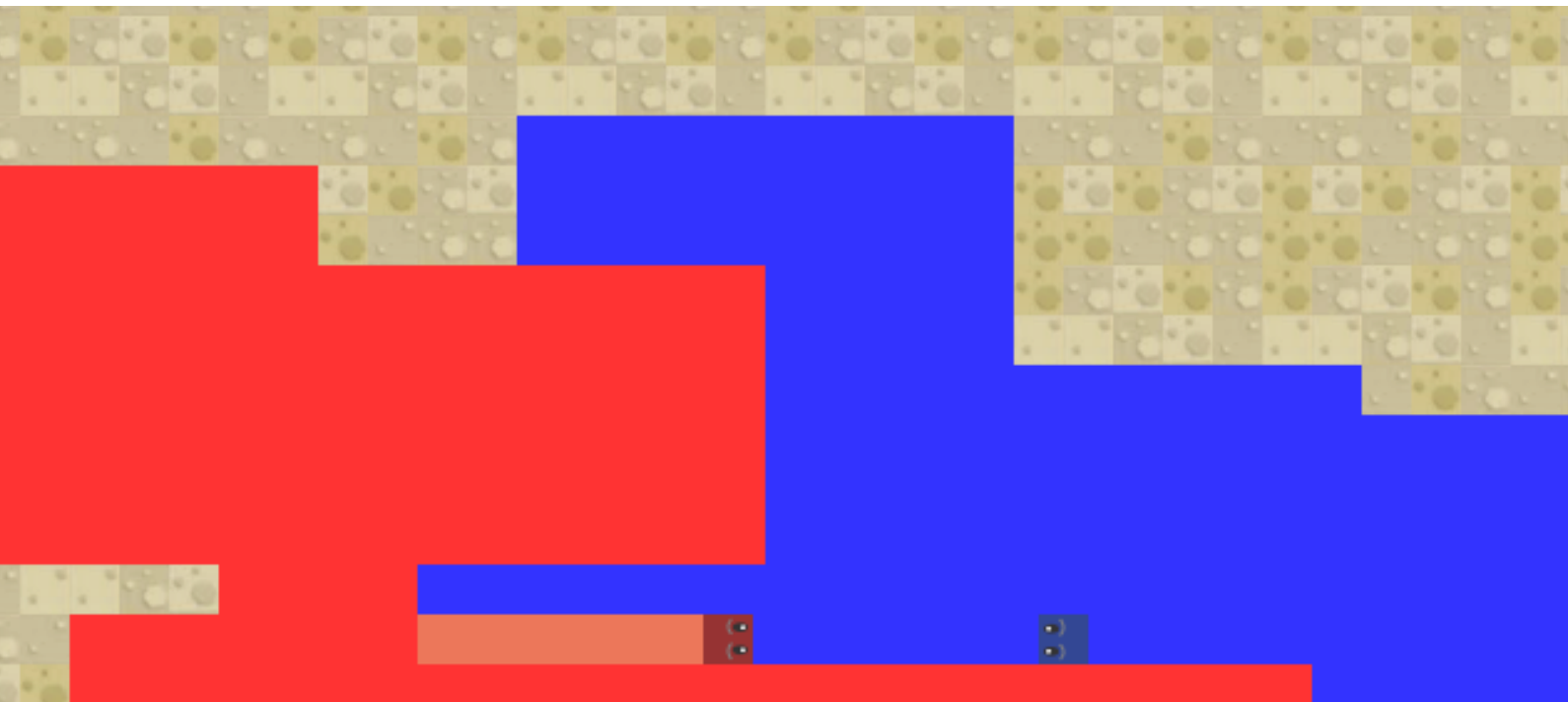
Faculdade de Engenharia da Universidade do Porto

Laboratório de Computadores

Turma 12 Grupo 2

Last One Standing

Miguel Lopes Guerrinha - up202205038
Miguel Simão Duarte - up202206102
Rui Pedro da Silva Cruz - up202208011



Índice

1. Introdução

Last One Standing é um jogo multiplayer desenvolvido no sistema operacional Minix que tem como modelo o jogo Land-io.

No decorrer do semestre na cadeira de Laboratório de Computadores, foi se aprendendo a controlar diferentes periféricos de entrada/saída e os seus modos de funcionamento, bem como a implementação de drivers e interrupt handlers para todos os dispositivos aprendidos na unidade curricular.

Como os dispositivos periféricos são uma parte fundamental dum computador, neste projeto foi incluído o uso de diferentes periféricos comuns como o rato, o teclado, o timer e a gráfica de vídeos. Além disso, foi também utilizado neste projeto o Real-Time-Counter.

Implementados estes dispositivos, o Last One Standing visa a colocar dois jogadores numa disputa de território durante um período específico de tempo. Ambos os jogadores conseguem matar-se um ao outro podendo assim mais facilmente lutar por uma maior quantidade de território e manter o espírito do jogo sempre vivo. No final do período de tempo de jogo, o jogador com maior percentagem de território leva por vencido o seu adversário.

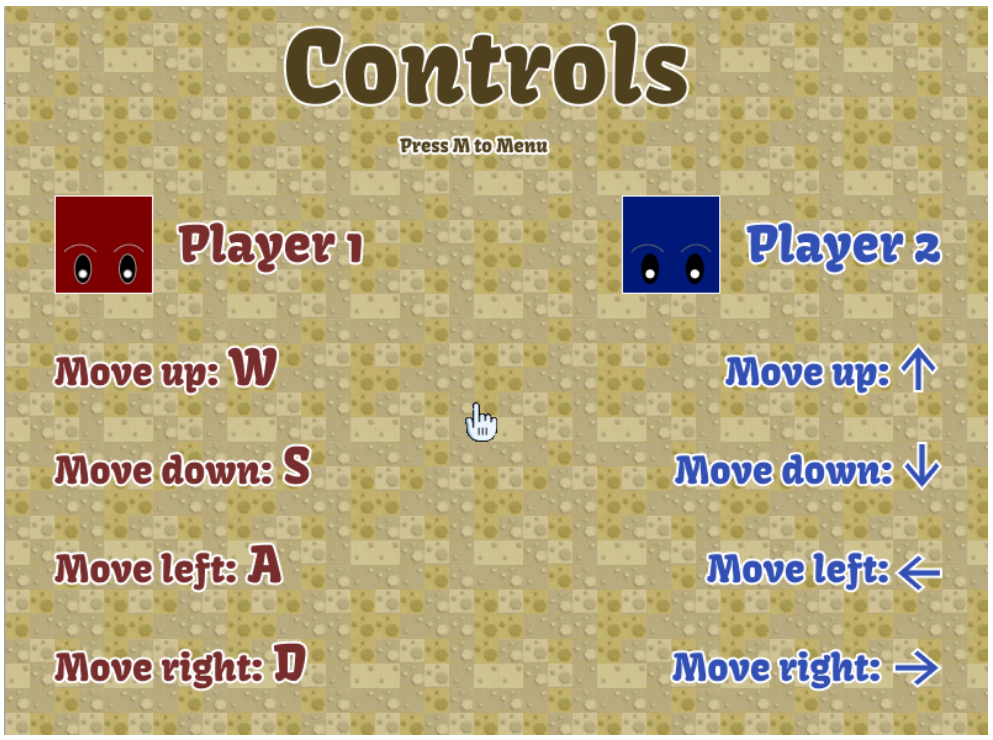
2. Instruções de Utilização

2.1 Menu Principal

Logo quando o jogo iniciado, é mostrado ao utilizador o menu principal:

Neste menu o utilizador pode clicar no “Enter” para iniciar o jogo, ou no “c” para aceder ao menu Controlos. Pode também utilizar o rato clicando com o botão esquerdo em cima das letras da funcionalidade que pretende selecionar.
Se pretender sair do jogo, o utilizador pode clicar na tecla “q”. Esta funcionalidade pode ser ativada a qualquer momento.

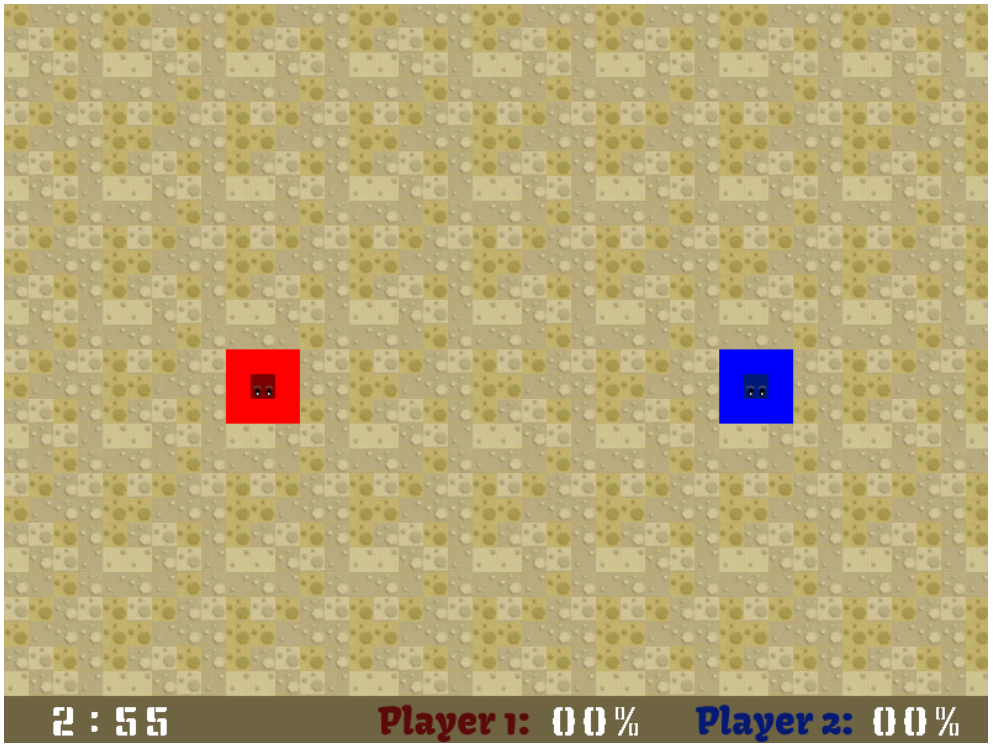
2.2 Menu de Controlos



Aqui são mostrados os comandos necessários para jogar.

Clicando na tecla “m” é possível retornar ao menu ou então, se preferimos, basta clicar com o botão esquerdo do rato em cima das letras “Press M to Menu”.

2.3 Arena de jogo



Ao iniciar o jogo aparecem dois jogadores como demonstrado na imagem acima. O jogador vermelho pode ser movido para cima, baixo, esquerda ou direita utilizando as teclas “W”, “A”, “S”, “D”, respetivamente. O jogador azul pode ser movido para cima, baixo, esquerda ou direita utilizando as respectivas setas. Além destas opções de movimentação também podemos utilizar o rato.

Ambos os jogadores movimentam-se sozinhos e os “inputs” fornecidos pelo utilizador apenas servem para controlar a direção que o jogador vai seguir.

Para conquistar território o jogador tem de sair do seu território, deixando um rasto. Regressando ao seu território toda a área delimitada pelo rasto é transformada em território. Quando fora do seu território o rasto pode ser interceptado pelo outro player fazendo com que todo o território conquistado seja perdido, voltando ao ponto inicial do jogo.

Chegando ao final do cronómetro que se encontra no canto inferior esquerdo, o jogo termina e é mostrado o menu de vitória, respectivo ao jogador que conquistou mais território.

2.4 Menus fim de jogo

Estes menus aparecem consoante o jogador que ganhar o jogo. Tal como nos outros menus podemos utilizar as teclas referidas ou o botão esquerdo do rato para aceder às páginas descritas.

3. Estado do Projeto

Dispositivos Implementados:

Dispositivo	Funcionalidades	Interrupções
Timer	Sincronizar tempo de jogo e dispositivos	Sim
Teclado	Controlar os movimentos dos jogadores e interação entre menus	Sim
Rato	Controlar os movimentos dos jogadores, selecionar opções desejadas relativamente aos menus	Sim
Placa Gráfica	Apresentar todas as interfaces do jogo, desde menus, mapa e jogadores	Não
Real Time Counter	Mostrar a data atual sempre que o jogo é corrido	Não

3.1 Timer

O timer foi utilizado para, a cada interrupção, sincronizar a informação presente no buffer secundário para o buffer principal (VRAM). Permite ainda sincronizar o tempo de jogo pré-definido (3 minutos), fazendo com o mesmo seja terminado quando o tempo chegar ao fim. O RTC é atualizado a cada 3600 interrupções (1 minuto), controlando de uma forma moderada a sua taxa de atualização.

Funções associadas:

- Todas as funções presentes no ficheiro timer.c
- change_timer_state() - gere a sincronização dos buffers, usando sync_to_main_buffer() e o tempo de jogo, ambas incrementando a variável global “timer_counter” em timer_int_handler()

3.2 Teclado

O teclado foi utilizado para processar todos os inputs do utilizador nas diversas páginas. A cada input é gerada uma interrupção, que após ser processada obtemos um scancode que irá executar uma certa operação. O teclado permite navegar entre menus, e mover os dois jogadores. Para mover o jogador vermelho utilizamos as teclas “W”, “A”, “S”, “D”, para mover o jogador azul utilizamos as 4 setas. A tecla “Enter” é usada para começar um novo jogo, a tecla “C” é utilizada para aceder ao menu controlos e a tecla “M” para aceder ao menu principal.

Funções associadas:

- Todas as funções presentes no ficheiro keyboard.c
- change_kb_state() - A cada interrupção é chamada a função kbc_ih() que irá ler o scancode gerado. O scancode é posteriormente processado e é executada uma certa funcionalidade

3.3 Rato

O rato é principalmente usado na movimentação e interação com os menus. O jogador ao clicar com o botão esquerdo dentro da área de um dos textos, por exemplo “Press Enter to Play”, será redirecionado para para a arena de jogo. Esta funcionalidade trabalha em paralelo com o teclado, uma vez que o “Enter” também poderia ser utilizado para iniciar o jogo.
Dentro do jogo, os jogadores têm a alternativa de se movimentar pelo mapa usando o rato (ainda que a forma mais comum seja utilizar o teclado). Se pressionarmos o botão do lado esquerdo, arrastarmos o rato para a direção que queremos e depois soltarmos o botão, o jogador vermelho irá mover-se nessa direção, o mesmo acontece para o jogador azul se utilizarmos o botão direito do rato.
Para tal, usamos uma estrutura personalizada do “MouseInfo”, onde cada três bytes gerados, são armazenados na estrutura, salvando informações consideradas pertinentes para salvar, ou seja, posição atual do rato e botões pressionados.

Funções associadas:

- Todas as funções presentes no ficheiro mouse.c
- change_mouse_state() - A cada interrupção, é gerado um byte, onde para cada três bytes é formado um packet, realizado em mouse_sync_bytes(), necessário à leitura dos dados do rato e mouse_move_to_info() para transferir os dados do packet atual para o “MouseInfo”

3.4 Placa Gráfica

A placa gráfica é essencial para o display de todos os elementos necessários para o funcionamento do jogo como, os menus, o mapa e o território associado a cada jogador, tendo em conta que o mesmo irá se alterar inúmeras vezes à medida que o jogo vai avançando.

Relativamente aos personagens e aos números, estes são efetuados via XPMs, gerados inicialmente de imagens PNG, no caso específico dos personagens, a imagem irá ajustar-se conforme a direção dos mesmos.

O modo de vídeo utilizado para desenvolver este jogo foi o VBE0x115, com uma resolução de 800x600 pixels, com cada cor ocupando três bytes (8:8:8) no modo direto, permitindo a utilização de aproximadamente 16,8 milhões de cores.

Para evitar problemas de lag utilizámos a técnica “Double Buffering” via copy, onde o buffer principal é atualizado a cada interrupção do timer copiando a informação presente no buffer secundário, enquanto que este apenas é alterado quando algum elemento muda de estado.

Nas situações em que utilizamos o rato criamos um buffer dedicado onde guardamos todo o background que posteriormente é constantemente copiado para o buffer principal. Assim apenas desenhamos o background uma vez, o que aumenta significativamente a velocidade de atualização da posição do rato.

Durante o jogo há situações em que os jogadores colidem, tanto com os limites da arena, como com os rastros, deles mesmos ou do jogador inimigo. Todas essas colisões são detetadas e devidamente processadas.

Funções associadas:

- Todas as funções presentes nos ficheiros graphics.c, view.c e sprite.c
- sync_to_main_buffer() - necessária à utilização do “Double Buffering” via copy

3.5 RTC

O RTC é utilizado para ler a hora, minutos, dia, mês e ano atuais em que o jogador está a jogar. Estas informações são apresentadas no menu principal.

Funções associadas:

- Todas as funções presentes no ficheiro rtc.c
- change_rtc_state() - Atualiza a struct, rtc_info, a cada 3600 interrupções do timer
- draw_time() - Atualiza no menu principal as informações presentes na struct rtc_info

4. Organização/Estrutura do Código

4.1.1 Módulo do Timer - 5%

As funções relativas ao timer baseiam-se nas desenvolvidas no Lab2, permitindo essencialmente ativar e desativar as interrupções do timer, obter a sua configuração de forma a configurar a sua frequência conforme a desejada e ainda incrementar uma variável global relacionada ao número de interrupções.

4.1.2 Módulo do Teclado - 8%

As funções relativas ao teclado baseiam-se nas desenvolvidas no Lab3, apesar de apenas necessitarmos especificamente das que utilizam interrupções e o kbc_ih() que trata de receber os scancodes do KBC.

4.1.3 Módulo do Rato - 8%

As funções relativas ao rato baseiam-se nas desenvolvidas no Lab4, onde à semelhança do teclado existe um interrupt handler, que através do KBC, recebe um byte recebido por cada interrupção do rato. Mantivemos funções que escrevem comandos para o rato, necessárias para ativar a “data report” e o “stream mode” do rato e que sincronizam as informações do packet atual com a estrutura personalizada “MouseInfo”.

4.1.4 Módulo do KBC - 2%

As funções do KBC servem de auxílio tanto ao teclado como ao rato, podendo ler o status e output do KBC ou escrever comandos necessários no mesmo.

4.1.5 Módulo da Placa Gráfica - 8%

As funções relativas ao rato baseiam-se nas desenvolvidas no Lab5, tendo como objetivo configurar a placa gráfica de acordo com os modos desejados e desenhar todos os elementos presentes no jogo. Optamos ainda por não implementar certas funções, como por exemplo o draw_rectangle() por não serem necessárias ao funcionamento geral do jogo.

4.1.6 Módulo do RTC - 2%

As funções do RTC servem apenas para a leitura da data atual em que o jogo está a ser jogado, sendo necessário a sua configuração, verificação em caso de ser lido em BCD ao invés de binário e update.

4.1.7 Módulo do Utils - 2%

Este módulo, a par do KBC, serve de auxílio principalmente à implementação do timer e restantes labs, uma vez que possui funções para obter no caso de valores de 16 bits, o LSB ou o MSB e a leitura de um valor de 8 bits através de um inicialmente com 32.

4.1.8 Módulo do Game - 20%

Neste módulo estão as funções necessárias ao funcionamento do jogo, nomeadamente o movimento dos jogadores e consequentes colisões que o mesmo possa proporcionar. É ainda necessário preencher o território conquistado quando um jogador retorna ao seu território original, obtido pela função fill_territory(). Para que a percentagem ganha seja incrementada conforme o jogo vai decorrendo utilizamos uma função auxiliar para contar o território atual, de cada vez que o jogador retorna.

4.1.9 Módulo do Model - 15%

O model serve para dar setup de todos os sprites que irão ser utilizados, seja no decorrer do jogo, seja nos menus. Nele estão ainda presentes todas as funções que tratam dos states do timer, rtc, teclado e rato, ou seja, como são tratados os outputs recebidos por cada controller, no caso do teclado, consoante a tecla pressionado, iremos para um dado local do jogo, o rato irá nos permitir clicar e navegar tanto pelos menus como pelo jogo. A arena/mapa é também inicializada neste módulo, onde cada jogador inicia o jogo com um certo território, com o objetivo de o poder expandir.

4.1.10 Módulo do Player - 8%

Neste módulo é tratada a lógica da movimentação de cada jogador, restringindo a área do mapa ao qual este pode atuar.

4.1.11 Módulo do Sprite - 2%

É neste módulo que se cria cada elemento visual do jogo, alocando a memória necessária à criação dos mesmos, e posteriormente, libertando-a evitando problemas de “memory leak”.

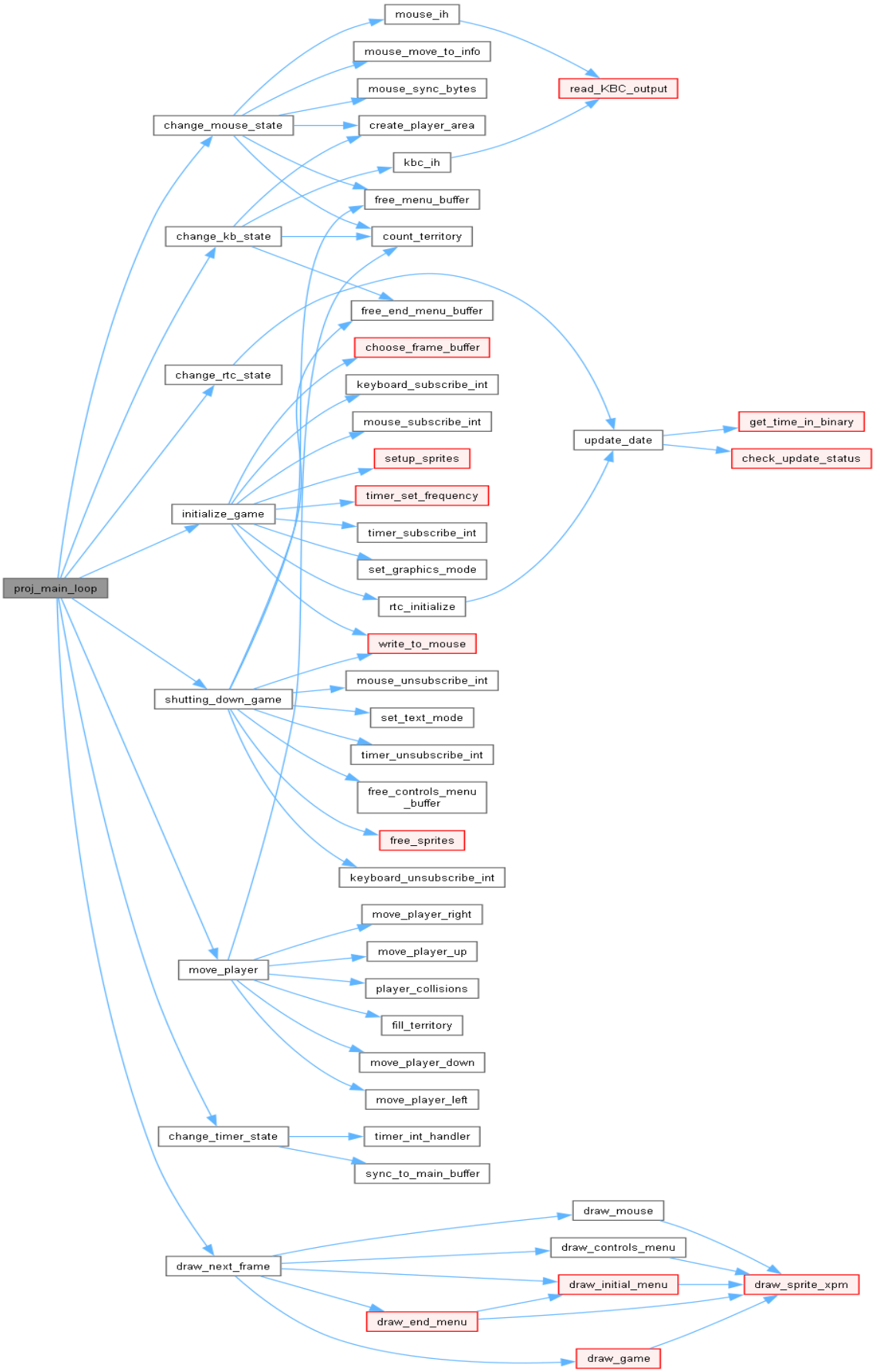
4.1.12 Módulo do View - 10%

Neste módulo são tratados todos os métodos de desenho dos elementos do jogo, recorrendo para isso às funções presentes no módulo anterior (Sprite).

4.1.13 Módulo do Proj - 10 %

Em suma, neste módulo encontra-se o loop principal do jogo, e onde são subscritas as interrupções de todos os dispositivos utilizados, sendo que no final do mesmo, estas são sempre “unsubscribe”, limpando ainda qualquer tipo de memória alocada seja nos sprites, ou nos buffers auxiliares que foram criados.

4.2 Função Call Graph



5. Detalhes de Implementação

5.1 RTC

Para podermos mostrar a data e hora a que estamos a jogar é necessário primeiro atualizar a struct “rtc_info” e como apenas mostramos a data, a hora e os minutos apenas fazemos essa atualização a cada 3600 interrupções do timer, evitando leituras desnecessárias.
Para lermos as informações pretendidas do RTC enviamos o código de comando “10” para o registo 0x70 e posteriormente lemos a informação do registo 0x71.
Para garantir que as leituras eram feitas corretamente, tivemos em atenção se o modo de contagem está em binário ou em BCD e se o RTC estava a atualizar valores. Após uma leitura correta todos os campos da struct “rtc_info” são devidamente preenchidos.

5.2 Buffers dedicados

Devido a problemas de delay no funcionamento do rato nos diversos menu decidimos, além do double buffering, utilizar buffers dedicados para esses menus de forma a evitar desenhar constantemente o background do menu e assim diminuindo significativamente o tempo de resposta do rato.

5.3 Detecção de Colisões

De forma a garantir que os jogadores não saem do mapa ou que colidiram com algum rastro presente na arena, foi necessário implementar a detecção de colisões entre os vários elementos do jogo.
A implementação desta funcionalidade partiu de verificar se a posição que o jogador irá ter após se movimentar, está livre ou não. Se não, o movimento não é realizado, no caso dos limites da arena, ou tem uma consequência associada, no caso de o elemento ser um rastro.

5.4 Animações

Implementámos animações no deslocamento dos jogadores que mudam consoante a direção que cada jogador tem.

5.5 Deslocamento do rato combinado com clique no botão

Por fim, implementámos a movimentação de ambos os jogadores com a utilização do deslocamento do rato combinado com o clique num dos botões do mesmo. Para conseguir utilizar esta funcionalidade verificamos primeiro em que momento o botão é clicado e guardamos a posição do rato nesse momento, posteriormente quando o botão é largado comparamos a posição nesse momento com o momento inicial e se o deslocamento no eixo dos x for maior que o deslocamento no eixo dos y a direção do player será alterada para a esquerda ou para direita consoante a direção do deslocamento e vice-versa.

6. Conclusão

Ao longo da elaboração do projeto tivemos, como é óbvio, bastantes desafios e obstáculos que com tempo e esforço conseguimos ultrapassar. Primeiramente tivemos a desistência de um colega de grupo que proporcionou uma maior carga de trabalho para cada um dos outros elementos. Além disso, tivemos outros desafios no projeto que nos obrigaram a pensar melhor e refazer certas funcionalidades para que não pudéssemos tirar o melhor proveito do produto. Infelizmente, não conseguimos ter tempo de terminar a implementação da Serial Port.

No futuro, caso pudéssemos adicionar algumas funcionalidades novas, optaríamos por ter um maior número de jogadores em ativo.

Como principal destaque é de realçar o preenchimento de novo território pelo jogador aquando da sua chegada ao seu território antigo. Além disso, também é de salientar a possibilidade de um jogador ser única e exclusivamente movido pelo rato.

Concluindo, aprimoramos ainda mais o nosso conhecimento acerca dos diferentes periféricos e dispositivos de um computador e um melhor controlo sobre estes. Apesar das dificuldades que existem não só neste mas em todos os projetos, foi um gosto e "divertida" a elaboração do Last One Standing. Estamos contentes com o resultado final.