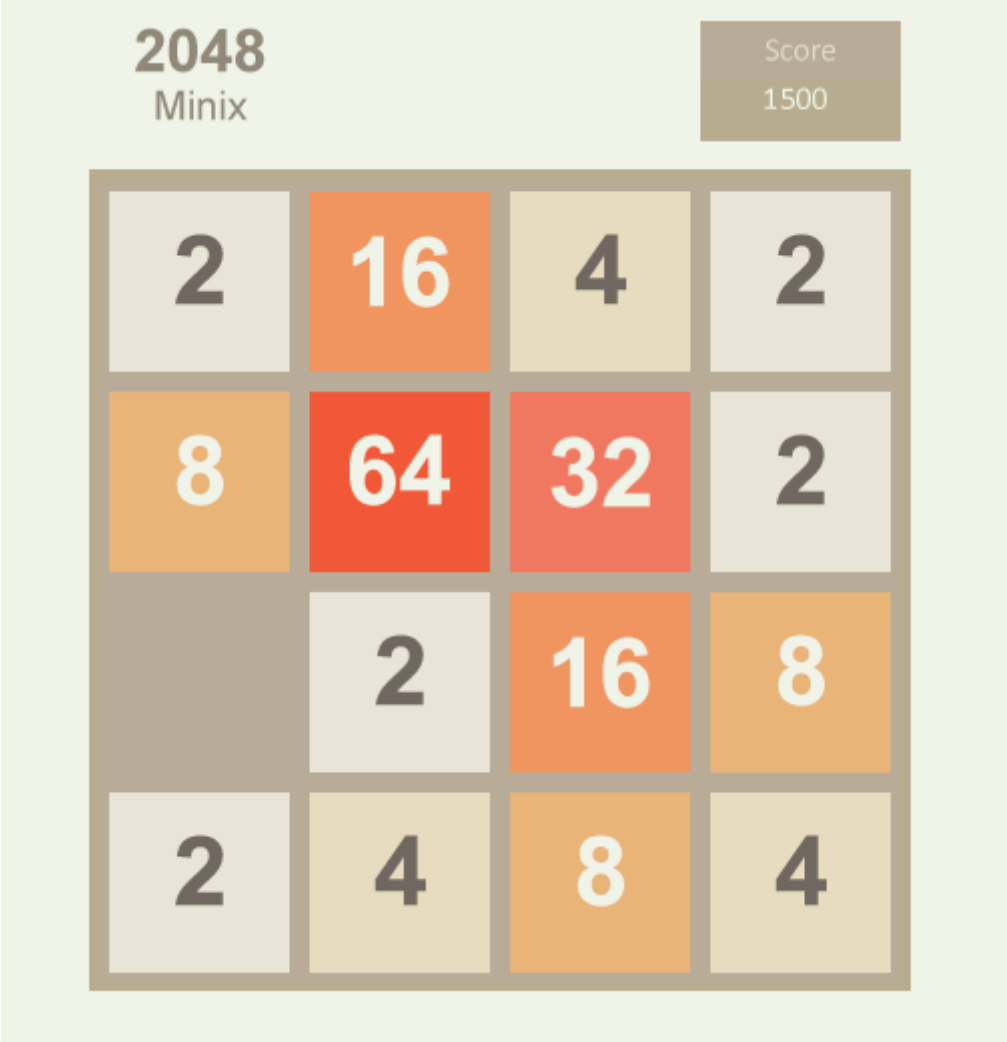
EIC0020 - Laboratório de Computadores

2015/2016 1º Semestre

2048 Minix



Turma 2, Grupo 10

Gonçalo Ribeiro - up201403877 - goncalo.ribeiro@fe.up.pt

Índice

Resumo

O projecto 2048 Minix, tráta-se de uma adaptação para o sistema operativo Minix 3, do famoso jogo viral 2048 - desenvolvido por Gabriele Cirulli no início de 2014.

O jogo consiste numa matriz 4x4 onde números são gerados automaticamente. O jogador pode arrastar os números dessa matriz em quatro direções diferentes: cima, baixo, esquerda, direita. Ao serem arrastados, os número iguais que colidem são fundidos, tornando-se no número correspondente à soma de ambos. A cada jogada (arrastamento) é gerado um novo número na matriz. O objetivo é obter o número 2048 sem que a matriz atinja um estado onde não existam mais espaços para se gerar números e não hajam número iguais adjacentes para se realizarem somas.

O propósito deste projecto, desenvolvido no âmbito da cadeira de LCOM, foi enriquecer o nosso conhecimento relativo ao funcionamento em baixo nivel de alguns componentes e periféricos de computador. Foi necessário programar uma “interface” de comunicação entre o kernel do Minix e o hardware acima referido.

Os perifericos usados neste projecto são:

- Teclado, através de interrupções, para jogar;
- Rato, também para jogar;
- Timer, para atualizar a imagem do display;
- Placa gráfica em modo de vídeo, produzindo a interface do jogo.

O software utilizado foi:

- Minix 3;
- Eclipse, foi o IDE utilizado;
- Redmine, utilizado para repositório e gestão de versões (Subversion);
- Virtual Box, para correr a imagem do Minix virtualmente;
- Adobe Photoshop, para desenhar manualmente as imagens BMP;

Descrição

Ao iniciar o jogo, é apresentado ao utilizador uma interface que contém a matrix de jogo já com um número. O jogador pode então utilizar as setas do teclado para deslizar o número previamente gerado para cima, baixo, esquerda ou direita. Alternativamente, o jogador também pode utilizar o rato, premindo o botão esquero e arrastando o rato na direção pretendida.

Sempre que o jogador faz as peças deslizar, é gerado um novo número na matriz: 90% de chance de ser um 2 e 10% de ser um 4. À medida que o utilizador vai somando números identicos, será possível ver no canto superior direito a sua pontuação atualizada a cada jogada.

O jogo termina automaticamente caso não existam mais arrastamentos possíveis na matrix, isto é, a matrix encontra-se cheia e não existem dois números iguais e adjacentes para se somarem. O utilizador pode abandonar o jogo a qualquer altura premindo a tecla ESC.

Implementação

Módulos Implementados

- main: Inicialização e destrução de um objeto game;
- 2048: Implementação da classe game e da estrutura coordinates;
- graphics: Contem funções básicas de desenho no display em modo gráfico;
- timer: Contém a declaração da classe timer
- keyboard: Contém a declaração da classe mouse
- mouse: Contém a declaração da classe mouse
- bmp: contem as funções de leitura e reprodução de imagens .bmp

Classes Implementadas

- game: Contem a máquina de estados, é responsável por ler os inputs do do utilizador, processá-los e atualizar o jogo;
- timer: Abstração de um timer;
- keyboard: Abstação de um teclado;
- mouse: Abstração de um rato.

Estruturas de Dados

- coordinates: Estrutura que contem um par de coordenada
- bmp: Estrutura que permite carregar, visualizar e destruir bitmaps.

Aspectos Relevantes

Neste projeto decidi concentrar cada periférico numa classe. Desta forma posso-me abstrair de subscrições e configurações (como por exemplo, mudar frequencia do timer ou colocar o rato em modo de stream de packets), implementando-as nos contrutores e destrutores.

O módulo dos bitmaps (bmp.h e bmp.c) é uma versão ligeiramente modificada do código cedido pelo colega de curso Henrique Ferrolho (https://github.com/ferrolho), que se baseou na seguinte thread pública para o desenvolver: http://forums.fedoraforum.org/showthread.php?t=171389. Eu alterei algumas das funções de visualização de bmp's de forma a poder escolher, por parametro, em que buffer colocar o bitmap.

Limitações/ Bugs

Apenas existe um bug conhecido: quando o utilizador alterna rápidamente entre a utilização do rato e do teclado, o rato "descontrola-se" durante uns momentos (1 a 2 sgundos), mas depois regressando ao normal.

Funcionalidades

Gráficos

Os gráficos desenvolvidos neste projecto beseiam-se principalmente na reprodução de bitmaps.

A classe game (onde se processa a mudança de estados) contém dois arrays de bitmaps: tiles (que são os quadrados númerados na matrix) e numbers (números que aparecem na pontuação).

Este projeto tem implementado um buffer duplo. Como o deslize dos número não é animado (eles simplesmente desaparecem e aparecem no local correcto), os gráficos são atualizados a cada vez que se verifica um arrastamento. Então, foi adicionado um duplo buffer para ser possível reproduzir a uma frequência constante o ponteiro do rato. A cada tick do timer, o conteúdo buffer onde são atualizadas as posições dos números e a pontuação é copiado para o segundo buffer. Neste segundo buffer é adicionado o ponteiro do rato na sua posição mais recente e esta imagem final é depois transferida para a memória da placa gráfica.

Periféricos

Todos os periféricos funcionam com interrupções. As interrupções são processadas dentro da classe game.

O modo de placa gráfica utilizado é o 0x114, com resolução 800x600 e cores 64 mil cores.

Instruções

Após o checkout do programa, basta executar o comando sh install.sh (que instala o ficheiro de configuração na pasta /etc/system.conf.d e compila o programa) e o comando sh run.sh (para executar o binário).

Conclusão e Autoavaliação

Em retrospectiva, este projeto, embora simples, apresentou-me bastantes obstáculos ao longo do seu desenvolvimento.

A lógica do jogo foi relativamente acessível, assim como a implementação do teclado. Rápidamente consegui desenvolver uma versão funcional (mas rudimentar) do projeto. A grande dificuldade esteve na integração de um rato funcional e responsívo, que implicou a adição de um buffer adicional e à restruturação do funcionamento das funções gráficas.

Acabei por não implementar, na totalidade, todos os módulos propostos por mim e pelo antigo colega, que acabou por abandonar o grupo no inicio do desenvolvimento deste projeto. No entanto, acho que o produto final, embora “minimalista”, consegue deter em si a essência do que nos foi proposto para este trabalho final: um programa funcional, baseado no paradigma da comunicação de baixo nivel entre o hardware e o software.