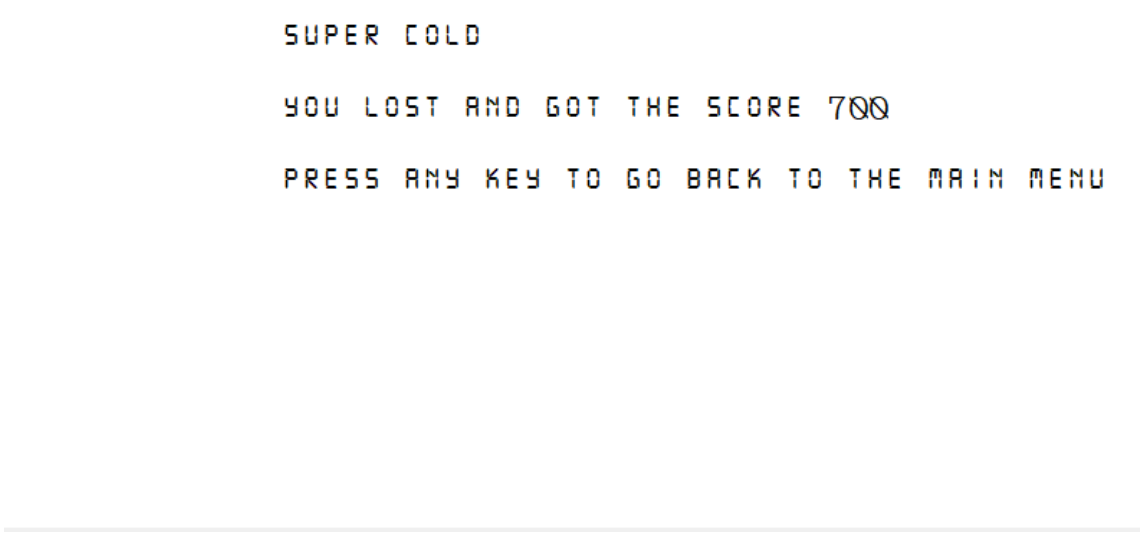
Realizado pelo grupo 9 da turma 2:

Pedro Pinheiro – up201906788

Bruno Gomes – up201906401



Índice

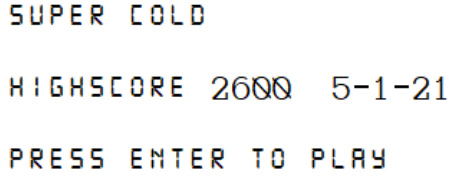
Introdução

No âmbito da unidade curricular de Laboratório de Computadores, propusemos como projeto final elaborar um jogo, de forma a possibilitar a demonstração do conhecimento que adquirimos ao longo do semestre. O jogo que criámos, Super Cold, é uma shooter com vista top-down, em que o tempo abranda quando o utilizador não move a sua personagem, inspirado no conhecido jogo Superhot. O utilizador controla uma personagem que pode disparar projéteis e é atacado por inimigos, que também disparam, o objetivo é eliminar os adversários sem ser atingido. Os inimigos aparecem a cada 30 segundos. Para além disto, o jogo tem um sistema de pontos, que são atribuídos ao jogador e a melhor pontuação e o dia em que foi atingida são apresentados no menu inicial.

1. Instruções de utilização

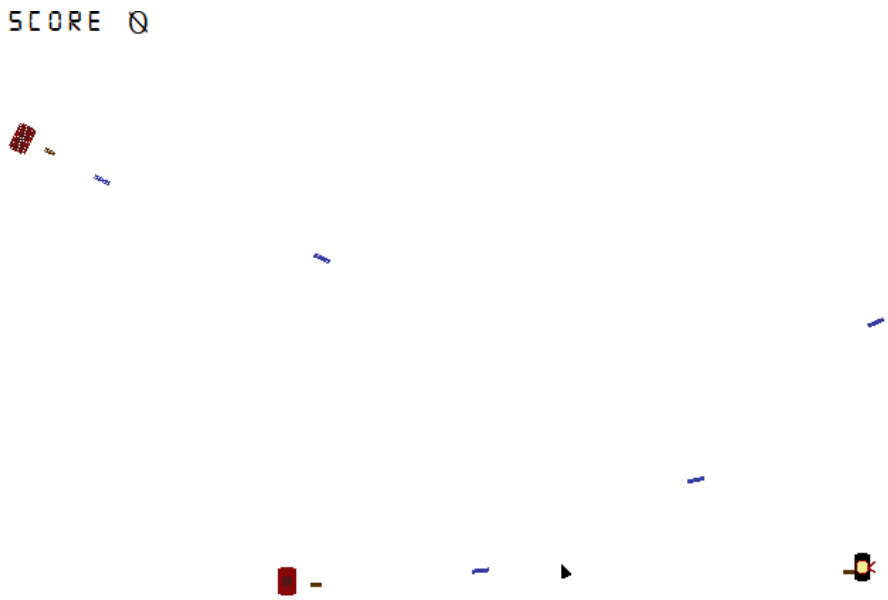
Menu inicial

O jogo começa por apresentar a pontuação mais alta, bem como a data em que foi atingida. Para além desta, mostra “Press ENTER to play”, ou seja, o jogador deve pressionar a tecla ENTER para começar a jogar.



Jogar

Caso o utilizador pressione ENTER, é direcionado para o ambiente de jogo. Para mover o personagem, deve utilizar as teclas W A S D, que movem para cima, esquerda, baixo e direita, respetivamente. Para diminuir a velocidade das entidades, basta não mover o personagem. O utilizador pode ainda apontar, movendo a mira com o rato, o que causará que o seu personagem rode, ficando virado para a posição da mira. Para disparar basta pressionar o botão esquerdo do rato, e um projétil será disparado desde o personagem na direção atual da mira. Os inimigos, que vão aparecendo periodicamente, também disparam contra o jogador, e se um projétil colidir com um inimigo, a pontuação aumenta e o inimigo é eliminado. Se o jogador for atingido por um projétil adversário, a pontuação é guardada se for das 3 melhores até á data, e o programa volta para o menu inicial após o utilizador clicar numa tecla qualquer. A tecla ESC pode ser premida para sair do jogo a qualquer momento.



2. Estado do projeto

Concluímos com sucesso todos os objetivos que foram autopropostos.

Dispositivo	Funcionalidade	Interrupções
Timer	Animações	Y
Keyboard	Controlo do movimento do personagem e começar ou terminar de jogar	Y
Mouse	Mover a mira e disparar	Y
Video card	Menu inicial, fundo, personagem, inimigos, armas, explosões e projeteis	N
RTC	Leitura da data para apresentar nas pontuações e alarmes para introduzir novos inimigos	Y

Video card

O modo gráfico usado é o 0x115, que tem a resolução 800x600, onde há 24 bits por pixel e as cores são introduzidas diretamente. Utilizámos page flipping.

Foram usadas duas funções vbe:

- 0x02(na função activateGraphicsMode(mode) para inicializar o modo gráfico 0x115
- 0x07(na função applyChanges()) para o page flipping.

Keyboard

O teclado é utilizado para começar a jogar, sair do jogo e mover a personagem do jogador. Em todos os casos são usadas interrupções.

Mouse

O rato é utilizado para mover a mira e disparar, através do movimento do rato e do botão esquerdo do esquerdo do rato.

Timer

O timer é utilizado nas animações, para mover as entidades, para calcular o tempo real do jogo e para detetar colisões.

RTC

O relógio interno é utilizado para obter a data atual, quando há uma pontuação máxima nova e para atualizar os movimentos dos inimigos a partir de periodic interrupts.

3. Organização e estrutura do código

enemy.c

Neste módulo está implementada a criação dos inimigos, com a sua velocidade, uma entre duas armas e um entre dois sprites. Também está implementado o seu movimento, disparos e parte da sua rotação.
Percentagem do módulo: 8%

game.c

Um dos módulos principais do projeto. É responsável por criar as condições em que o jogo se inicia, dar load de todos os xpm's, criar e mover e remover as diferentes entidades do jogo, atualizando os seus sprites caso seja necessário, receber e tratar parcialmente os interrupts do teclado, dar load e save da pontuação mais alta, aumentar gradualmente a dificuldade do jogo, lidar com as colisões entre as diferentes entidades e ainda tem um papel no tratamento de interrupts recebidos do rato.
Percentagem do módulo: 20%

graphics.c

Este módulo é responsável pelas funções que desenham os sprites, inicializam o modo gráfico e pelo page flipping. Parte dele foi criada durante as aulas
Percentagem do módulo: 10%

gun.c

Neste módulo estão implementadas as animações das armas, bem como a velocidade e cadência de disparo, que difere nas duas armas, parte da rotação, e direcionamento do disparo.
Percentagem do módulo: 5%

kbd.c

Neste módulo, desenvolvido nas aulas práticas, encontram-se funções necessárias á manipulação do teclado, através dos registos. É útil para outros módulos, nomeadamente player.c e proj.c.
Percentagem do módulo: 5%

letters.c

Este módulo é responsável por dar load do xpm de qualquer letra, número ou traço (-) pretendido.
Percentagem do módulo: 2%

linkedEntities.c

Neste módulo, e na respetiva header file, foi implementada uma estrutura de dados, Double Linked List. Cada node representa uma entidade, com um apontador void para a entidade, e guarda pointers para o node anterior e posterior. Também inclui as funções para adicionar e remover entidades.
Percentagem do módulo: 5%

mouse.c

Neste módulo, desenvolvido no âmbito das aulas laboratoriais, encontram-se funções necessárias para obter os data packets do mouse, que iram ser processados no game.c, uteis para move a mira do personagem do utilizador e para a sua rotação.
Percentagem do módulo: 5%

player.c

Neste módulo, está implementado o tratamento dos inputs do mouse/keyboard, assim como o movimento e as animações do player.
Percentagem do módulo: 10%

proj.c

Neste módulo, inicializa o modo da gráfica, inicia o jogo, dá subscribe e unsubscribe dos interrupts necessários, ativa os periodic interrupts do RTC e altera o sample rate do mouse.
Percentagem do módulo: 10%

rtc.c

Este módulo permite obter a data atual a partir do rtc, assim como criar, dar disable e ler interrupts periódicos. Ainda tem uma função para criar alarmes, que não foi usada.
Percentagem do módulo: 5%

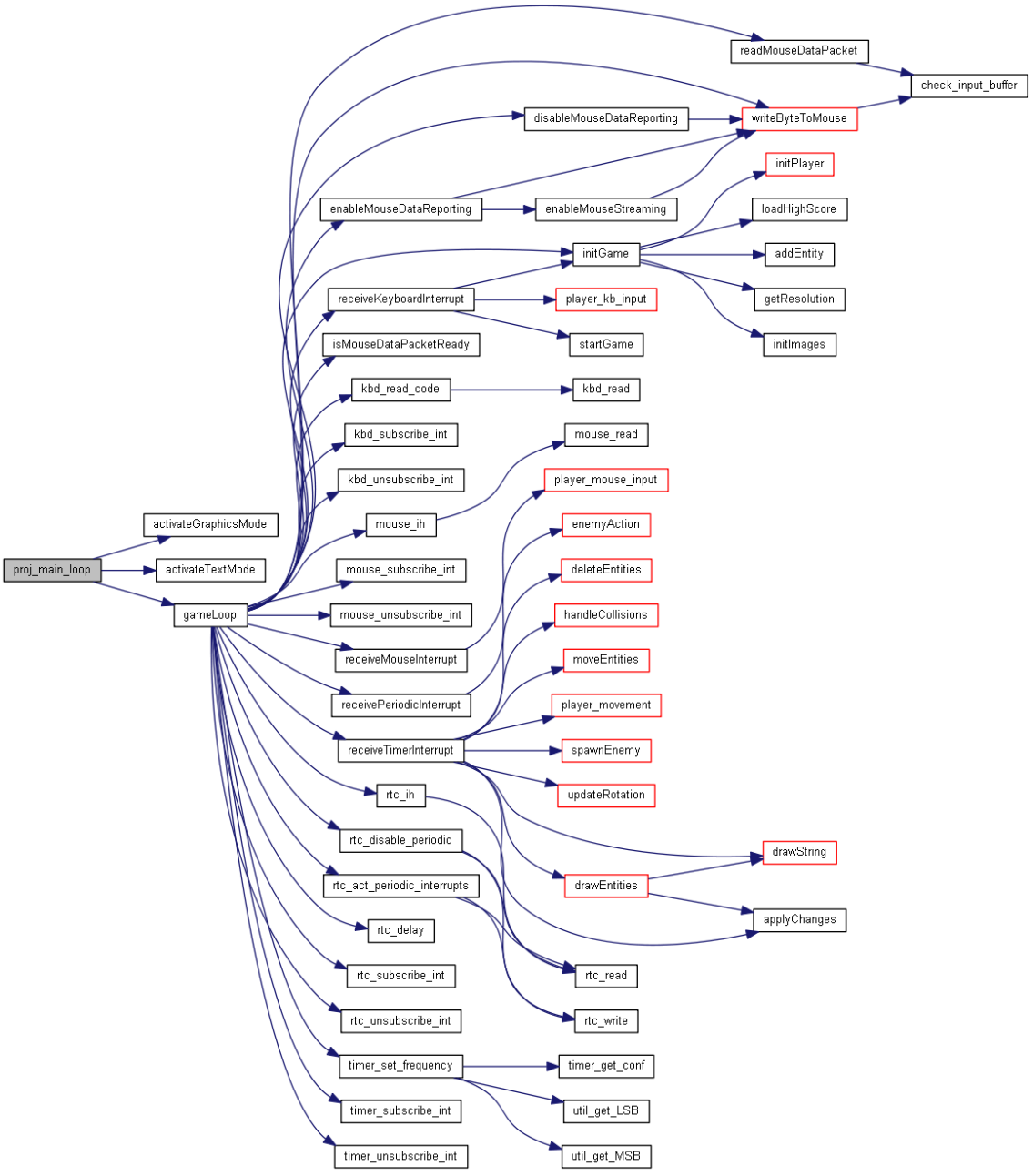
sprite.c

Neste módulo as funções conseguem criar sprites, fazer a sua rotação (mais

informação nos detalhes da implementação), verificar a ocorrência de colisões e fazer com que entidades, neste caso a personagem do utilizador e o cursor, não se movam para fora do ecrã.
Percentagem do módulo: 10%

timer.c

Neste módulo, desenvolvido no âmbito das aulas laboratoriais, encontram-se funções necessárias para a manipulação do timer, através dos registos.
Percentagem do módulo: 5%



4. Detalhes da implementação

- Double Linked List – Estrutura de dados para guardar todas as entidades do jogo, escolhemos utilizar esta estrutura pois a eliminação e inserção de entidades seria de complexidade O(1), o que melhora bastante a performance. Em cada node, é possível saber o tipo de entidade a partir do enumerator type que pode tomar um dos seguintes valores: PLAYER, ENEMY, BULLET, GUN e SPRITE.

- Rotação – Para permitir disparar a 360º, foi necessário implementar rotações nos sprites, para os movimentos não ficarem estranhos. A implementação consiste em adicionar um nova variável à sprite, rotation, que representa a rotação do sprite em radianos. Para desenhar a sprite rodada, na função drawSprite (graphics.c), é chamada a função getRotationCoordinates (sprite.c) , que consite em transformar a posição de cada pixel em coordenadas polares, adiciona a rotação do sprite, e volta a transformar em coordenadas cartesianas, com o objetivo de obter a posição do pixel na sprite rodada.
  Foi utilizada a rotação nos inimigos, no player, e nos projeteis.

- Sistema de highscore – O highscore atual é guardado e carregado no ficheiro score.txt, sendo a data atual obtida através do rtc. O highscore é atualizado se, quando o player é atingido, o score do jogo é maior que o highscore atual.

- Incremento de dificuldade – A dificuldade do jogo é incrementada a cada 2 spawns de inimigos, que consiste em diminuir o tempo entre cada spawn, até um certo limite. Esse limite é atingido após 24 spawns.

- Comportamento dos bots – De modo a que os bots

ataquem o jogador e o sigam, foi utilizado periodic interrupts do RTC. A cada interrupt, atualiza o movimento e dispara, se for possível.

- Dois tipos de inimigos – A cada spawn, tem 50% de probabilidade de o inimigo ter uma SMG, ou uma pistola, que diferem no fire rate.

5. Conclusão

Como grupo consideramos que, sendo os conteúdos lecionados em LCOM úteis e interessantes, nomeadamente o projeto, o tempo exigido pela unidade curricular é bastante elevado, quando temos em consideração o número de créditos que lhe são atribuídos. Gostávamos de ter tido aulas presenciais todas as semanas, mas entendemos e respeitamos as restrições. Seria interessante ter implementações das funções ligadas aos dispositivos curadas pelos professores, de modo a entender o modo ideal de os operar.