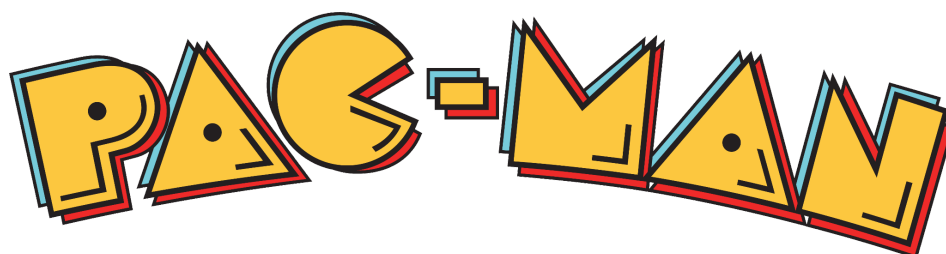


U. PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO



Diogo Rodrigues - up202103629

Diogo Câmara - up201905166

Mário Ferreira - up201907727

Pedro Moreira - up201904642

Licenciatura em Engenharia Informática e de Computação

Laboratório de computadores

Turma 1 - Grupo 08

13 de Junho de 2022

Índice

User Instructions	3
1.0 Main Menu	3
1.1 High Scores Menu	4
1.2 Game	5
1.3 Quit	6
Project Status	6
Timer	6
Functions used from the lab	7
Functions that use the timer	7
Keyboard	7
Functions used from the lab	7
Functions that use the keyboard	7
Mouse	7
Functions used from the lab	8
Functions that use the mouse	8
Video Card	8
Functions used from the lab	8
Functions that use the video card	8
Code Structure	9
Modules	9
Timer	9
Keyboard	9
Mouse	9
Graphics Card	9
Game	9
User Interface elements	9
Implementation Details	10
Collision	10
Call Graph	10
Conclusion	10

User Instructions

Main Menu

After running the command `lcom_run proj`, minix will display our Main Menu, which the user can use to select several options using both the keyboard or the mouse. Using the keyboard, pressing the arrow keys, the “a” or the “d” keys results in the selected button changing. Using the mouse, hovering over one of the buttons also changes the selected button. Pressing the “Enter” key or clicking the mouse left button while hovering over one of the buttons selects it.



1.Play - Takes the user from Main Menu to the Main Game.

2.Score - Opens a High Score Menu where it shows the top 5 highest scores of the game.

3.Quit - Quits the Menu and goes back to minix shell

High Scores Menu

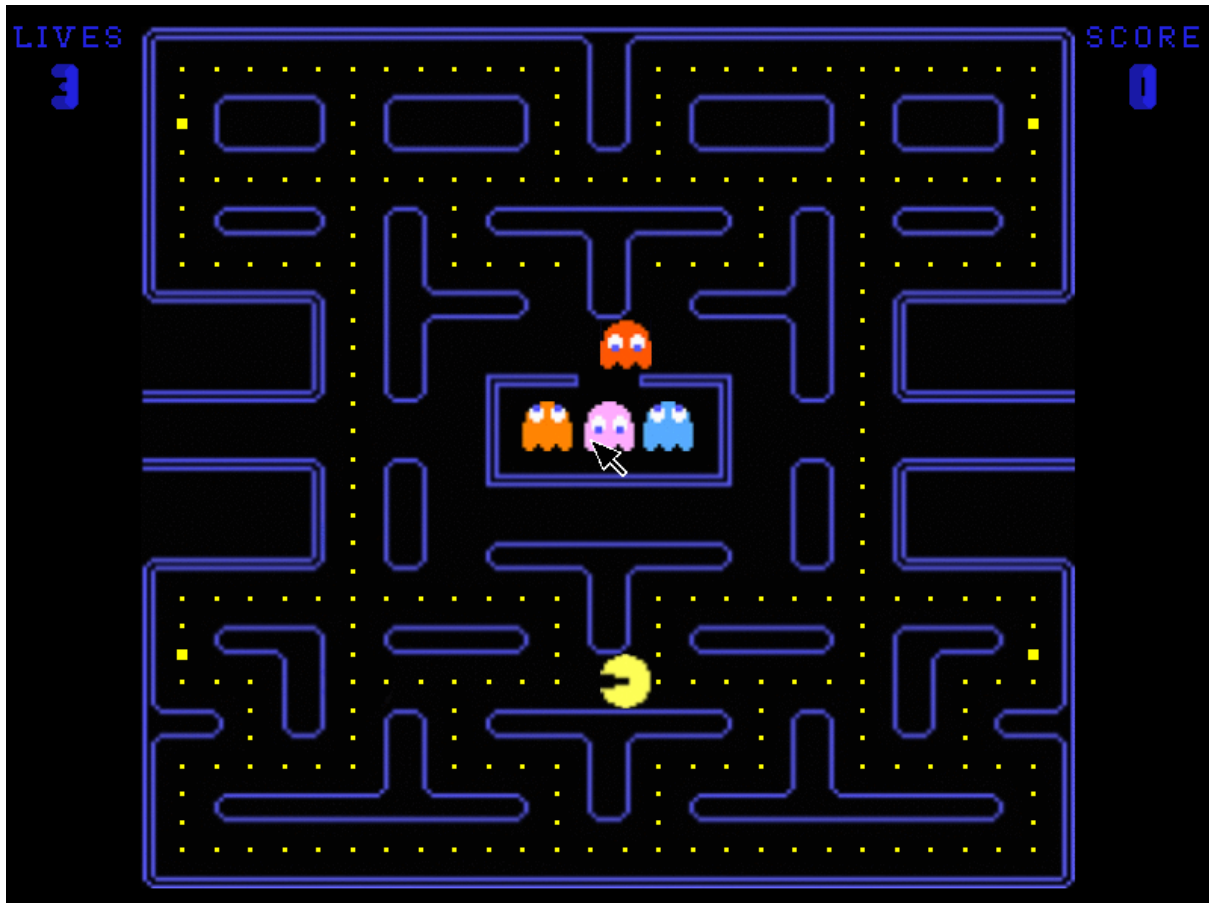
This Menu presents the user with the top five scores of the current game session, in descending order. The user can press “Enter” or click the mouse left button while hovering over the “Back” button to return to the Main Menu.



1.Back - Takes the user from the High Scores Menu to the Main Menu.

Game

After pressing the Play button on the main menu we are taken to the main game screen. The game is found in a “frozen” state and will start as soon as the player presses any of the movement buttons.






During the game, the player can move his pacman with the keyboard keys “w”, “s”, “a” and “d” .

The objective of the game is for the player to catch all the coins in the game board while avoiding colliding with the ghosts. If the player catches a “bigger” coin, power up, it enables the player to eat the ghosts for a duration of 30 seconds.

The player can press “q” if he wishes to leave the game and go back to the menu.

Features :

	Number of lives of the player.
	Current score of the player.
	Power up “coin” that allows player to kill ghosts.

Quit

After pressing this button, the program closes and returns Minix to text mode. Similarly to the other buttons, the user can use the keyboard or the mouse to select and press the button.

Project Status

Device	What for
Timer	Control the frame rate Control the time the ghosts take to come out of the center Control the time the ghosts take to reappear after being eaten by pacman
Keyboard	Control pacman Change menu selection
Mouse	Menu selection Highscore menu selection
Video Card	Initialize and exit video mode Draw the game

Timer

The timer interrupts were used to implement the drawing of all the moving elements of the game, control the frame rate and the time ghosts spend either in the center or dead.

Functions used from the lab

- timer_int_handler();
- timer_set_frequency();

Functions that use the timer

- pickup_coins();
- GP_collisions();
- reset_pacman_ghost_values();
- collision();
- ghost_movement();
- draw_background();
- draw_coins();
- draw_pacman();
- draw_ghost();
- draw_score();
- draw_lives();
- draw_mouse();
- draw_menu();
- draw_scores_menu();

Keyboard

The keyboard interrupts were used to get player input in the menu and highscore menu or control pacman. It's also used to quit the game using "ESC" or go to the menu using "Q".

Functions used from the lab

- kbc_ih();

Functions that use the keyboard

- reset_board();
- draw_background();
- draw_coin();
- draw_pacman();
- draw_score();
- draw_lives();

Mouse

The mouse interrupts were used to get player input in both the menu and the highscore menu.

Functions used from the lab

- mouse_ih();
- assemble_mouse_packet();
- parse_mouse_packet();

Functions that use the mouse

- game_mouse_hover();
- game_mouse_click();

Video Card

The video card was to get Minix on the video mode 0x115 (800x600 with 24 bits direct color). We also used a video memory buffer in order to make the drawing of xpm's smoother.

Functions used from the lab

- vg_init();
- vg_exit();
- move_buffer();
- vg_draw_xpm();

Functions that use the video card

- draw_lives();
- draw_score();
- draw_background();
- draw_coins();
- draw_mouse();
- draw_menu();
- draw_scores_menu();
- draw_number();

Code Structure

Modules

Timer

This module has the functionalities developed for lab2. It can mainly subscribe to one timer interrupts and change the frequency of a timer.

Keyboard

This module has the same functionalities developed for lab3. Its main functions are the `kbc_ih()` and `kbc_assemble_scan_code()`, which are responsible for reading a scancode from the keyboard controller output register and assembling and printing the scancode, useful in case it is a 2 bytes scancode.

Mouse

Similarly, this module has the same functions developed for lab4. It is mainly responsible for building the mouse packet.

Graphics Card

It has similar functions to the ones developed for the lab5, with minor changes to some functions implementation and names. The main function of this module is `vg_draw_xpm()`, which is responsible for drawing a xpm to the video buffer.

Game

Deals with all the game logic, initializing the graphic mode and all the necessary variables and subscribing to the devices interrupts. This modules is also responsible for receiving the interrupts and

User Interface elements

The UI uses the function `xpm_load()` to load the several sprites that will be needed throughout the application.

Implementation Details

Collision

In order to implement collisions, we first had to create a function that was able to convert the pacman or ghost position on the screen to a position on the game board matrix. After this was done, we were able to easily check for collisions between pacman, ghosts and walls.

Call Graph

Conclusion

With this course and specifically with the labs and mainly the project, we got a better understanding of programming on C and this language features such structs and enums, as well as programming some low level devices(device drivers) and usage of kernel functions .