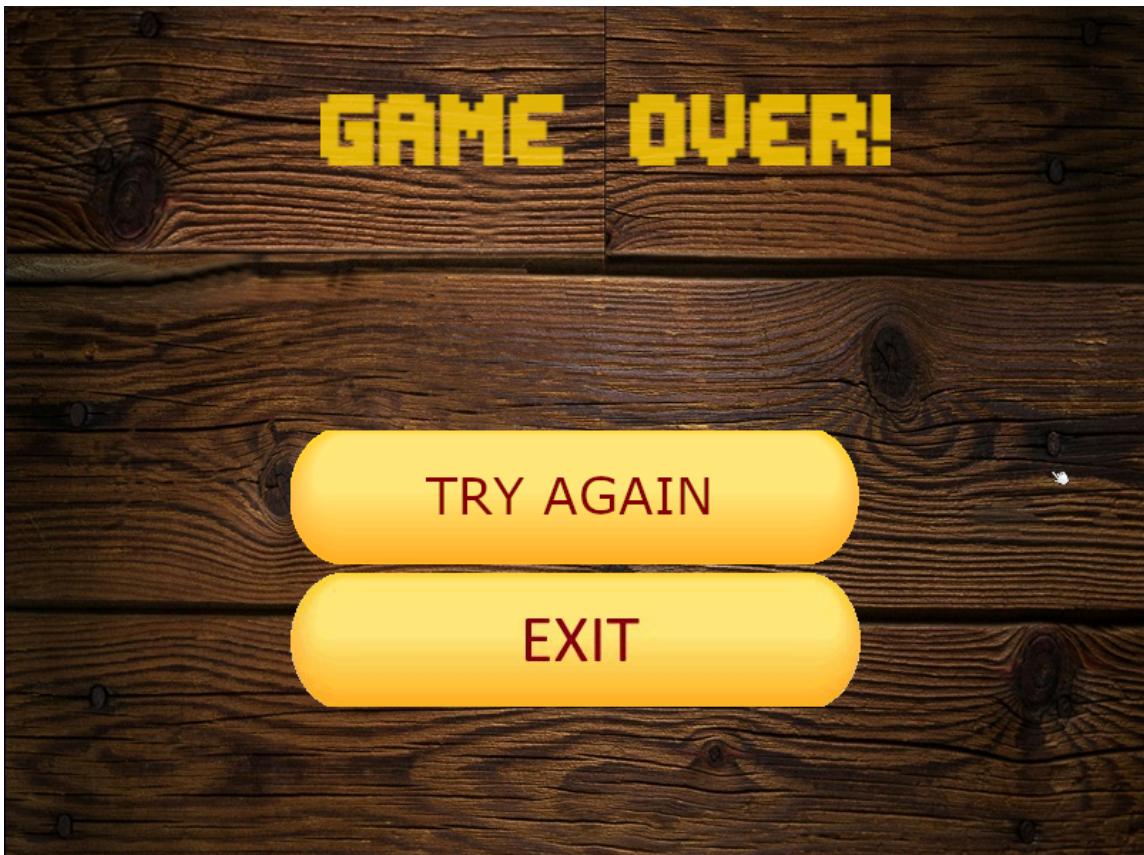
A screenshot of a game's leaderboards. The title "LEADERBOARD" is at the top in large yellow letters. Below it, there are two columns: "DATE" and "TIME". Five entries are listed, each consisting of a date (2/6/24), a time (in yellow), and a score (in red). A red arrow points to the left from the top-left corner of the board.

DATE	TIME	
2/6/24	15:31	43
2/6/24	15:44	45
2/6/24	15:08	46
2/6/24	15:09	50
2/6/24	15:14	55

## Menu de vitória



## Menu de derrota



Depois de completar os níveis com sucesso, o utilizador é redireccionado para o ecrã “You won!” que apresenta 2 botões, “Try again” e “Exit”. Assim o jogador pode repetir o jogo do início ou voltar logo para o menu.

# Estado do Projeto

## Tabela de funcionalidades

Funcionalidades Implementadas	Dispositivos Utilizados
Navegação entre menus	Rato, teclado e video graphics
Leaderboard	RTC e video graphics
Mostrar os menus e os diferentes elementos	Video graphics
Exibir o tempo do jogo atual	Timer e video graphics
Controlar o movimento dos bonecos	Teclado
Interagir com elementos dos níveis	Rato

## Tabela de dispositivos e usos

Dispositivo	Interrupções
Timer	Sim
Rato	Sim
Teclado	Sim
RTC	Sim
Video Graphics	Não

# Dispositivos usados e as suas funcionalidades

## Timer

O timer, mais propriamente o timer 0, é a base deste projeto e tem diversas funcionalidades como:

**Controlo da framerate:** Através da função `timer_set_frequency()`, o timer é configurado com uma frequência específica que corresponde ao número de interrupções do timer que serão efetuadas por segundo. Isto permite modificar a framerate do jogo e fazer com que o jogo seja atualizado a uma taxa constante.

**Atualização do estado do jogo:** A função `update_timer()`, chamada a cada interrupção do timer, é responsável por atualizar o estado do jogo, pela verificação do estado de vitória e o desenho de cada frame, que depende da framerate mencionada anteriormente.

**Atualização do tempo do jogo:** Durante o percurso de um jogo, o jogador poderá observar no campo superior direito o tempo que está a demorar para completar os 3 níveis implementados através do número de interrupções desde o início do jogo (com a função `timer_int_handler()`), e o número de interrupções que ocorre num minuto, dado pela framerate.

## Teclado

Tal como o timer, o teclado também funciona por interrupções (sendo chamada a função `update_keyboard()` em cada interrupção do teclado), e dependendo da página atual, a leitura das teclas premidas, lidas pela função `kbc_ih()`, atualizam o estado do jogo de forma diferente:

**Navegação entre menus:** Ao pressionar a tecla ESC, é possível navegar de volta ao menu inicial ou sair do jogo completamente, dependendo do estado do menu.

**Controlo do movimento dos bonecos:** Se o estado do menu for GAME (se estivermos a tentar passar os níveis) a função `action_handler()` vai ver se as teclas W, A, D, foram pressionadas e movimentar o *Aquaboy* de acordo, e faz o mesmo para a *Lavagirl*, com a diferença de verificar as setas.

## RTC (Real Time Clock)

Tal como os dois dispositivos anteriores, o RTC também funciona por interrupções, mas ao contrário deles, tem apenas uma funcionalidade:

**Registrar data/horas:** Quando um jogador completa os níveis com um tempo baixo, a data e horas atuais são passadas pelo RTC para a função `add_to_leaderboard()` juntamente com esse tempo, que insere a nova pontuação na posição correta do array “leaderboard”, empurrando as entradas com tempos mais elevados para baixo.

## Rato

Para este projeto, usámos não só os botões do rato como também a sua posição para, usando a função `check_mouse_click()`, interagir com os diversos elementos das diversas páginas com os objetivos de:

**Navegar entre menus:** No menu inicial, a função verifica se o cursor do rato está dentro do elemento definido pelo botão de jogar (play), o botão da leaderboard (leaderboard) ou o botão de saída (exit). Se o cursor do rato estiver dentro de um desses elementos quando o botão esquerdo do rato for pressionado, a função alterará o estado do menu ou o estado do sistema de acordo. O mesmo princípio aplica-se aos outros estados do menu (Gameover, Win, Leaderboard), em que a função verifica se o cursor do rato está na mesma posição de um botão específico quando o botão esquerdo é pressionado e, em seguida, realiza uma ação correspondente.

**Interagir com elementos do jogo:** Para além da navegação entre menus, o rato também é crucial na jogabilidade em si, uma vez que por vezes é necessário ativar alavancas para eliminar obstáculos no caminho dos nossos personagens, e isso é realizado com o rato de uma forma idêntica à interação da navegação dos menus, com a diferença de serem as alavancas os elementos a ser avaliados.

## Video Graphics

O video graphics é o dispositivo usado para mostrar todas as páginas e elementos do nosso projeto. Usamos este dispositivo no graphics mode na sua totalidade, com o modo 0X115, na resolução 800x600 e no modo de cor direto em que cada cor tem 3 bytes e portanto são suportadas  $2^{24}$  cores.

O display de todas as páginas e elementos é feito através de XPMs gerados através de imagens com recurso ao software GIMP para fazer a sua conversão. Os diversos números que aparecem no jogo são também XPMs, cada um deles. Para mostrar os XPMs em si, usamos como base a função `vg_draw_pixel()` implementada nas aulas práticas, para desenhar os pixels dos anteriormente mencionados XPMs. Para além disso, usamos este display de XPMs de formas diferentes e com utilidades diferentes:

**Double Buffering com Page Flipping:** Com o objetivo de evitar o flickering na renderização dos gráficos, implementámos um double buffering, em que usamos a função `buffer_copy()` para copiar o conteúdo do buffer duplo (onde os gráficos são desenhados) para o segundo buffer do ecrã. Isto é feito usando a função `memcpy` para copiar os bytes de um buffer para o outro. Para o page flipping, é chamada a função `flip_screen()` que por sua vez chama a função `buffer_copy()` para copiar o conteúdo do buffer duplo para o segundo buffer do ecrã, e em seguida é verificada qual ecrã está a ser exibido. Se for o primeiro (`video_mem`), ela muda para o segundo ecrã (`video_mem2`) e ajusta a posição vertical do ecrã para 0 usando a

função `change_screen_vertical_retrace(0)`. Se for o segundo ecrã aquele que está a ser exibido, ela muda para o primeiro e ajusta a posição vertical para `vres` usando `change_screen_vertical_retrace(vres)`, com recurso à função `0x07` da VBE.

**Sprites Animados:** Para representar os diversos elementos que compõem o nosso projeto, usamos uma classe “Sprite” muito semelhante àquela explorada nas aulas teóricas, através das funções `create_sprite()` que cria o sprite, `draw_sprite()` que desenha o sprite nas coordenadas do sprite, e `draw_sprite_pos()`, que desenha o sprite nas coordenadas passadas por argumento. Contudo, nem todos os Sprites são desenhados de forma igual, mais propriamente os personagens que, devido à sua característica única de movimento, são compostos por um array de Sprites e o seu display depende dos seus estados (`NORMAL`, `WALKRIGHT`, `WALKLEFT`, e `WINNING`), criando sprites animados.

**Deteção de colisões:** Sendo o nosso jogo um “jump ‘n’ run” a deteção de colisões é fulcral para a jogabilidade do jogo, e a função `checkCollision()` faz isso mesmo, tratando da colisão dos personagens com as diversas paredes, plataformas e “lagos” tóxicos, de lava e de água.

# **Estrutura/organição do código**

## **Módulo timer.c - 10%**

Este módulo do timer provém das aulas práticas, mais propriamente do Lab2, e é responsável pela manipulação e controlo do timer do sistema. As suas funções fornecem mecanismos para definir a frequência do timer, subscrever e cancelar a subscrição de interrupções do timer.

## **Módulo keyboard.c - 10%**

Este módulo do teclado, proveniente do Lab3, tem as funções necessárias para a subscrição e cancelamento da subscrição das interrupções do teclado, para verificar se o buffer de saída pode ser lido e também para ler o buffer de saída do teclado, ou seja, ler as teclas que foram pressionadas.

## **Módulo mouse.c - 10%**

O módulo do rato, tal como os anteriores, tem como origem as aulas práticas, mais propriamente o Lab4, com a adição de uma função que permite alterar a “sample rate” do rato. Ele lida também com a subscrição e cancelamento das subscrições do rato, comunicação com o seu controlador, envio de comandos e interpretação dos dados do rato.

## **Módulo vbe.c - 8%**

O módulo vbe, desenvolvido no Lab5, implementa funcionalidades relevantes para este projeto como a manipulação da interface gráfica do sistema, as funções necessárias para realizar o double buffering e page flipping, e o mapeamento da memória de vídeo para o espaço de endereçamento do processo.

## **Módulo video.c - 8%**

Tal como o módulo anterior, o módulo de vídeo também tem como origem o Lab5, sendo responsável por manipular os modos de vídeo do sistema, permitindo a transição entre o modo gráfico e o modo de texto, bem como a alteração da posição vertical do ecrã e a manipulação de artefactos do ecrã.

## **Módulo rtc.c - 7%**

Este módulo RTC, que não foi explorado nas aulas práticas, é usado na interação com o Real Time Clock, permitindo a leitura e atualização dos valores de tempo e data. As suas funções fornecem funcionalidades para subscrição e cancelamento de subscrição de interrupções do RTC, a verificação se o RTC está em atualização e se está em modo binário, e também a conversão de números do modo BCD para binário.

## **Módulo utils.c - 2%**

O módulo utils.c fornece funções utilitárias para a leitura de portas de entrada/saída e a extração de bits mais significativos (MSB) e menos

significativos (LSB) de um valor de 16 bits e também para ler bytes de uma porta de entrada/saída especificada.

## **Módulo draw.c - 15%**

Este módulo é dos módulos mais importantes, uma vez que gera a renderização gráfica do projeto, incluindo a exibição das diferentes páginas e sprites, sendo o desenho destas dependente do estado do menu. É também responsável por desenhar elementos dinâmicos como o tempo e data da leaderboard e as animações das personagens do jogo.

## **Módulo model.c - 15%**

O módulo model.c, como o anterior, tem uma grande importância, uma vez que gera todos os sprites e as suas interações dentro do jogo, focando na atualização e renderização dos estados do jogo, no tratamento de inputs do utilizador (tanto do teclado como do rato), e na gestão de mecânicas do jogo, como movimentos de sprites, colisões e estados do jogo.

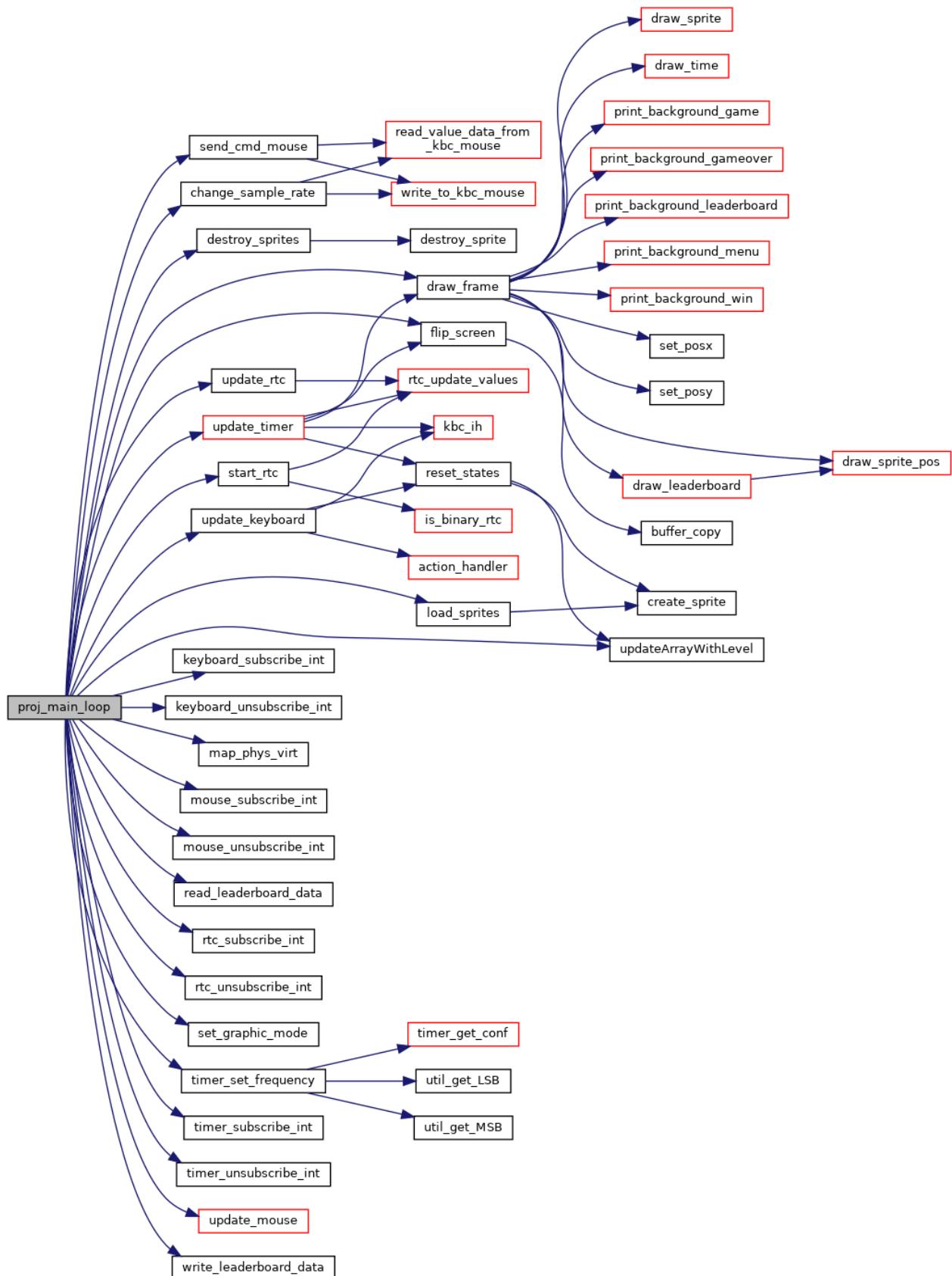
## **Módulo sprite.c - 5%**

Este módulo sprite.c proporciona as funções necessárias para a criação e destruição dos sprites, criando sprites a partir de arquivos xpm, e libertando a memória alocada para eles quando não forem mais necessários na sua destruição, tendo por base o código das aulas teóricas.

## **Módulo proj.c - 10%**

Este módulo proj.c é onde o jogo é executado. Ele inicia os dispositivos necessários, como teclado, mouse e RTC, e configura a interface gráfica. O loop principal controla a lógica do jogo, atualizando continuamente os eventos do teclado, mouse, temporizador e RTC. Enquanto o sistema estiver em execução, o loop principal gera as interrupções de hardware e atualiza o estado do jogo. Ao sair do loop, os recursos são libertados e os dispositivos são desativados, encerrando a execução do jogo.

# Grafo da chamada da função



# Detalhes de Implementação

A implementação da criação de frames no nosso projeto foi bastante influenciada pelo estado do menu, onde o conteúdo visual dependia do estado atual do menu. Embora a fusão entre a criação de frames e a lógica do estado do menu não tenha sido extremamente difícil, também não foi trivial. Envolveu muitos detalhes, como a reinicialização de variáveis de acordo com o estado do menu, o que aumentou a complexidade da integração desses dois aspectos.

Em relação ao uso do RTC, não encontramos grandes dificuldades. Com base nos materiais fornecidos, como os slides sobre o RTC, e nas informações disponíveis no [GitHub do Fábio Sá](#), que também foram muito úteis não só na clarificação de dúvidas que tivemos ao longo do projeto mas sim ao longo do semestre todo, conseguimos implementar uma leaderboard dependente do RTC que regista a data em que o jogo foi completado sem grandes problemas.

Além dos tópicos abordados em aula, enfrentamos desafios adicionais ao lidar com a animação dos sprites e a deteção de colisões. Embora trabalhosas, essas tarefas não foram particularmente difíceis. No entanto, um dos maiores obstáculos que enfrentamos foi a responsividade do rato, que às vezes era lenta. Isso levou-nos a desenvolver uma função para ajustar a taxa de amostragem do rato, um tópico que não foi abordado nas aulas.

# Conclusão

No projeto conseguimos implementar efetivamente todos os dispositivos e funcionalidades propostos apesar das dificuldades que foram surgindo ao longo do seu desenvolvimento, destacando como uma delas, por exemplo a fluidez dos movimentos. Na nossa opinião os aspetos que poderiam ser melhorados seriam neste caso a implementação do Serial Port, de modo a que fosse permitida a jogabilidade entre 2 computadores diferentes em simultâneo, e aumentar a complexidade do jogo no que concerne ao nível de mapas e dificuldade. Contudo, a elaboração deste projeto resultou na melhor compreensão da funcionalidade dos dispositivos abordados durante as aulas práticas, e no aumento do nível de conhecimento acerca da linguagem de programação C.

# Recursos utilizados

[GitHub Lara Bastos](#) → alguns sprites utilizados

[GitHub Fábio Sá](#) → melhor compreensão da matéria ao longo do semestre e clarificação de dúvidas no decorrer do projeto