Pong Game

By Rimon Islam and Arvin [2/29/2024]

Objective and Requirements: The objective of this project is to develop a Pong game in C and then have it run on the ChipKit Uno32 with the basic I/O-shield for the users. The main requirements include:

- **Graphics:** Implement visually appealing graphics for the Pong game.
- **Game Modes:** Support one and two-player modes. In one-player mode, the player faces an AI opponent with different difficulty levels.
- **Bounce Angle Calculation:** Determine the bounce angle based on the ball's impact position on the paddle and its current trajectory. Ensure continuous modification of the angle, particularly towards the sides of the paddle. Utilize floating-point numbers to represent ball velocity for smooth angle adjustments.
- **Ball-Paddle Interaction:** Allow the ball to hit the paddle from the Y-direction, resulting in a heavily modified angle of reflection.
- **Highscore:** Implement a highscore system for tracking player performance.

Solution: The solution involved developing the game in C and was developed for the ChipKIT Uno32 platform using the MCB32tools. Graphics were displayed on the small screen on the Basic I/O shield, and controls were managed through the available buttons. The game logic, including AI opponent behavior, the movement of the paddles/ball, creating the win condition, was implemented in C.

Verification: The main method of testing involved running the code consistently on the Chipkit platform to ensure it operated correctly. We closely monitored the input and output functionality to ensure it behaved as expected.

Contributions: Work division will be as follows: Rimon will focus on graphics, AI opponent behavior. Arvin will handle the motion control features, ball-paddle interaction.

Reflections:

The Pong game we developed for the Chipkit platform turned out to run smoothly and effectively. However, it fell short of our initial aspirations for the project. Our primary goal was to create an AI opponent with varying difficulty levels, but we could only implement a default AI with a fixed difficulty level. Despite this limitation, we managed to incorporate features such as a menu system offering different game modes. Players can choose to compete against an AI or engage in a two-player mode.

Unfortunately, due to time constraints, we were unable to implement additional features such as a high score list or fine-tuning the angle adjustment during paddle interactions with the ball. Because of this we had to rethink our work division. Despite these shortcomings, the game still provided an enjoyable experience. We integrated some visually appealing effects, such as using LEDs to indicate scoring, which added to the game's atmosphere. Additionally, the display functionality worked seamlessly, enhancing the overall gaming experience.

Working on this project allowed us to explore the relationship between low-level programming and high-level programming. Although we primarily worked with C code and didn't delve deeply into assembly language, we gained valuable insights into embedded systems and computer technology. The project's tight deadline posed a challenge, and the features we missed out on were a result of time limitations rather than lack of effort.

Overall, the project was a rewarding experience, providing us with practical knowledge and hands-on experience in developing games for embedded systems. While there were areas for improvement, we're proud of what we accomplished given the constraints we faced.