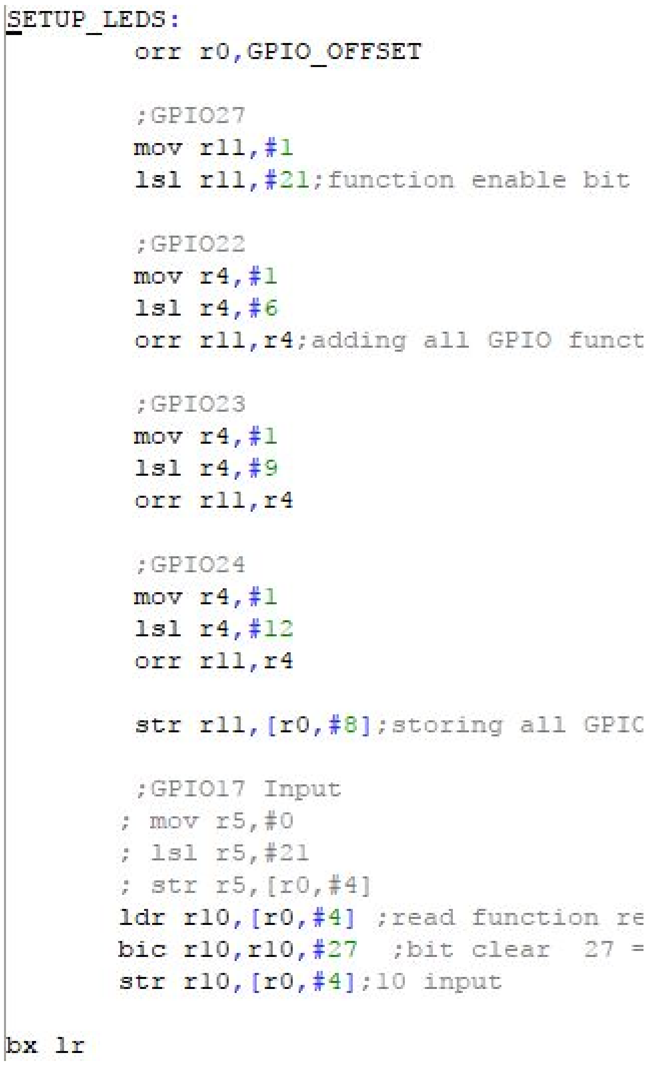
Simon Dunkley
1628623
COS10004
LA1-06

Part B Raspberry Pi Assembly

Brief description
I have developed a simple game in which you use a button in combination with 4 LEDs to test the player's reflexes. Press and hold the button until the green light is displayed.

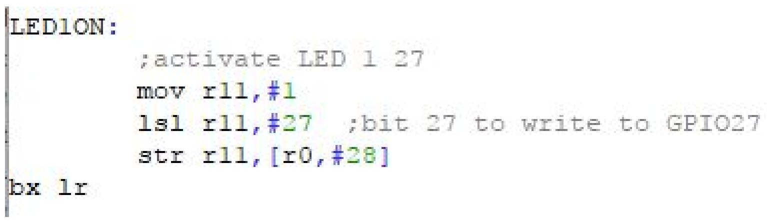
Outline of design
Initially the red LED is displayed to inform the user, the game is ready. Press and hold the button and watch the first 2 yellow LEDs light up in sequence then wait until the green LED lights up, then release the button. If released too early the red button will be displayed. The green LED will then light up for the amount of time you took to release the button and the circuit will reset.

GPIO_Init.asm
The first function initialises the four LEDs to allow them to display output as well as initialising the input for the button. The rest of the code is for activating and deactivating all



```
SETUP_LEDS:
        orr r0,GPIO_OFFSET

        ;GPIO27
        mov r11,#1
        lsl r11,#21;function enable bit

        ;GPIO22
        mov r4,#1
        lsl r4,#6
        orr r11,r4;adding all GPIO funct

        ;GPIO23
        mov r4,#1
        lsl r4,#9
        orr r11,r4

        ;GPIO24
        mov r4,#1
        lsl r4,#12
        orr r11,r4

        str r11,[r0,#8];storing all GPIC

        ;GPIO17 Input
       ; mov r5,#0
       ; lsl r5,#21
       ; str r5,[r0,#4]
       ldr r10,[r0,#4] ;read function re
       bic r10,r10,#27  ;bit clear  27 =
       str r10,[r0,#4];10 input

bx lr
```

the LEDs.

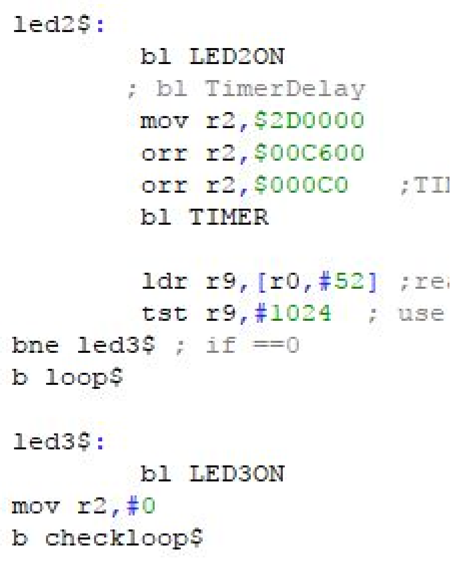


```
LED1ON:
        ;activate LED 1 27
        mov r11,#1
        lsl r11,#27  ;bit 27 to write to GPIO27
        str r11,[r0,#28]
bx lr
```

Timer
 Creates a delay between events so it doesn't happen all too fast.

Main.asm
Multiple branching conditional statements for determining the result while playing the game. This is the core of my work on this assignment. Constantly checking to see if the button has been released, and would return to the main function if input is released. Otherwise will light up the next led. Once all LEDs are lit up it will count and loop until the button is released and then display the delay from when the green button lit up to when the user let go of the button, by flashing the lights once.

```
;start turn off all but red led and wait for button press
loop$:
        bl LED1OFF
        bl LED2OFF
        bl LED3OFF
        bl LED4ON

        ldr r9,[r0,#52] ;read gpios 0-31
        tst r9,#1024  ; use tst to check bit 10
bne start$ ; if ==0
b loop$


;once green light/last led has gone off count up time del
checkloop$:

        add r2,#1

        ldr r9,[r0,#52] ;read gpios 0-31
        tst r9,#1024  ; use tst to check bit 10
beq displayloop$ ; if ==1
b checkloop$

;display delay between button release and green light
displayloop$:
        bl LED1OFF
        bl LED2OFF
        bl LED3OFF
        mov r3,r2
        mov r2,$2D0000
        orr r2,$00C600
        orr r2,$000C0   ;TIMER_MICROSECONDS = 500,000
        bl TIMER
        bl LED1ON
        bl LED2ON
        bl LED3ON
        mov r2,r3
        bl TIMER


b loop$
```

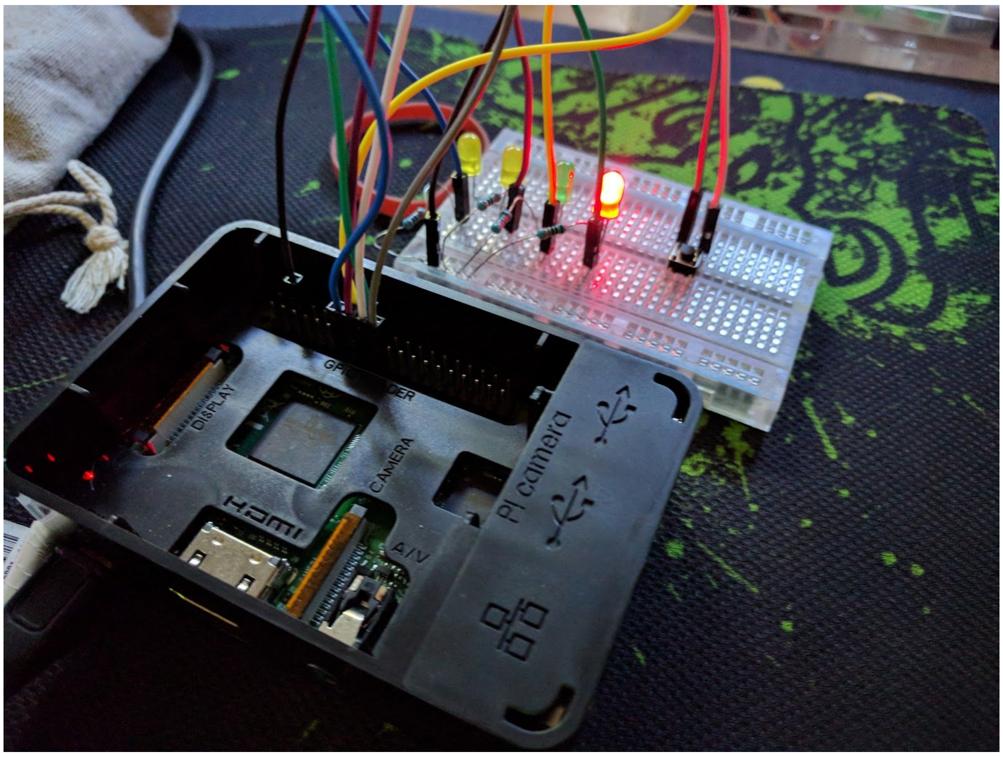


```
led2$:
        bl LED2ON
       ; bl TimerDelay
        mov r2,$2D0000
        orr r2,$00C600
        orr r2,$000C0   ;TI
        bl TIMER

        ldr r9,[r0,#52] ;re
        tst r9,#1024  ; use
bne led3$ ; if ==0
b loop$

led3$:
        bl LED3ON
mov r2,#0
b checkloop$
```

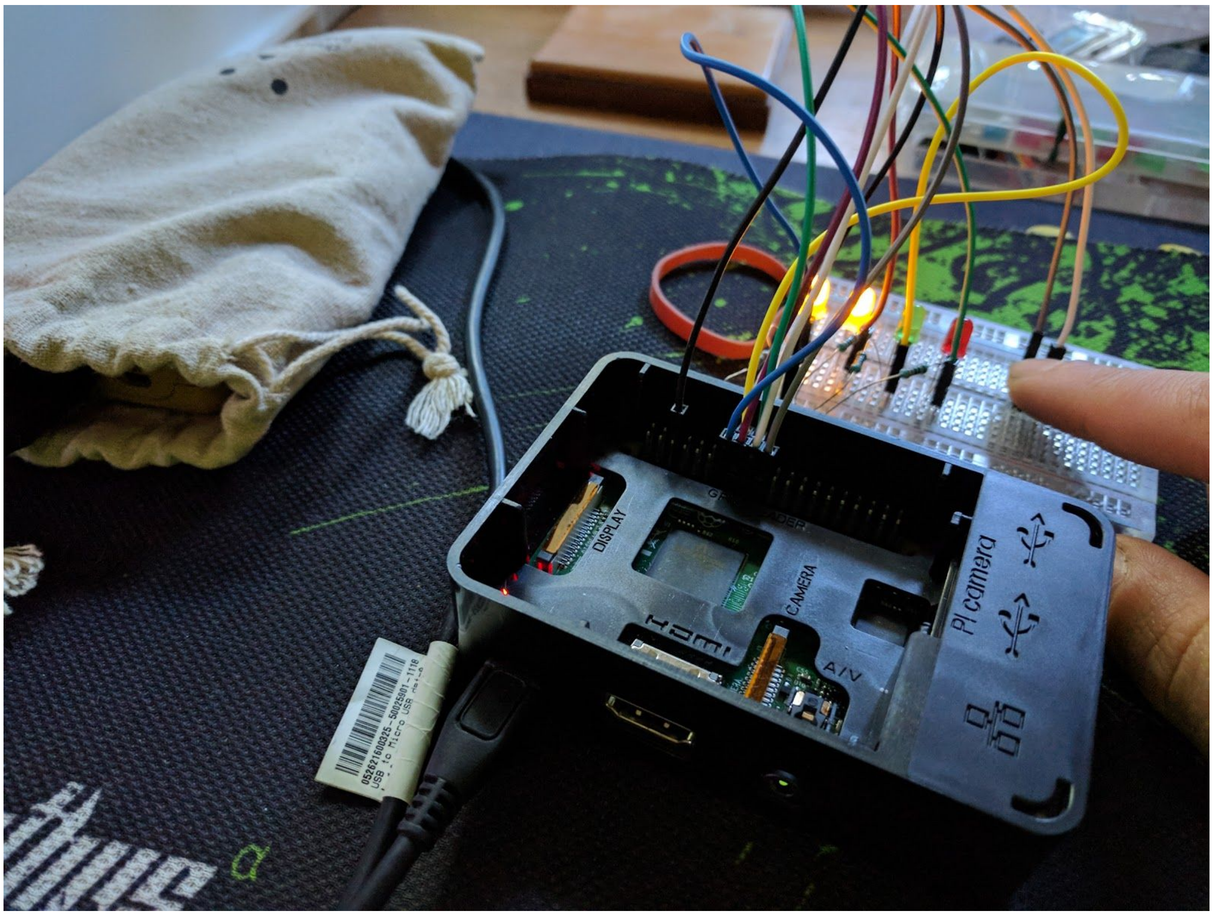
Assumptions made
The user knows how to play the game or will figure it out with pressing the button.
User has to press and hold the button to play
The system works without flaw and cannot get stuck in a strange loop.

Unresolved Problems
So far could not get the smart timer to work in my situation, so instead used the dumb timer which counts loops. This has affected the accuracy of the delays between actions between events.
Also could not get a proper counter to work so the delay at the end is not accurate to the reaction time of a player.
Otherwise the code was working well.
Images

Displaying the Pi is initialised and ready to be used



Mid cycle with 2 of the three lights on



Ready for button to be released with all 3 LEDs on