Hardware Software Interface

F28HS

Coursework 2- MasterMind using Pi

Tehan Ranuk Miskin – tr2080@hw.ac.uk

Simon Getachew Girma – sgg2002@hw.ac.uk

Dubai Campus

1. Problem Specification:

This project uses the Raspberry Pi to implement a MasterMind game. MasterMind game requires players to guess a random generated sequence and are required to guess the sequence within a certain number of attempts. This version of MasterMind use:

- A 3 colour- system (blue, green and red) with a sequence of 3.
- The game will allow a maximum of 5 attempts to allow the player to guess.
- A button is used to input the number in the sequence.
- After a round has finished, LEDs will blink giving feedback to show the player how many exact matches and how many approximate matches are found.
- The LCD will display the outputs.

2. Hardware Specification and Wiring:

Components used:

- Raspberry Pi 3 Model B
- LEDs- A Green LED connected to GPIO pin 26 and a Red LED connected to GPIO pin 5
- Button- Connected to GPIO pin 19
- 16x2 LCD Display- LCD controls connected to GPIO Pin 24 and GPIO Pin 25. LCD data connected to GPIO Pin 23, GPIO Pin 10, GPIO Pin 27 and GPIO Pin 22.
- Wiring- A slightly altered version of the diagram in the CW2 specification by connecting certain wires directly from the Pi to the LCD

3. Code Structure:

Hardware Interface-

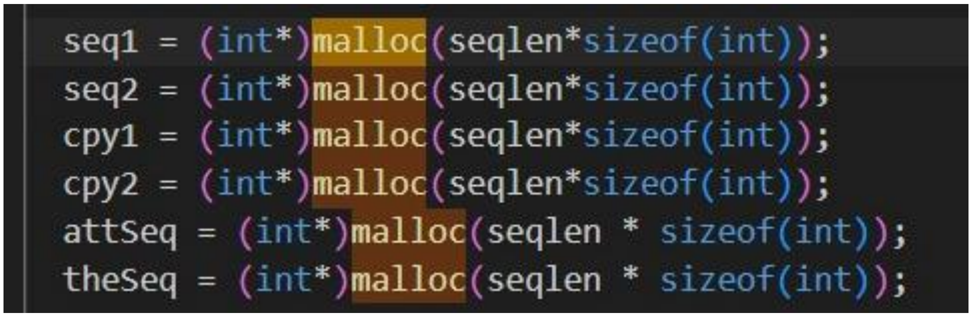
Function	Purpose
digitalWrite	Sets the value of the GPIO Pin
writeLED	Controls the state of LEDs

Game Logic-

Function	Purpose
BlinkN	To check how many times the LED blinks
initSeq	Intializes the secret sequence
showSeq	Shows the secret sequence
countMatches	Compares users guess and secret sequence
showMatches	Displays the results on the LCD
redSeq	Parses the integer as a sequence of digits.

4. Performance-Relevant Design Decisions:

Dynamically Allocated Memory:



```
seq1 = (int*)malloc(seqlen*sizeof(int));
seq2 = (int*)malloc(seqlen*sizeof(int));
cpy1 = (int*)malloc(seqlen*sizeof(int));
cpy2 = (int*)malloc(seqlen*sizeof(int));
attSeq = (int*)malloc(seqlen * sizeof(int));
theSeq = (int*)malloc(seqlen * sizeof(int));
```

Free Memory:

```
free(theSeq);
free(seq1);
free(seq2);
free(cpy1);
free(cpy2);
free(attSeq);
free(lcd);
```

5. Hardware-Accessing Functions:

Function	Hardware	Implementation	Description
digitalWrite()	GPIO	ARM Assembly	Sets GPIO high or low
pinMode()	GPIO	ARM Assembly	Configures GPIO as input or output
writeLED()	LED	ARM Assembly	Controls the LEDs
readButton()	Button	ARM Assembly	Reads the Button state
waitforButton()	Button	C	Waits for button press event
sendDataCmd()	LCD	C	Sends data to LCD
lcdPutChar	LCD	C	Sends character to LCD
lcdPut4Command()	LCD	C	Sends a 4 bit command to the LCD
lcdPutCommand()	LCD	C	Sends commands to the LCD

6. ARM Assembly Matching Implementation:

Inputs:

- seq1 : Pointer to the secret sequence
- seq2 : Pointer to the guessed sequence

Both sequences are in the form of integers that are called and defined by SEQL which is a global constant

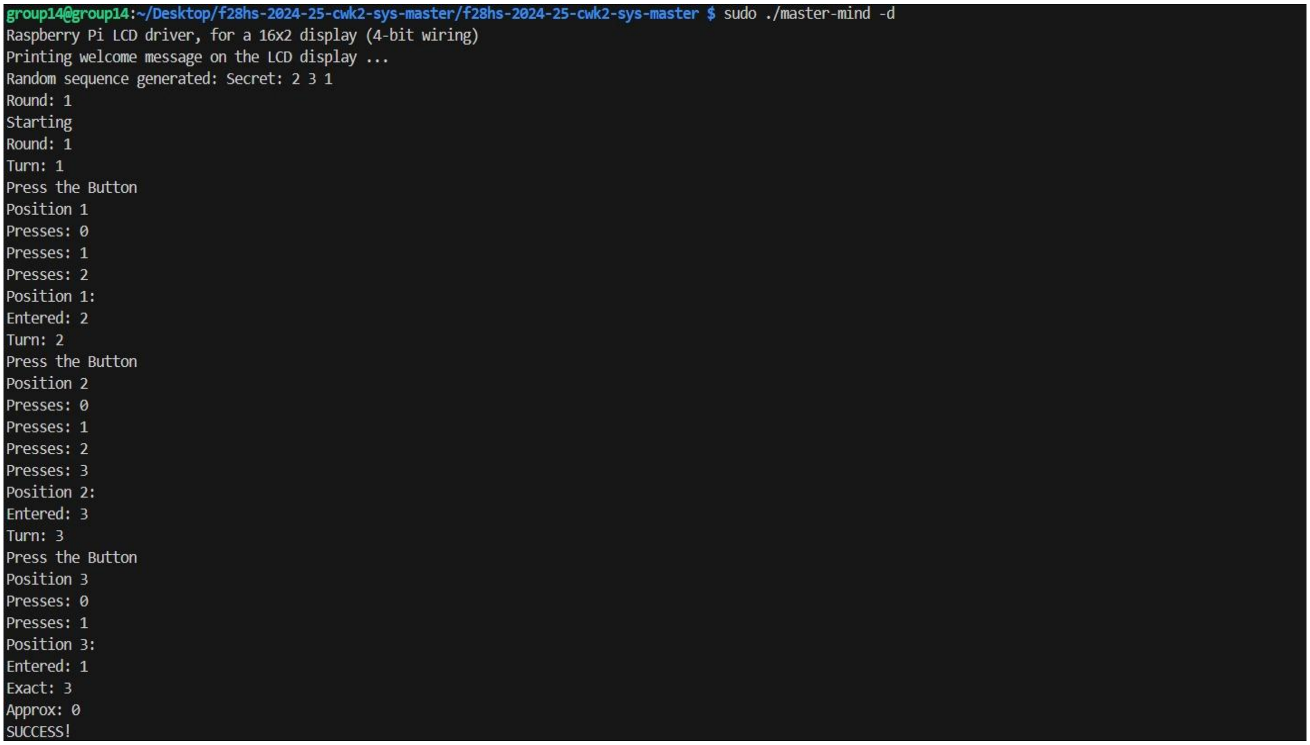
Outputs: Returns 2 sets of 16-bit integers with

- The lower 16 bits return the exact positions
- The upper 16 bits return the approximate matches

Sub-routine Breakdown:

- It counts the exact matches by using direct position to position comparison.
- It will then count the approximate matches, it will carefully check to not count the exact matches as well
- Finally the results are combined shifting the approximate matches left by 16 bits.

7. Example Game Output:



```
group14@group14:~/Desktop/f28hs-2024-25-cwk2-sys-master/f28hs-2024-25-cwk2-sys-master $ sudo ./master-mind -d
Raspberry Pi LCD driver, for a 16x2 display (4-bit wiring)
Printing welcome message on the LCD display ...
Random sequence generated: Secret: 2 3 1
Round: 1
Starting
Round: 1
Turn: 1
Press the Button
Position 1
Presses: 0
Presses: 1
Presses: 2
Position 1:
Entered: 2
Turn: 2
Press the Button
Position 2
Presses: 0
Presses: 1
Presses: 2
Presses: 3
Position 2:
Entered: 3
Turn: 3
Press the Button
Position 3
Presses: 0
Presses: 1
Position 3:
Entered: 1
Exact: 3
Approx: 0
SUCCESS!
```

8. Summary and Conclusions:

What was achieved:

MasterMind was successfully implemented with somewhat efficient usage of resources using high level programming language.

What was not achieved:

The implementation of countMatches in MasterMind was done in C and not in ARM Assembly.

Outstanding Features:

Simple program using LCD and button, which are connected by using an optimization wiring system.

What was learned:

We gained experience in using ARM assembly programming skills and maintained functionality within the constraints of the Raspberry Pi. How to properly connect circuit components such as buttons and potentiometers. Finaly, how to use both C and ARM assembly to create an engaging user experience.