프로세서응용종합설계

2주차 실습 결과보고서

담당교수 : 황신환 교수님

제출일자 : 20.10.14

학 과 : 전자공학과

학 년 : 3학년

이 름 : 배준성

학 번 :

한국외국어대학교
HANKUK UNIVERSITY OF FOREIGN STUDIES

1. 코드

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setup(17, GPIO.OUT)
GPIO.setup(18, GPIO.IN)



i = 1
state = 'off'
try:
    while True :
        if GPIO.wait_for_edge(18, GPIO.RISING, timeout=i*100): #18 번 핀으로
스위치가 눌러지는 신호를 받으면 동작
            i += 1 #단계증가
            if i > 4:i=1 #1 단계로 돌아감
        elif state == 'off': #LED 의 ON/OFF 상태 구분
            GPIO.output(17, GPIO.HIGH) #17 번 핀으로 LED 제어
            state = 'on'
        elif state == 'on' :
            GPIO.output(17, GPIO.LOW)
            state = 'off'

except KeyboardInterrupt:
    pass
```

2. 결과 사진

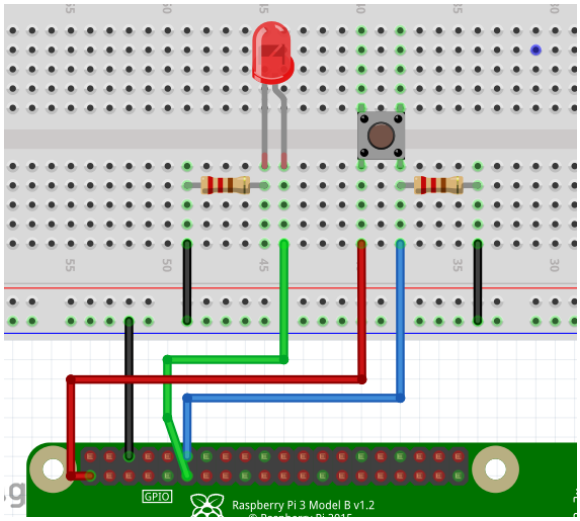

사진 2 – Fritzing으로 그려본 회로

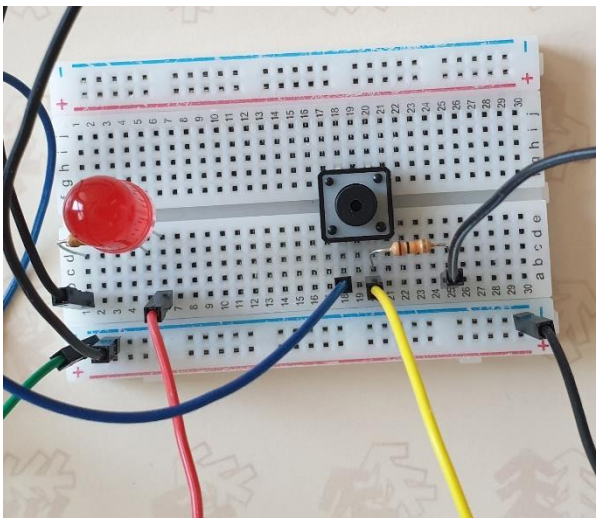

사진 3 - 실제로 구현한 회로

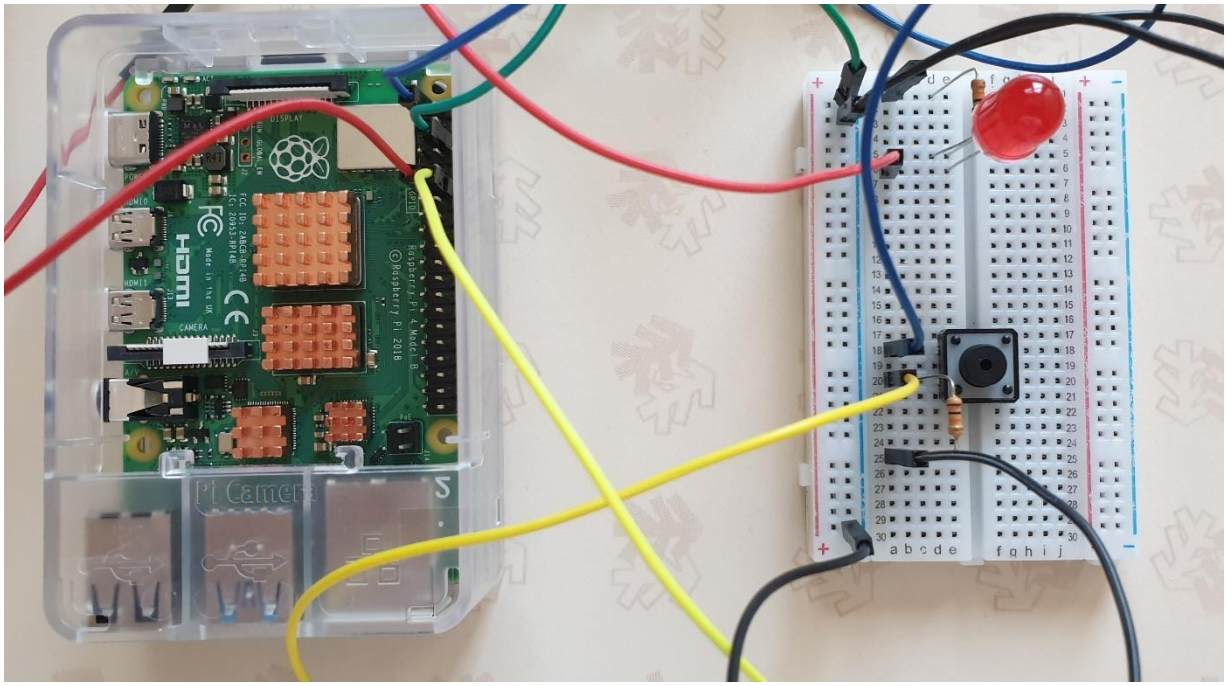

사진 1 - 라즈베리파이를 포함한 전체 회로

3. 결론 및 고찰

본 실습을 통해서 스위치가 눌러지면 4단계로 LED 점멸 상태가 변하는 프로그램을 작성해 보았다. 우선 if문을 이용하여 18번 핀에 입력이 있으면 점멸 단계가 변하고 이를 바탕으로 또 하나의 if문을 추가하여 LED를 점멸하는 프로그램을 만들어보았다. 하지만 이 방법을 이용하면 LED가 켜지거나 꺼지고 time.sleep()이 동작하여 멈춰 있는 동안 18번 핀으로 값을 받아들일 수 없어서 버튼이 눌러져도 동작하지 않는 경우가 있었다. 따라서 time.sleep()을 이용하는 대신 다른 방법을 이용하여 입력을 받아들여 동작하도록 만들어주어야 정상적으로 동작할 수 있을 것이다.

따라서 GPIO.wait_for_edge() 함수를 이용하여 입력을 받아들이는 동시에 delay를 만들어주는 코드를 작성하였다. wait_for_edge 함수는 edge가 발생할 때까지 프로그램을 수행하지 않고 기다리는 프로그램이다. 괄호 안에는 (핀번호, 엣지상태, 대기시간)이 들어간다. 따라서 wait_for_edge(18, GPIO.RISING, timeout = i*100)이라는 형태로 작성하였는데 그 의미를 해석해보면 ' 18번 핀에서 Rising edge(0->1)로 바뀌기 전까지 대기한다. 만약 시간이 i*100 ms만큼 지나면 대기를 멈추고 None을 출력한다. 혹은 Rising edge가 입력되어 핀 번호를 출력한다.' 이다. 대기시간에 있는 i의 경우 단계를 판단하는 기준으로도 사용되는데 i = 1로 최초에 초기화되고 18번 핀에 Rising edge가 입력되면 함수는 숫자 18을 출력하게 된다. 만약 버튼이 눌러지지 않고 timeout만큼의 시간이 지나게 되면 None을 출력하게 된다. 이것을 if문과 연결하여 사용하면 파이썬의 경우 if문의 조건이 None이면 False로 동작하고, 정수라면 True로 동작하게 되므로 이 경우에는 스위치가 눌러졌을 때 단계를 하나 카운트하는 코드로 사용할 수 있게 된다. 다음의 elif문은 LED의 ON/OFF를 state라는 변수를 통해서 판단하여 ON/OFF한다. 즉 wait_for_edge() 함수는 버튼이 눌러지지 않았을 때는 이 함수가 time.sleep으로 동작하고 버튼이 눌러지면 input의 기능을 하게 된다고 할 수 있다.

이렇게 이번 실습을 통해서 GPIO의 기본적인 함수들을 사용해보고 또한 wait_for_edge함수도 사용해 볼 수 있었다. 이번 실습의 경우 이 함수를 통해서 스위치의 입력을 제대로 받을 수 있었지만 만약 전체적인 코드가 길어지거나 다른 요인으로 인하여 수행시간이 길어지면 delay에 의해서 또 문제가 발생할 수 있다. 그러나 인터럽트를 사용하면 이러한 문제를 완전히 배제할 수 있을 것이라고 생각된다.