Hardware-Software Interface (F28HS) Course Work Master Mind Application

Group Members:

Ayushi MadhuKumar Amin – H00331154

Harini Balamurugan – H00334975

Degree Programme: BSc(Hons) Computer Science

Campus: Dubai

The coursework requires us to implement a simple instance of the MasterMind board game using the Raspberry Pi2, with two LEDs and a button attached to it.

Hardware Specification and Wiring

Hardware Environment: The Raspberry Pi2 is used. The keyboard and mouse is used for input and the monitor is used for output.

Wiring: Two LEDs are used as output, a green LED and a red LED. The green LED is connected to the RPi2 using GPIO pin 13 and the red LED is connected to the RPi2 using GPIO pin 5. A button is used as an input device and is connected to the RPi2 using GPIO pin 19.

Specification of the Code

Pin 13 (Green LED) and pin 5 (Red LED) are set to output mode and pin 5 (button) is set to input mode. The secret is generated randomly using the rand() function and is stored in an integer array (int*) *secret* of length *N*. The green LED blinks once to indicate the beginning of the game.

blinkLED Function

The value of pin is checked and setOff and clrOff are set to 8 and 11 if pin>=32 or 7 and 10 if pin<32.

1 is written in pin number *pin* in GPSET and after a time delay of 1 second, 1 is written in pin number *pin* in GPCLR. There is a time delay of 1 second. This process repeats for *noOfTimes* times.

getButtonValue Function

The value of pin is checked and gplev is set to 14 is pin>=32 and 13 is pin<32.

If the value in pin number *pin* of GPLEV is 1, we return 1 else 0 is returned.

buttonClick Function

Within a time period of 7 seconds, we check the value of the pin number *pin* in GPLEV using the getButtonValue() function and we increment the *buttonClickNo* counter when the button is released. The red LED blinks once to indicate that the input has been accepted. If the button has not been pressed or if it is clicked more than the required number of times, the red LED blinks 5 times and the buttonClick() function is called again.
The green LED blinks the number of times the button has been clicked.

In main()

We get the guess from the user using the buttonClick() function and the guess is stored in an integer array (int *) *guess* of length *N*. The red LED blinks twice to indicate the input has been completed.

Two integer arrays (int*) *guessFlag* and *secretFlag* of length *N* are declared and are used as flags.
Using a for loop, first the j^{th} position of *secretFlag* and *guessFlag* are set to 0. If the elements in the j^{th} position in *secret* and *guess* are equal (which indicates that the correct colour is in the correct position), we increment the *exact* counter and we set first the j^{th} position of *secretFlag* and *guessFlag* to 1.
Then using another for loop, if the j^{th} position in *guessFlag* is 0, we compare that element with all the elements in *secret*. If they are

equal and if the jth position in *secretFlag* is 0, then we increment the *approximate* counter.

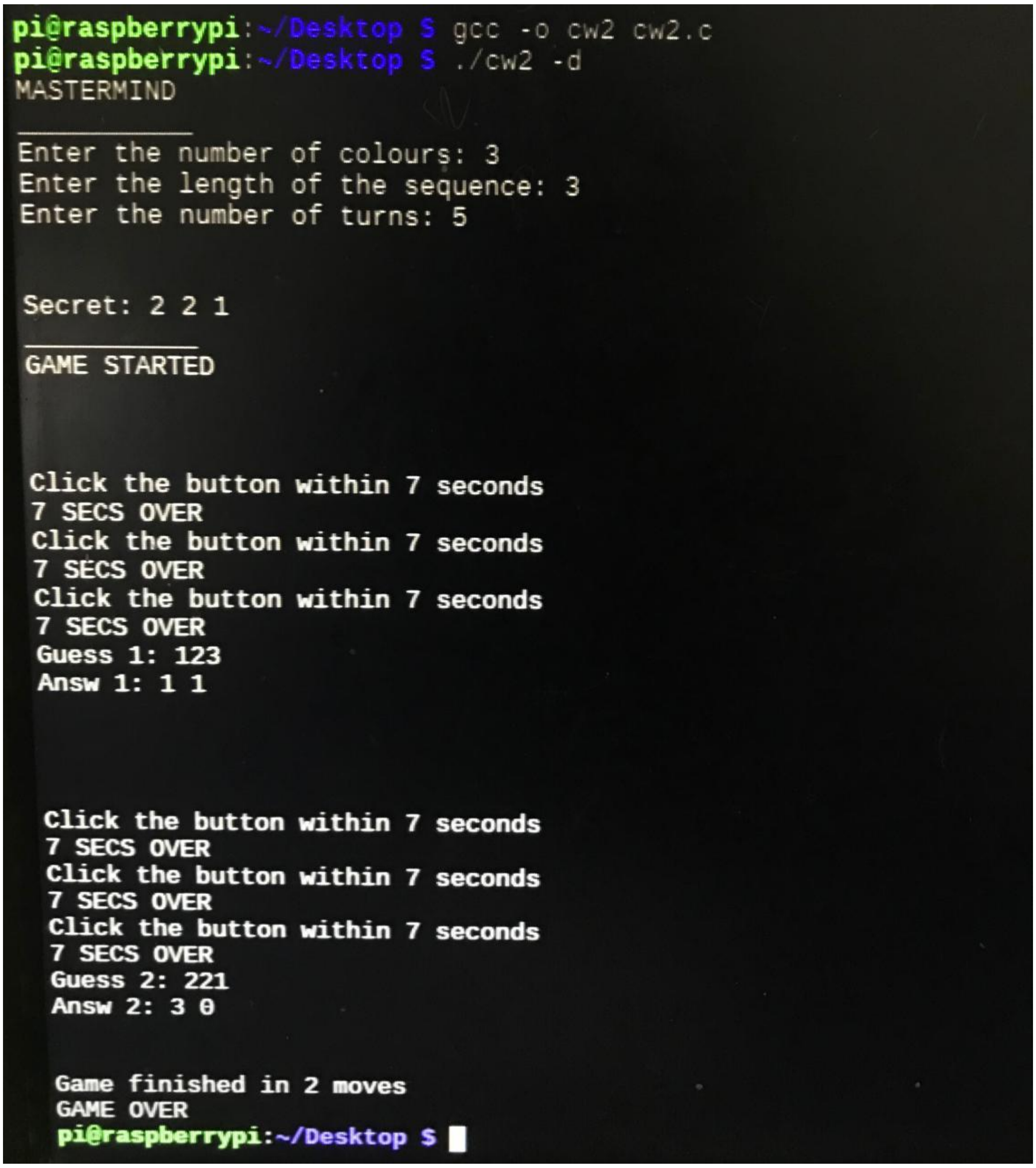
The green LED blinks *exact* number of times and then the red LED blinks once and then the green LED blinks *approximate* number of times.

If *exact* is equal to *N*, then green LED blinks 3 times when the red LED is on indicating that the game has been won or else the red LED blinks 3 times and this process repeats for *noOfTurns* times.

<u>Functions that directly access the hardware (All the functions use C)</u>

- void blinkLED(int pin, int noOfTimes)
- void setOutputMode(int pin)
- void setInputMode(int pin)
- int getButtonValue(int pin)
- int buttonClick()
- void main(int argc, char* argv[])

Sample execution in debug mode



```
pi@raspberrypi:~/Desktop $ gcc -o cw2 cw2.c
pi@raspberrypi:~/Desktop $ ./cw2 -d
MASTERMIND
____________
Enter the number of colours: 3
Enter the length of the sequence: 3
Enter the number of turns: 5


Secret: 2 2 1
____________
GAME STARTED



Click the button within 7 seconds
7 SECS OVER
Click the button within 7 seconds
7 SECS OVER
Click the button within 7 seconds
7 SECS OVER
Guess 1: 123
Answ 1: 1 1




Click the button within 7 seconds
7 SECS OVER
Click the button within 7 seconds
7 SECS OVER
Click the button within 7 seconds
7 SECS OVER
Guess 2: 221
Answ 2: 3 0


Game finished in 2 moves
GAME OVER
pi@raspberrypi:~/Desktop $
```

Summary

Our program successfully implements the MasterMind game. It meets all the requirements and functionalities. It has been written fully using C.

Limitation: We did not use Assembly to implement the low-level operations of setting the mode of a pin, writing to an LED and reading from a button.

This coursework has helped us in learning more about the Raspberry Pi2 and the GPIO pins. It also helped us learn how to set the mode of a pin, write to an LED and read the value from a button using C. It helped us learn how to use the breadboard in connecting the LEDs, resistors and the button to the RPi2 using connecting wires. We also learnt about various functions available in the C library such as the clock() in time.h and mmap() in sys/mman.h.

References

1. https://www.geeksforgeeks.org/generating-random-number-range-c/ (for generating random within a range)
2. https://cboard.cprogramming.com/c-programming/112560-timers-c.html (for time delay)