

**Analog Pong Game**

CS132 Coursework – 2

LED Panel

Aaditya Nair

Supervisor: Prof. Richard Kirk

Department of Computer Science

University of Warwick

2023- 2024

Acknowledgments

We would like to express our deep and sincere gratitude to those who played a pivotal role in the successful culmination of this project. Their dedication, expertise, and support were invaluable throughout the entire process.

* Professor Richard Kirk for his invaluable guidance and mentorship throughout the entire process
* Seniors and mentors for their guidance and assistance with the code and information necessary for the completion of this project.
* Professor Feng Hao, whose wisdom, encouragement, and constructive feedback were instrumental in my personal and professional growth throughout the project.
* The University of Warwick, for providing a conducive environment for learning and research, and for the resources that supported the execution of this project.

I am deeply appreciative of the opportunity to work with such a talented and committed group. Their collective efforts created a positive and impactful project experience.

CONTENTS

Acknowledgments 1

Contents 2

Introduction 3

Timeline 4

Implementation 6

Mind map 9

Result 11

Conclusion 13

References 14

INTRODUCTION

In the realm of technology and innovation, the convergence of creativity and functionality often leads to groundbreaking projects This project draws inspiration from the timeless allure of ping pong, channelling it into a dynamic and visually striking experience through the integration of LED panels. The Pong game, a global sensation since its inception in 1972, continues to captivate audiences with its timeless appeal, offering an enduring experience of immersive and enjoyable gameplay. This iconic arcade game brought two-dimensional paddles and a bouncing ball to life on cathode-ray tube screens, however in this implementation we use a different approach.

In this project, we have been tasked with designing a two-player pong game using an LED Panel (32 pixels high and 16 pixels long) as well as STM32 controllers and joysticks. Using the course material, guidance from staff and tutors, as well as our own knowledge on C programming and gaming algorithms we have been able to design the final system. Given the time provided to complete the project we have been able to design the interface but we have had a few issues with connecting the joysticks to the panel. Over the Christmas break we both have tried collaborating and added a few extra functions to display the winner of the game.





IMPLEMENTATION

The code provided contains various sets of functions which have various functionalities and allow in the smooth and seamless processing of the code. The variables defined (global variables included) have been assigned their corresponding values based on the dimensions of the LED Panel and can be changed later on for different panels requirements. In the initial section the input, clock and latch GPIO ports including the ADC channels for the joysticks have been assigned variable names which makes the program easier to understand. The global variables including their usage has been clearly mentioned in the code.

For this implementation of the game, we have set the speed of the ball at 1 and incremented the speed by a unit every time it hits a paddle. Once the ball reaches a maximum speed it continues to stay at this speed and moves with a constant velocity until a player gains a point.

A few of the functions such as ClockPulses(), updateDisplay(), setRowAndPerformClockPulses() have been included in the code due to the requirements as provided in the LED Panel lab sheet. After a researching and testing out the code we have come up with an easier implementation of these processes.

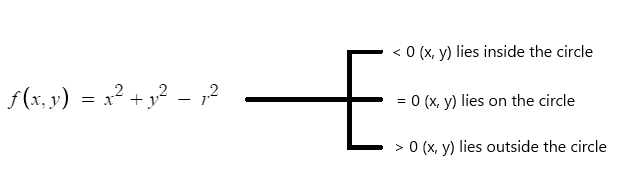
The bullet points below contain each function’s functionality as well as how they have been implemented in the program.

Main functions:

* input() – This function gets the values from both the joysticks through ADC register 1 (both the joysticks use the same register) and defines input channels for each direction using the channelSetup() function. Using the uint32\_t values from each channel, the function assigns the left and right bat offset values
* update()- This function is responsible for the ball’s movement and detecting whether the ball collided with a paddle. It compares the scores of each player and resets the game if the maximum score has been achieved.
* render() – This function renders the entire interactive display. Uses functions such as – drawZone(), drawScores etc. to design the game interface. If a winner has been determined the function displays an end screen.

Functions which contribute to displaying visuals in the LED Panel:

* drawPoint() – This function forms the bases for the rest of the functions responsible for adding visuals in the program. This function accepts the x and y co-ordinates of a point on the panel and converts the x coordinate to uint16\_t format using bit shifting operations. Then we use the value obtained from this to select the row selection pin. Then we go through the basic procedure of clearing the latch as well as all the row pins and then set the specific row which can be obtained by performing bit shifting using the y coordinate as well as the value for all the row pins and then finally set the pin.
* drawVerticalLine() and drawHorizontalLine() – This function accepts the starting point coordinates of the line which is to be displayed as well as the length of the line and uses the drawPoint() to display the line.
* drawRect() – Accepts, the x and y coordinate of one corner of the rectangle as well as the breadth and height. This function is specially used for creating the paddles as well as the border
* drawCircle() – As it can inferred from the name of the function, this creates a circle for the ball. Applying the mid-point circle algorithm which is used to determine the points needed for rasterizing a circle, the function accepts the co-ordinates as well as the radius of the circle and calculates a decision parameter. This variable determines the next pixel to be plotted as the algorithm traverses the circumference of the circle. The value of d is updated at each step of the algorithm. The figure below contains a simple mathematical formula for this algorithm:



If the decision parameter is greater than 0 the value of j which is initially the radius decreases because the circle progresses downwards and gets updated based on the values of i and j

If the decision parameter is less than 0 the decision parameter gets updated based on the current value of i

* drawZone() – draws the boundary of the LED Panel
* drawBall() and drawBats()– draws the ball and the paddle respectively
* drawScores() and drawDigit() – both these functions are used to display the score for each player
* drawWinner() – displays a winning message as well as the name of the winner.

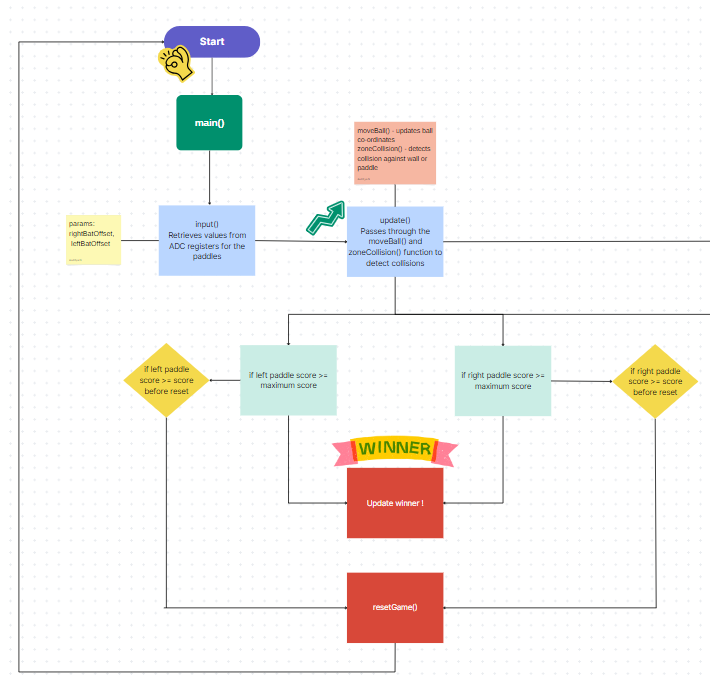
Functions responsible for the functioning of the game

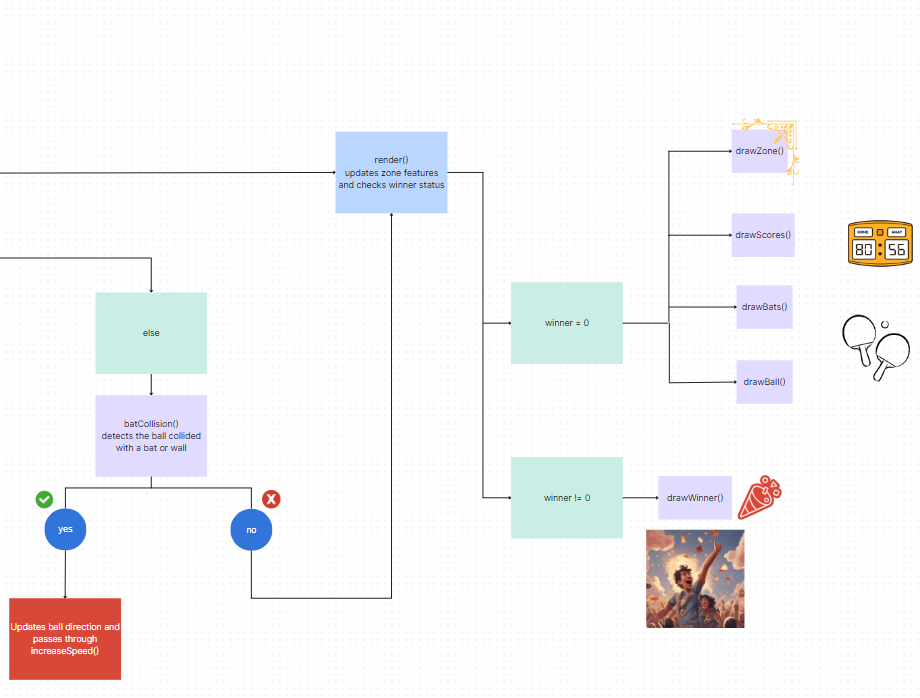
* batCollision() – Rebounds the ball in the opposite direction when it collides with a paddle
* increaseSpeed() – increases the speed of the ball as it collides with a bat. If the speed has reached the maximum threshold, it stays constant at that speed.
* resetBall() – Resets the ball when a player scores a point or when someone wins the game.
* resetGame() – this function is called during the initialization of the game or once someone wins the game.

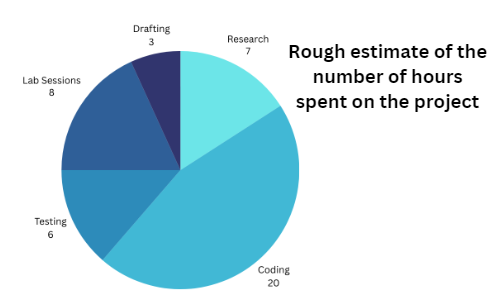
Functions responsible for initializing registers:

* configRegisters() – Configures the registers during the initialization of the game as well as enables the clock
* setup() – sets the each GPIO port as well as specifying the GPIO Mode and output motions.
* clock() – clears and sets the clock
* set() and clear() – for setting and clearing the input pin
* pulse() – performs input and a clock pulse









RESULT

The entire code seems to compile completely while using the gcc compiler. Although we do not have images of the final implementation of the game we did succeed in achieving valuable results

1. Game Interface Development:

Successfully created the game layout as well as the paddle figures and the ball.

1. Incorporated algorithms:

Algorithms such as the Mid-Point Circle algorithm, helped in improving the code and making it more efficient.

1. Refinement of Game Dynamics:

Managed to incorporate the balls movement as well as steady increase in its velocity as it bounces of paddles.

1. Scoring system:

Incorporated the scoring system as well as fixed a maximum achievable score along with a system to update the score after each player wins a round.

Future Improvements :

Throughout the entire journey of creating this project, there were many instances and parts of the program which could be improved or extra features which could be added.

* Better implementation of connecting the joystick to the ADC ports could have been achieved.
* A more concise and improved setup function which does not require too many lines of code.
* Adding a more descriptive winner animation as well as adding more visuals to make the game more appealing
* Adding powerups as mentioned in the coursework lab sheet on the DCS website. Most of the pong games we have researched have special power ups which can either alter the ball by making it small and faster or creating multiple balls. We thought of implementing this by creating mini-circles on the screen and whenever the ball lands inside the dimensions of these circles, the power up activates. If given more time, we believe we will be able to implement such an addition to the game.

CONCLUSION

This experience afforded us invaluable insights and profound knowledge in navigating the intricacies of the OpenCM library and delving into the realm of C programming. Within the expansive library, a plethora of functions lay dormant, promising a transformative enhancement of the game's interface. Our journey extended into the vast realm of processors and microcontrollers, offering a captivating perspective as we embarked on the development of this engaging game. Beyond sharpening our technical prowess, this project served as a crucible for refining our time management and collaborative skills. Through meticulous communication and strategic task allocation, we achieved an accelerated project completion. In retrospect, this experience has been a reservoir of valuable lessons, contributing significantly to our collective growth and proficiency

REFERENCES

Timeline: <https://www.canva.com/design/DAF5B7IbSQ0/ZjSvkz8vVik-UdR8VU5TPw/edit?utm_content=DAF5B7IbSQ0&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton>

Bitwise operations: <https://en.wikipedia.org/wiki/Bitwise_operations_in_C#:~:text=Right%20shift%20%3E%3E,-The%20symbol%20of&text=For%20its%20operation%2C%20it%20requires,three%20places%20and%20so%20on>.

Mid-Point Circle Algorithm: <https://fci.stafpu.bu.edu.eg/Computer%20Science/4899/crs-12417/Files/Midpoint%20Circle%20Algorithm.pdf>

Algorithm wiki: <https://en.wikipedia.org/wiki/Midpoint_circle_algorithm#:~:text=In%20computer%20graphics%2C%20the%20midpoint,further%20generalized%20to%20conic%20sections>.

RGB Matrix pixels: <https://www.sparkfun.com/news/2650>

OpenCM3 Library: <https://github.com/libopencm3/libopencm3/wiki>

OpenCM3 website: <http://libopencm3.org/>

OpenCM3 Reddit page: <https://www.reddit.com/r/embedded/comments/jl6neh/getting_started_with_libopencm3/>

Sample report: <https://www.overleaf.com/latex/templates/university-of-warwick-report-template/kfpfvcbkdcdk.pdf>

