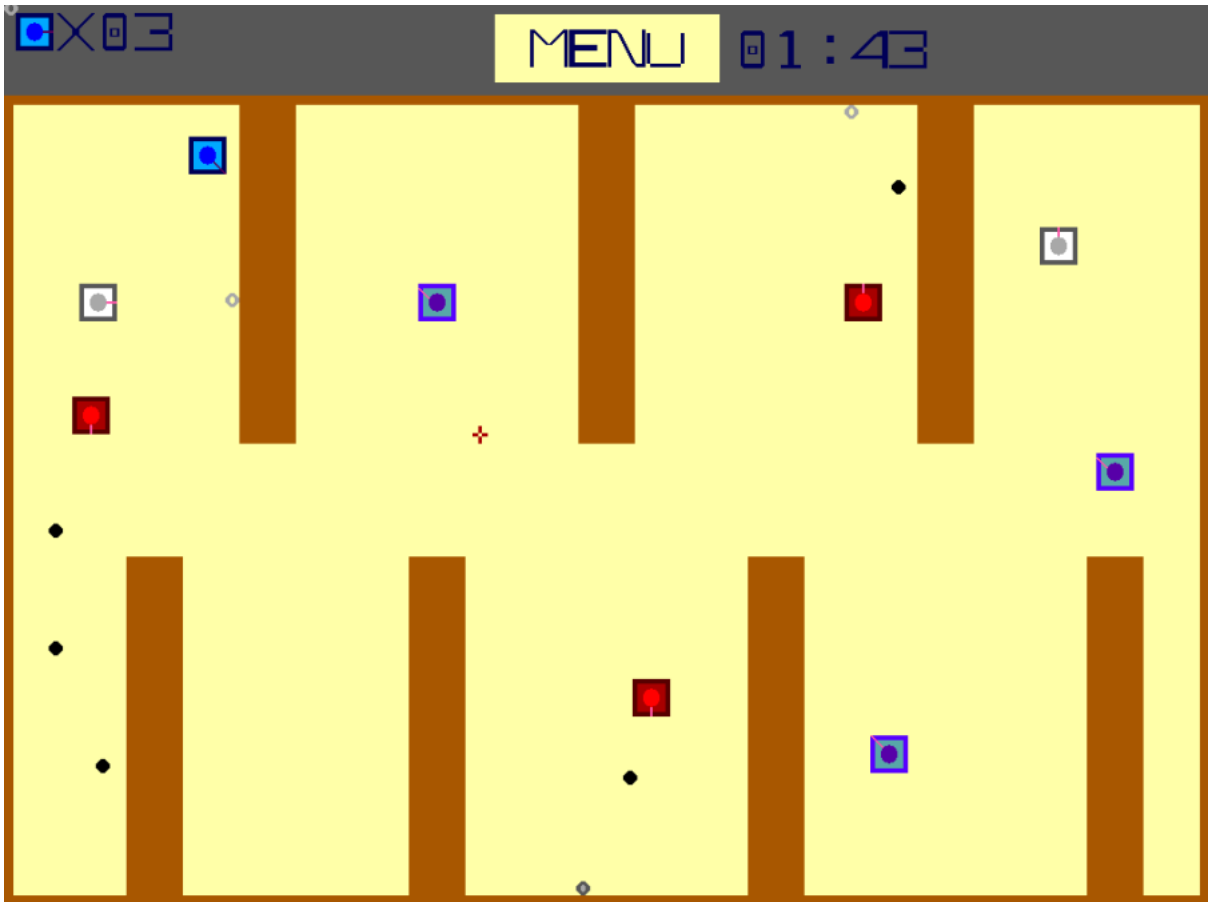
U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Projeto de LCOM: TANKINIX



LCOM 2020/2021

Mestrado Integrado em Engenharia Informática e Computação:

Trabalho realizado por:

José Pedro Ferreira up201904515@fe.up.pt

José Frederico Rodrigues up201807626@fe.up.pt

Índice

Lista de Figuras

Lista de Tabelas

1. Introdução

O nosso projeto, denominado **Tankinix**, consiste num jogo de ação 2D com uma prespetiva “top-down” (de cima para baixo). Neste jogo, o jogador controla o tanque azul, através de níveis, geralmente repletos de tanques inimigos. O objetivo do jogador pode ser o de sobreviver por um determinado tempo ou eliminar todos os restantes tanques num determinado nível.

Figura 1 – Exemplo da ação de um nível do Tankinix, com o tanque azul a representar o jogador no centro do ecrã

2. Instruções do Utilizador

O principal meio de navegação do utilizador nos menus do Tankinix é o apontador do rato. Este tem um aspeto “retro” e é interativo na medida em que alterna entre a sua aparência geral branca, para um *design* preto quando o utilizador tem pressionado o botão esquerdo do rato.

Figura 2 – Aspeto do apontador do rato nos menus

Os botões com os quais o utilizador pode interagir, mudam de aparência quando a seta lhes passa por cima, de modo a ser claro para o “user” que pode selecionar a referida opção

Figura 3 – Mudança das caixas de opção dos menus. Na esquerda a versão da caixa quando a seta não está a selecionar a opção. Na direita o botão na sua versão alternativa, com o apontador a passar-lhe por cima

2.1. Menu Inicial

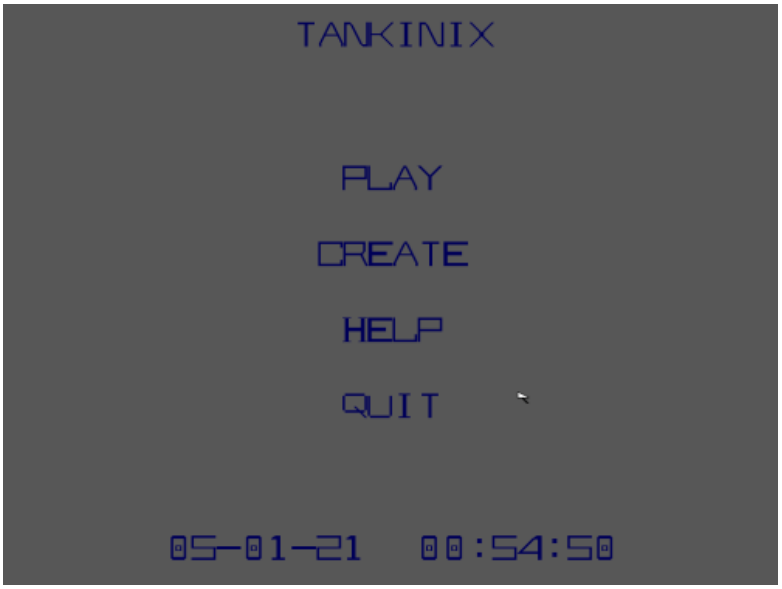


Figura 4 – Menu Inicial

O Menu Inicial apresenta as opções “Play”, “Create”, “Help”, e “Quit”. Se o utilizador

escolher “Quit”, o jogo fecha-se e devolve o Minix à sua configuração inicial (text mode). No caso de o utilizador optar por “Play” será levado ao menu de seleção de nível.

2.2. Menu Escolha do Nível

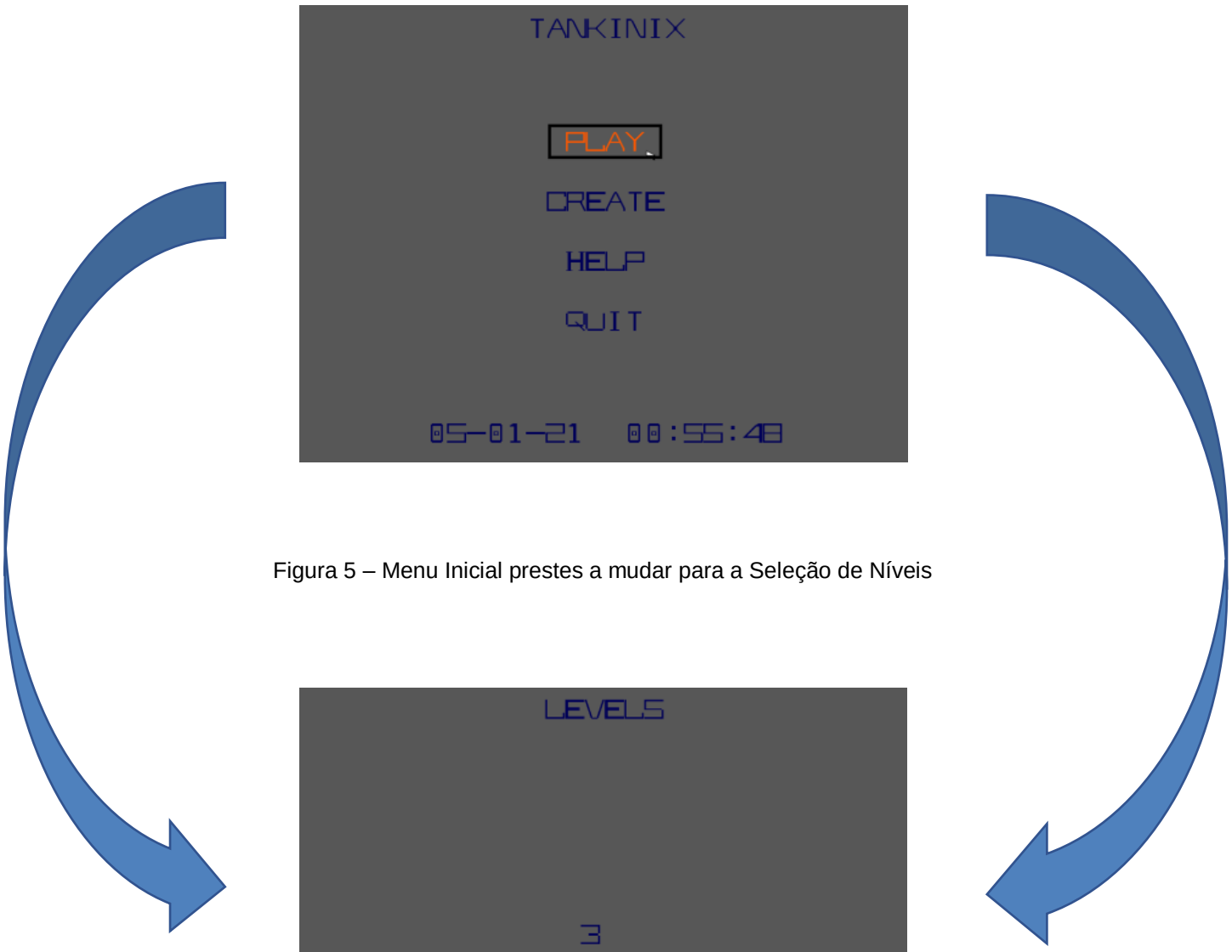


Figura 5 – Menu Inicial prestes a mudar para a Seleção de Níveis

Figura 6 – Menu de Seleção de Níveis

No menu de escolha de nível o utilizador pode escolher o nível que pretende jogar utilizando os botões de menos e mais presentes no ecrã (ou com as teclas “A” e “D” respetivamente) o nível varia entre 0 e 999. Se for selecionado um número para o qual não exista um nível, o jogador será encaminhado para um nível por defeito.

2.3. Nível

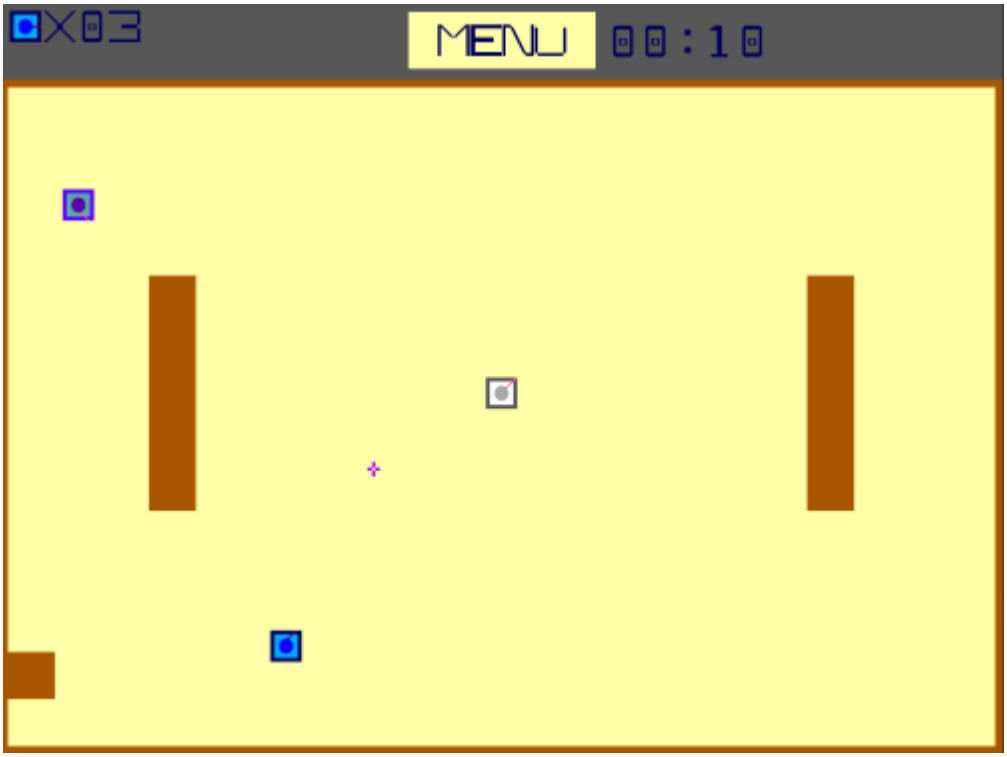


Figura 7 – Exemplo de um nível e da sua interface

Uma vez dentro do nível, o utilizador pode ver as vidas que tem no canto superior esquerdo, o contador do tempo do nível no centro à direita e por fim, também na parte superior do ecrã, tem o botão de pausa. Mal entra no nível o jogador assume o controle do tanque, através das teclas “W”, “A”, “S” e “D”. O apontador do rato transforma-se em mira e o jogador facilmente se apercebe que controla essa mesma mira com o “mouse”. Para disparar deve ser premido o botão esquerdo do rato. Se o contador do nível for crescente, o jogador deve eliminar todos os restantes tanques de modo a passar o nível, mas se o relógio estiver em contagem decrescente o jogador só tem de sobreviver de modo a completar o desafio.

3. Estado do Projeto

O projeto inclui todos os dispositivos disponíveis, à exceção da Serial Port. Tendo sido o ponto do projeto que ficou de facto a faltar. Também no menu inicial existe a opção-fantasma “Create” que seria um editor de níveis, relativamente simples de implementar, dado que conseguimos comprimir os níveis basicamente a matrizes. Porém esta e outras funcionalidades que queríamos implementar (como outros tanques inimigos e obstáculos) não são tanto do âmbito da unidade curricular, pelo que, devido à restrição de tempo, optamos por módulos mais relacionados com LCOM como o RTC, por exemplo.

3.1. Funcionalidades Implementadas

Dispositivo	Utilidade	Int
Timer	Controla a framerate e sincroniza animações	Sim
KBD	Movimentar o tanque e pequenos pormenores do navegação do menu	Sim
Mouse	Mira e navegação do menu. Disparar.	Sim
RTC	Visualização da data e hora	Sim
Video card	Ecrã	Não

Tabela 1 – Resumo das funcionalidades implementadas

3.2. Uso dos Dispositivos

Timer

O timer é utilizado para o controle do frame rate do jogo. Para esse controle utilizamos as interrupções do timer (usando a frequência de 60 interrupções por segundo). As animações também se regem pelo timer. Por fim, os contadores dentro dos níveis são naturalmente responsabilidade do timer.

Figura 8 – Contador do Nível

Teclado

O teclado é no geral usado para o movimento do tanque do jogador. São usadas as teclas “W”, “A”, “S” e “D” como é hábito nos videojogos de computador. O ESC pode ser usado para forçar o fecho do programa a qualquer momento (sem recurso a menus, como seria normal). No menu de seleção dos níveis, o “A” simula um clique no botão menos e o “D” um clique no botão mais.

Rato

O rato é talvez o periférico mais determinante no Tankinix. Desde logo, controla o cursor nos menus, através da deslocação. Também com recurso à deslocação controla a mira do tanque, com um alcance de 360º resulta numa movimentação suave e agradável. O botão esquerdo trata dos disparos realizados pelo tanque do jogador. A mira e o disparo estão relacionados com o mesmo dispositivo, resultando assim numa experiência consistente e intuitiva.

Figura 9 – Mira no seu estado pronto a disparar

Placa gráfica

Usamos o modo 0x105 com cor indexada. O projeto tem double buffer implementado. Utilizamos sprites e algumas funções a estes associados para detetar colisões entre paredes, bolas e tanques. No fim do jogo é utilizada a função “vg_exit()” para libertar a memória alocada para os buffers e para retornar ao modo de escrita default do Minix.

RTC

A função do RTC no projeto, reside na leitura da data e hora, leitura essa exibida no menu numa forma de fácil compreensão devido à utilização de uma struct “rtc_time” para organizar as várias variáveis (segundos, minutos … anos)

Figura 10 – Exibição da data e hora fornecida pelo RTC

4. Organização e Estrutura do Código

- Módulo keyboard
- Módulo mouse
- Módulo rtc
- Módulo timer
- Módulo video_gr
- Módulo utils
- Módulo sprites
- Módulo proj_aux

4.1. Módulos

Módulo Keyboard

Neste módulo encontram-se funções para manipulação direta do teclado (handler do keyboard, funções de subscrição de interrupções, funções de escrita e leitura em registos do kbc, …). Este módulo foi desenvolvido no decorrer do Laboratório 3, tendo sido realizado igualmente por ambos os membros.

Módulo Mouse

Neste módulo encontram-se funções para manipulação direta do mouse (handler do mouse, funções de subscrição de interrupções, funções de escrita e leitura em registos do kbc, …). Este módulo é igualmente responsável por juntar os 3 pacotes que constituem a informação fornecida pelo rato e garante a sincronização desta informação. Este módulo foi desenvolvido no decorrer do Laboratório 4, tendo sido realizado igualmente por ambos os membros.

Módulo RTC

Neste módulo encontram-se funções para manipulação direta do RTC (handler do RTC, funções de subscrição de interrupções, funções de escrita e leitura em registos do RTC, funções para programação de alarmes, …). Também se encontram neste módulo todas as constantes simbólicas para manipulação do RTC. Neste módulo as informações do RTC são guardadas e organizadas numa “human-friendly struct” para fácil acesso no restante projeto.

Módulo Video_Gr

Neste módulo encontram-se funções para manipulação direta dos buffers de vídeo (buffer e video_mem). Este módulo foi inicialmente realizado para o Laboratório 5, tendo sido completado igualmente por ambos os membros.

Módulo Timer

Neste módulo encontram-se funções para manipulação direta do timer (handler do timer, funções de subscrição de interrupções). Este módulo foi desenvolvido no decorrer do Laboratório 2, tendo sido realizado igualmente por ambos os membros.

Módulo Utils

Neste módulo encontram-se pequenas funções auxiliares, desenvolvidas quer no decorrer dos Laboratórios quer à medida que pequenas funcionalidades foram sendo necessárias para o projeto.

Módulo Sprites

Este módulo foi-nos parcialmente fornecido após o Laboratório 5, contém as structs relativas a Sprites e Sprites Animados. Contém os inicializadores dos mesmos, bem como as funções de "draw" para ambas as classes.

Módulo Proj_Aux

Este é o módulo principal do projeto. Contém as structs Aim, Ball, Tank, Player, Box, Clock, Level e Menu. Aqui também está centrado todo o interrupt handling, nas funções menu_ih e game_ih. Utiliza os restantes módulos e trabalha já a um nível mais alto. Todas as funções de colisões, algumas de desenho e os inicializadores das structs, entre outros também podem ser encontrados aqui

```
typedef enum {DEATHMATCH, SURVIVAL, GOAL} LevelType_t;
typedef enum {NO_C, CX, CY, NCX, NCY} Coordinate_t;
typedef enum {NO_COLISION, TANK_C, PLAYER_C, BALL_C, DESTRUCT_C, SOLID_C, REBOUND_CX, REBOUND_CY,
MINE_C} Colision_t;
typedef enum {FLOOR = 55, WALL = 20, PLAYER_TILE = 200, XTANK_U_TILE = 201, XTANK_D_TILE = 202,
YTANK_R_TILE = 203, YTANK_L_TILE = 204, RSTANK_TILE = 205, DSTANK_TILE = 206, ISTANK_TILE = 207}
Tile_t;
typedef enum {TIMER_INT, KBD_INT, MOUSE_INT} Interrupt_t;
typedef enum {XP, YP_XP, VP, YP_XN, XN, YN_XN, VN, YN_XP} Aim_Quadrant_t;
typedef enum {ALIVE, BOOM, DESTROYED} State_t;
typedef enum {TPLAYER, XTANK, YTANK, RANDOM_STATIC_TANK, DIRECT_STATIC_TANK, INDIRECT_STATIC_TANK}
TankType_t;
typedef enum {NO_FIRE, NORMAL_AIM, SHOOTING} AimStatus_t;
```

Figura 11 – Enumerações presentes no proj.aux

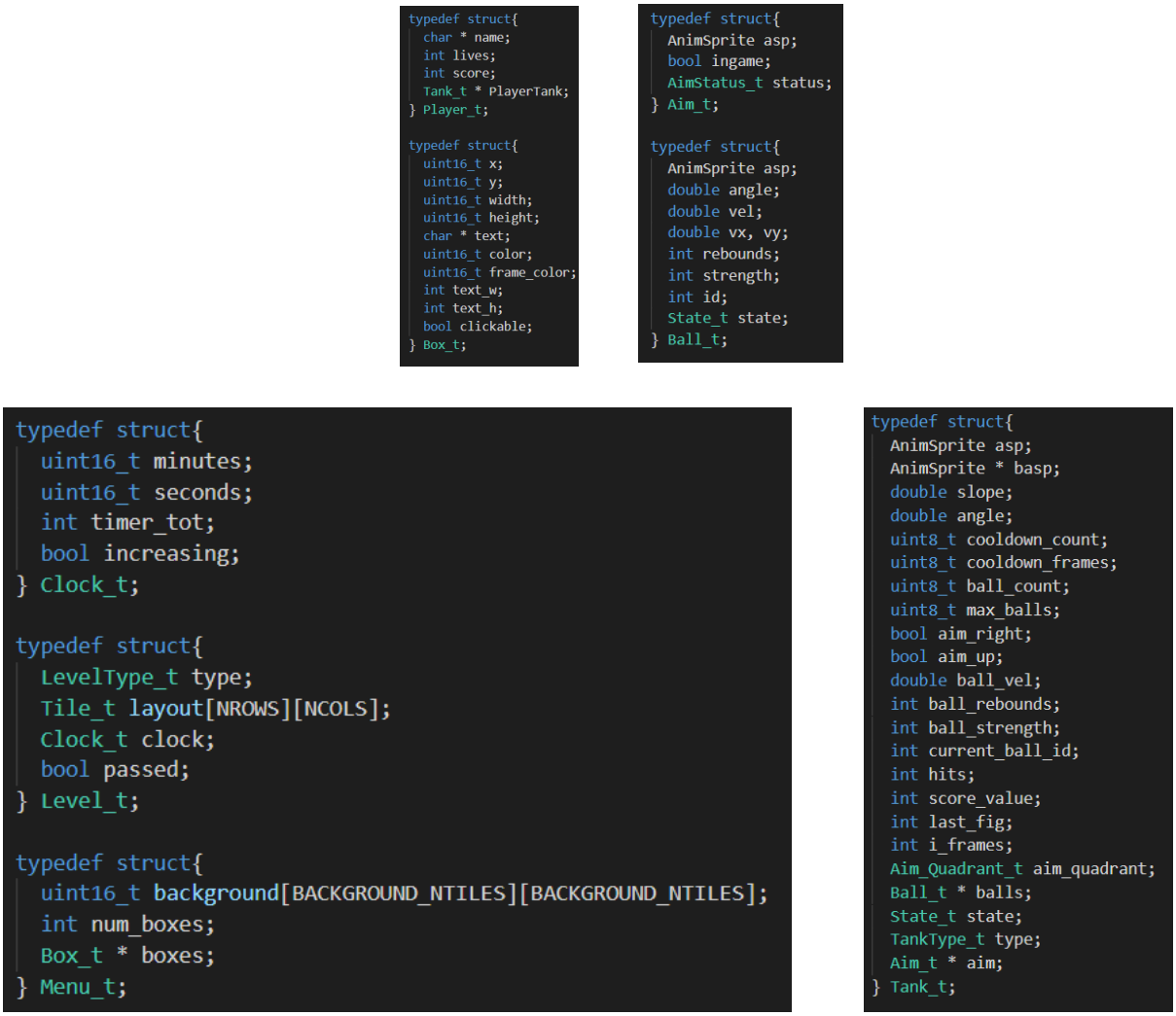


Figura 12 – Principais structs do Tankinix

Figura 13 – Documentação de funções do Mouse

4.2. Diagrama de Classes

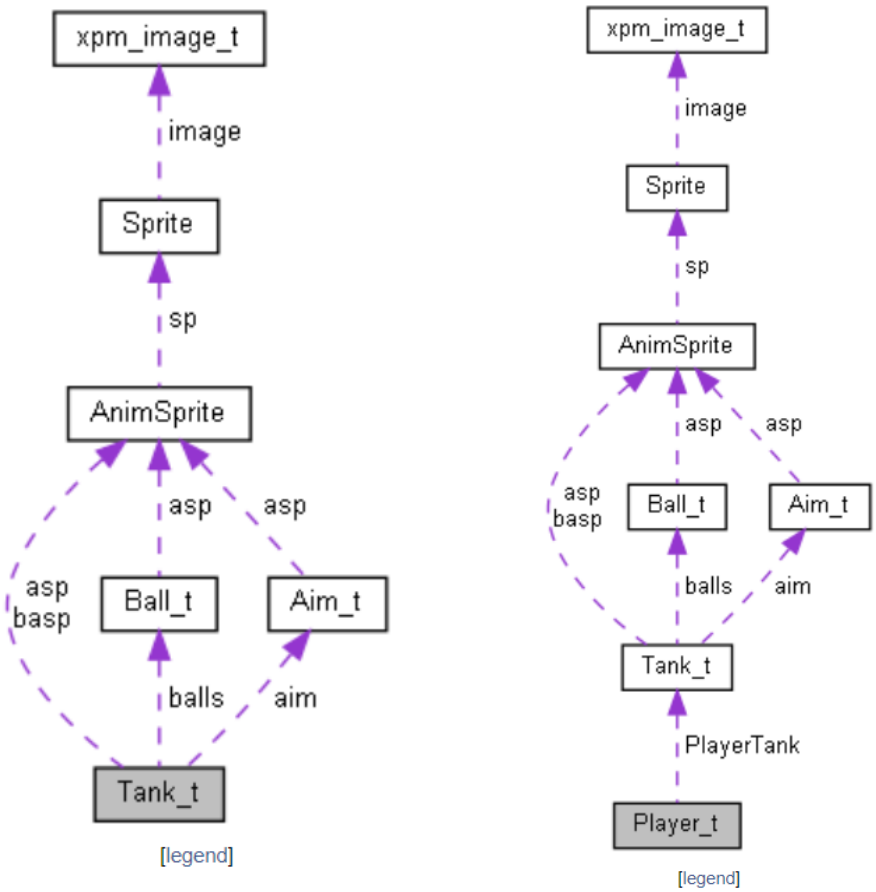


Figura 14 – Diagrama de colaboração para Tank_t e Player_t

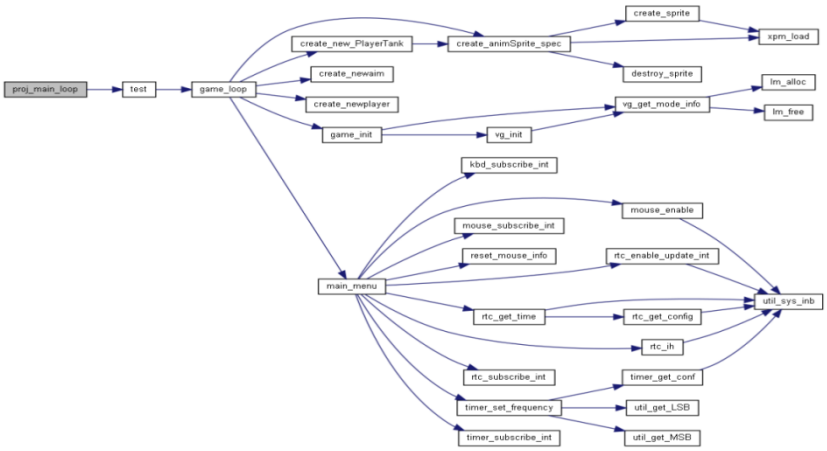


Figura 15 – Diagrama Loop Principal

5. Detalhes da Implementação

No nosso projeto, tal como no decorrer dos laboratórios, desenhamos as funções de forma a estas ficarem distribuídas por várias camadas, de modo ao código ficar mais bem estruturado e organizado e também a facilitar a construção do projeto: uma “layer” de

funções que interage mais com os periféricos, estando este código, geralmente nos ficheiro com o nome do periférico desenvolvidos nos Laboratórios.

O nosso jogo é também fortemente orientado a objetos , sendo todos os elementos

instâncias de uma classe/struct. Todos os objetos têm construtor e funções para desenho dos mesmos. Alguns exemplos de classes usadas são as classes Tank, Ball, Clock, Box, Menu, Player e Level.

Parte da piada do Tankinix está nas colisões e na forma como estas são tratadas. Os ricochetes da bola exigiram algum conhecimento matemático (vetores e outros…), resultando numa sensação final agradável. Os sprites quer dos tanques, quer das bolas são quadrados facilitando depois as funções de colisão que assim só necessitam de trabalhar com o x, y, width e heigth dos respetivos sprites. Todas as bolas são guardadas num array no tanque que as disparou facilitando a iteração por todas estas.

```
Tile_t level_4[14][21]= {
  { F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F},
  { F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F},
  { F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F, RS,  F,  F},
  { F, RS,  F,  F,  W,  F,  F, DS,  F,  F,  W,  F,  F,  F,  F, XU,  W,  F,  F,  F,  F},
  { F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F},
  { F,  F,  F, XD,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F},
  { F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F, DS,  F},
  { F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F,  F},
  { F,  F,  W,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F},
  { F,  F,  W,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F},
  { F,  F,  W,  F,  F,  F,  F,  W,  F,  F, XD,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F},
  { F,  F,  W,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F, DS,  F,  F,  F,  W,  F},
  { F,  F,  W,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F},
  { P,  F,  W,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F,  F,  F,  F,  F,  W,  F}
};
```

Figura 16 – Exemplo de Matriz de um Nível

Uma das melhores decisões do projeto foi a de guardar os layouts dos níveis em matrizes de tamanho constante. Isto permitiu que a criação de níveis fosse muito simplificada, tanto que agora após a entrega facilmente podemos em pouco tempo criar dezenas de níveis interessantes. Outro aspeto muito positivo desta opção está relacionado com as colisões que podem ser verificadas simplesmente através dos 4 cantos do objeto.

Por fim, surgiram alguns problemas que não existiram nos labs, por exemplo a sincronização do rato sempre que a ação deste era interrompida, desde logo quando o jogador perdia uma vida e o nível tinha de reiniciar. Nesse caso resolvemos o problema com recurso ao terceiro bit do primeiro byte do packet (tem de ser 1) e situações como esta obrigaram-nos a rever os Lab Handouts e a matéria no geral.

6. Conclusão

A cadeira de LCOM foi, sem dúvida, a cadeira mais difícil, desafiante e exigente que já tivemos até agora. A carga de trabalho é maior de que nas outras UCs e os próprios conteúdos mais complexos mas, por outro lado, foi na nossa opinião a cadeira em que mais contactamos com programação computacional e em que podemos dar asas à criatividade com a hipótese de criar algo único com o nosso projeto.