```
GPIO.setup(18, GPIO.IN)
GPIO.setup(22, GPIO.IN)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)
while True:
    #초점용 누름 버튼이 눌렸는지 확인한다.
    if(GPIO.input(18) == False):

        #초점 선에 연결된 옵토아이솔레이터를 트리거한다.
        GPIO.output(4, GPIO.HIGH)
        #초점을 맞추기 위해 일정 시간 대기한다.
        time.sleep(.2)

        #옵토아이솔레이터를 초기화한다.
        GPIO.output(4, GPIO.LOW)

    #셔터 누름 버튼이 눌렸는지 확인한다.
    if(GPIO.input(22) == False):
        #셔터선에 연결된 옵토아이솔레이터를 트리거한다.
        GPIO.output(17, GPIO.HIGH)
        time.sleep(.2)
        GPIO.output(17, GPIO.LOW)

    #플래시 누름 버튼이 눌렸는지 확인한다.
    if(GPIO.input(23) == False):
        #플래시 선에 연결된 옵토아이솔레이터를 트리거한다(리비전2 보드이므로 27번 핀임).
        GPIO.output(27, GPIO.HIGH)
        time.sleep(.2)
        GPIO.output(27, GPIO.LOW)
```

아래와 같이 터미널 창에서 직접 프로그램을 실행할 것을 추천한다.

```
sudo python CameraControl.py ↵
```

각각의 누름 버튼을 눌러 보고 그에 맞게 작동하는지 확인하자. 확인을 마쳤으면, ^c를 입력해서 프로그램을 종료하자.

여기에서 GPIO 핀을 HIGH 상태로 만든 후에 다시 LOW로 초기화했다는 점에 주목하자. 이를 통해 다음 작동을 위한 올바른 상태를 유지할 수 있다.

SoundModTest.py

SoundModTest.py는 소리 모듈을 테스트하기 위한 프로그램으로, 소리 모듈이 소리를 감지했을 때 초점, 셔터, 그리고 플래시를 트리거할 수 있는지 확인한다. 프로그램의 전반부는 CameraControl.py와 동일하다. 이 프로그램은 화면에 수행하고자 하는 테스트에 해당하는 번호를 표시하는 방식으로 사용자에게 UI를 제공한다. 각각의 테스트와 번호는 아래와 같다.

- 1: 셔터만 테스트하기
- 2: 플래시만 테스트하기
- 3: 셔터와 플래시를 함께 테스트하기
- 0: 프로그램 종료

이 프로그램은 요청받은 기능을 수행하기 전에 소리 모듈에 트리거가 들어오기를 기다리는 일련의 반복문으로 짜여 있다. 이 책의 웹사이트(www.mhprofessional.com/raspi)에서 내려받을 수 있다. 래치 작동에 유의하여, 트리거가 이루어진 이후에는 소리 모듈을 반드시 초기화하도록 하자. 박수 한 번으로 트리거를 하고 나서 모듈의 LED를 끄고 적합한 대기 상태로 돌아가게 하면 된다.

SoundModTest.py

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(17, GPIO.OUT)
#여기에서는 리비전 2 보드를 사용하므로, 21번 핀 대신 27번 핀을 사용하였다.
GPIO.setup(27, GPIO.OUT)
GPIO.setup(18, GPIO.IN)
GPIO.setup(22, GPIO.IN)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
```

```
GPIO.setup(25, GPIO.IN)
select = input('Enter 1 (shutter only), 2 (flash only), 3 (shutter and flash), 0
(exit): '):
while select > 0:
    #셔터 시험
    if select == 1:
        while True:
            if(GPIO.input(24) == False):
                GPIO.output(17, GPIO.HIGH)
                time.sleep(.2)
                GPIO.output(17, GPIO.LOW)
                break
    #플래시 테스트
    if select == 2:
        while True:
            if(GPIO.input(24) == False):
                GPIO.output(27, GPIO.HIGH)
                time.sleep(.2)
                GPIO.output(27, GPIO.LOW)
                break
    #셔터와 플래시 테스트
    if select == 3:
        while True:
            if(GPIO.input(24) == False):
                GPIO.output(17, GPIO.HIGH)
                GPIO.output(27, GPIO.HIGH)
                time.sleep(.2)
                GPIO.output(17, GPIO.LOW)
                GPIO.output(27, GPIO.LOW)
                break
    select = input('Enter 1 (shutter only), 2 (flash only), 3 (shutter and flash), 0
    (exit): '):
print('All done')
```

LightModTest.py

LightModTest.py는 광 모듈을 테스트하기 위한 프로그램이다. SoundModTest.py와 거의 동일하다. 차이가 있다면, GPIO.input에 대한 if 조건문 3개가 가지는 조건이 if(GPIO.input(24) == False)에서 if(GPIO.input(25) == True)로 바뀐다는 것뿐이다. 이러한 변경이 일어난 이유는 광 모듈이 24번 핀 대신 25번 핀에 연결되어 있고, 앞에서 언급했던 것처럼 그 릴레이 접점은 평상시에 열려 있는 것이 아니라 닫혀 있기 때문이다.

변경 사항이 많지 않으므로 지면을 아끼기 위해 코드를 다시 싣지는 않았다. 앞서 소개한 두 프로그램과 동일한 웹사이트에서 내려받을 수 있다.

이 모듈을 테스트하기 위해서는 광 트랜지스터를 조준하고 레이저를 작동시켜서 광 검출기 보드의 LED가 꺼지게 해야 한다. 다음으로, 불투명한 물체로 레이저 빔을 막아 광 모듈을 트리거하여 요청한 기능이 작동하게 만든다. 광 모듈은 래치 작동을 하지 않으므로 초기화는 필요하지 않다.

저속촬영 기능

마지막으로 테스트할 카메라 컨트롤러의 기능은 저속촬영 기능이다. 여기에서는 TimeLapse.py라는 프로그램을 사용할 것이다. 이것은 최대 촬영 시간을 설정한 후, 그 시간 동안 미리 정한 시간 간격으로 계속 사진을 찍는 기법이다. 주로 셔터만을 사용하기 때문에 프로그램에서도 셔터 기능만을 사용하도록 했다. 보통 저속촬영 기법은 플래시가 필요 없는 야외 촬영에서 사용된다. 또한 이 기능에는 소리나 빛을 사용한 트리거가 필요하지 않기 때문에 앞서 소개한 프로그램에 비해 과정이 비교적 간단하다. 마찬가지로 이 책의 웹사이트(www.mhprofessional.com/raspi)에서 내려받을 수 있다.

TimeLapse.py

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT)
max = input('Enter maximum time (minutes): ')
interval = input('Enter interval time (minutes): ')
#전체 경과 시간을 기록하기 위한 변수.
```