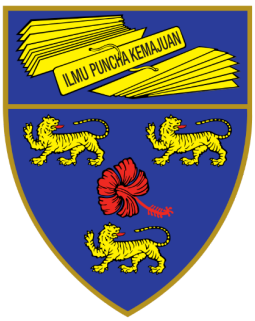

UNIVERSITY OF MALAYA

WHACK-A-MOLE

WIC3003 EMBEDDED SYSTEM PROGRAMMING

SEMESTER 2 2022/23

GROUP 4

DR. ROZIANA RAMLI

23 JUNE 2023

RAJA ZAREEF FIRDAUS BIN RAJA AZMAN NAHAR

17207394/2

FUNCTIONALITY OF INPUT/OUTPUT NO. 1:

Game Menu/ Message Display on LCD

- The LCD display is initialized with the necessary pins and configured.
- The "Whack-A-Mole!"-message is displayed on the LCD to indicate the name of the game.
- The "Press button to start" message is displayed on the LCD to prompt the user to press the button to start the game.
- These messages provide instructions and information to the user, guiding them on how to interact with the game.

CODE EXPLANATION 1:

Code to display messages on LCD

```
#include "mbed.h"
#include "C12832.h"

C12832 lcd(p5, p7, p6, p8, p11);
DigitalIn button(p10);

int main()
{
    int timer = 0;
    bool gameRunning = false;

    lcd.cls();
    lcd.locate(0, 15);
    lcd.printf("Whack-A-Mole!");
    lcd.locate(0, 26);
    lcd.printf("Press button to start");
```

LCD Initialization:

- The LCD display is initialized using the C12832 class from the "C12832.h" library.
- The lcd object is created, and the pin connections p5, p7, p6, p8, p11 are passed as parameters to the constructor.
- This initializes the LCD display and sets up the necessary communication between the microcontroller and the display.

Displaying Messages on LCD:

- After the LCD initialization, the program proceeds to display messages on the LCD screen.
- The cls() function is called to clear the LCD display, ensuring a clean slate before printing messages.
- The locate() function is used to set the cursor position on the LCD screen.
- The printf() function is used to format and print messages on the LCD.
- "Whack-A-Mole!": This is the title of the game and indicates the name of the game being played.

- "Press button to start": This message prompts the user to press the button to start the game.

Updating Displayed Messages:

- Throughout the program, whenever there is a need to update the messages displayed on the LCD, the program follows a similar approach.
- It first clears the LCD using cls() to ensure a clean display area.
- The locate() function is used to set the cursor position.
- The printf() function is used to update and display new messages on the LCD.
- The combination of cls(), locate(), and printf() functions allows the program to control the content displayed on the LCD and update it based on the game's state and user actions.

FUNCTIONALITY OF INPUT/OUTPUT NO. 2:

Push Buttons & Timer

- The button input is initialized and connected to a digital pin on the board.
- Inside the main while loop, the program checks if the button is pressed using the condition button == 1.
- If the button is pressed:
- If the gameRunning variable is true, indicating that the game is already running, the program resets the timer to 30, clears the LCD display, and displays the "Game Reset!" message on the LCD.
- If the gameRunning variable is false, indicating that the game is not running, the program sets gameRunning to true, sets the timer to 30, clears the LCD display, and displays the "Game started!" message on the LCD.
- While the timer is greater than 0, the program enters a nested while loop to display the remaining time on the LCD and decrease the timer by 1. It also checks if the button is pressed during this time using the condition button == 1. If the button is pressed, it exits the inner loop.
- If the timer reaches 0, indicating that the game has ended, the program sets gameRunning to false, clears the LCD display, and displays the "Bye bye!" message on the LCD.
- The program then continues to wait for a short duration (0.1 seconds) before checking the button again, allowing for the button to be released before starting a new game or resetting the current game.

CODE EXPLANATION 2:

```
#include "mbed.h"
#include "C12832.h"

C12832 lcd(p5, p7, p6, p8, p11);
DigitalIn button(p10);

int main()
{
    int timer = 0;
    bool gameRunning = false;

    lcd.cls();
    lcd.locate(0, 15);
    lcd.printf("Whack-A-Mole!");
    lcd.locate(0, 26);
    lcd.printf("Press button to start");

    while (true) {
        if (button == 1) {
            if (gameRunning) {
                // Restart the game and timer
                timer = 30;
                lcd.cls();
                lcd.locate(0, 26);
                lcd.printf("Game Reset!");
            } else {
                // Start the game and timer
                gameRunning = true;
                timer = 30;
                lcd.cls();
                lcd.locate(0, 26);
                lcd.printf("Game started!");
            }

            while (timer > 0) {
                lcd.locate(0, 15);
                lcd.printf("Time Left: %2d s", timer);
                timer--;

                wait(1.0);

                if (button == 0) {
                    // Exit the inner loop if button is released
                    break;
                }
                button.read(); // Read the button state to update its value
            }

            if (timer == 0) {
                // End the game if the timer reaches 0
                gameRunning = false;
                lcd.cls();
                lcd.locate(0, 15);
                lcd.printf("Bye bye!");
            }
        }
```

Button Initialization:

- The button is initialized using the DigitalIn class from the mbed library.
- The button object is created, and the pin connection p10 is passed as a parameter to the constructor.
- This sets up the button for digital input and allows the program to read its state.

Button Functionality:

- The program enters an infinite loop using while (true).
- It continuously checks the state of the button using button == 1.
- If the button is pressed (logic level 1), the program executes the code inside the if (button == 1) block.
- Inside the block, there is a check to determine whether the game is running or not based on the gameRunning boolean variable.
- If the game is running, it means the button press is to restart the game. In this case:
- The timer is reset to its initial value of 30.
- The LCD is cleared using cls() and the message "Game Reset!" is displayed.
- If the game is not running, it means the button press is to start the game. In this case:
- The gameRunning variable is set to true.
- The timer is set to its initial value of 30.
- The LCD is cleared using cls() and the message "Game started!" is displayed.

Timer Functionality:

- After the button press, the program enters a while loop that runs as long as the timer is greater than 0.
- Inside the loop, the current value of the timer is displayed on the LCD using printf().
- The timer is decremented by 1.
- The program waits for 1 second using wait(1.0) before moving to the next iteration of the loop.
- During each iteration of the loop, the program also checks the state of the button using button == 1.
- If the button is pressed during the game, it breaks out of the inner loop, allowing the game to be restarted or reset.

Game End:

- After the timer reaches 0, the program checks if the timer is equal to 0.
- If the timer is 0, it means the game has ended.
- The gameRunning variable is set to false.
- The LCD is cleared using cls() and the message "Bye bye!" is displayed.
- This combination of button handling and timer functionality allows the program to start, reset, and end the game based on the user's button presses and the timer reaching 0.

FEATURE DEMO VIDEO:
https://drive.google.com/file/d/1bcrt5uIom_bz7yLNqrKiSMAtOKOpqByl/view?usp=sharing
GAME DEMO VIDEO:
https://drive.google.com/file/d/1x3SjtlB_7Oq73XJm3xmP5DqZni3t0uaH/view?usp=sharing

Full Game Code: https://github.com/zareefrj/Mbed-Whack-A-Mole/blob/main/Whack-A-Mole.txt