임베디드응용 및 실습

(7주차 과제)

2020161047 박종혁

1. 코드를 추가하여 스위치를 눌렀을 때만 화면에 “click”이 표기되도록 변경
다음은 소스코드입니다.

```
import RPi.GPIO as GPIO
import time

SW1 = 5
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(SW1,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)

try:
    while True:
        sw1Value = GPIO.input(SW1)
        if sw1Value == 1:
            print('click')
        time.sleep(0.2)
except KeyboardInterrupt:
    pass

GPIO.cleanup()
```

코드해석: if문을 추가하여 sw1Value값이 1일 때 click이 출력하도록 하여 버튼이 눌렸을 때만 click이 출력되도록 합니다.
다음은 실행결과입니다.

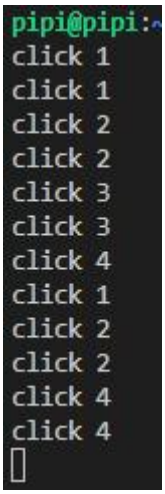
2.몇번 스위치가 눌렸는지 확인 가능하도록 “click x”등으로 화면 출력

다음은 소스코드입니다.

```
import RPi.GPIO as GPIO
import time

SW1 = 5
SW2 = 6
SW3 = 13
SW4 = 19
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(SW1,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(SW2,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(SW3,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(SW4,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)

try:
    while True:
        sw1Value_L = GPIO.input(SW1)
        sw1Value_U = GPIO.input(SW2)
        sw1Value_D = GPIO.input(SW3)
        sw1Value_R = GPIO.input(SW4)
        if sw1Value_L == 1:
            print('click 1')
        elif sw1Value_U ==1:
            print('click 2')
        elif sw1Value_D ==1:
            print('click 3')
        elif sw1Value_R ==1:
            print('click 4')
        time.sleep(0.1)
except KeyboardInterrupt:
    pass

GPIO.cleanup()
```

코드해석:
SW1~SW4까지 버튼에 맞는 숫자를 대입해주고 GPIO.setup()으로 네 스위치를 활성화 시켜줍니다.
if문,elif문을 활용하여 각 버튼이 눌렸을 때 click x를 출력하도록 합니다.

다음은 실행결과입니다.



3.스위치를 눌렀을 때 0->1, 눌렀다가 떼었을 때 1->0으로 값이 변경되므로
0->1인 경우에만 동작되도록 변경하시오.

다음은 소스코드입니다.

```
import RPi.GPIO as GPIO
import time

SW1 = 5
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(SW1,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
pre_sw1Value=0
try:
    while True:
        sw1Value = GPIO.input(SW1)
        if pre_sw1Value ==0 and sw1Value == 1:
            print(sw1Value)
        time.sleep(0.1)
except KeyboardInterrupt:
    pass
GPIO.cleanup()
```

코드해석: 핵심은 pre_sw1Value를 선언하여 이전의 sw1Value값을 저장하고 있는 것입니다. if문으로 이전 값이 0이고 현재 값이 1일때만 print문을 수행합니다.

다음은 실행결과입니다.

4. 4개의 스위치 입력을 받도록 해보자. 화면에 아래와 같이 출력되도록 한다.
단, 리스트를 최대한 활용하여 GPIO 전/후 값을 저장한다.

다음은 소스코드입니다.

```
import RPi.GPIO as GPIO
import time

switch_pins = [5, 6, 13, 19]
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

for pin in switch_pins:
    GPIO.setup(pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

switch_click_count = [0, 0, 0, 0]
prev_switch_values = [0, 0, 0, 0]

try:
    while True:

        for i, pin in enumerate(switch_pins):
            sw_value = GPIO.input(pin)

            if prev_switch_values[i] == 0 and sw_value == 1:
                switch_click_count[i] += 1
                print(f"('SW{i+1} click', {switch_click_count[i]})")
            prev_switch_values[i] = sw_value

        time.sleep(0.1)

except KeyboardInterrupt:
    pass

finally:
    GPIO.cleanup()
```

코드해석:
switch_pins 리스트에 sw1부터 sw4까지 순서대로 핀 번호를 입력해줍니다.
for문으로 GPIO.setup()을 1번부터 4번 핀까지 설정해줍니다.
스위치를 누른 횟수를 저장하는 리스트와 이전 스위치 상태를 저장하는 리스트를 초기화합니다.
while문,for문 안에 있는 if 문의 조건을 보면 스위치가 0->1로 변환된 경우에만 작동하도록 했습니다. if문으로 들어가면 일단 switch_click_count[i]+=1로 클릭 횟수를 1만큼 증가시킵니다. print문으로 몇 번째 스위치를 눌렀는지와 누적횟수를 출력합니다. prev_switch_value[i] = sw_value로 현재 상태를 이전 상태로 저장합니다.

다음은 실행결과입니다.

1) “도레미파솔라시도” 음계를 출력

다음은 소스코드입니다.

```
import RPi.GPIO as gpio
import time

buzzer = 12

gpio.setwarnings(False)
gpio.setmode(gpio.BCM)
gpio.setup(buzzer,gpio.OUT)

p = gpio.PWM(buzzer,261)
p.start(50)

try:
    while True:
        p.start(50)
        p.ChangeFrequency(262)
        time.sleep(1)
        p.ChangeFrequency(294)
        time.sleep(1)
        p.ChangeFrequency(330)
        time.sleep(1)
        p.ChangeFrequency(350)
        time.sleep(1)
        p.ChangeFrequency(392)
        time.sleep(1)
        p.ChangeFrequency(440)
        time.sleep(1)
        p.ChangeFrequency(494)
        time.sleep(1)
        p.ChangeFrequency(523)
        time.sleep(1)
        p.stop()
        time.sleep(1)
```

```
except KeyboardInterrupt:
    pass
p.stop()
gpio.cleanup()
```

코드해석: buzzer에 해당하는 핀 번호인 12를 활성화해주고 p.ChangeFrequency로 음계에 맞는 각 주파수를 입력해주어 “도레미파솔라시도”를 출력하였습니다.

실행결과는 영상파일로 올립니다.

2) 나만의 경적 소리 구현

다음은 소스코드입니다.

```
import RPi.GPIO as gpio
import time

buzzer = 12

gpio.setwarnings(False)
gpio.setmode(gpio.BCM)
gpio.setup(buzzer,gpio.OUT)

p = gpio.PWM(buzzer,261)
p.start(50)

try:
    while True:
        p.ChangeFrequency(330)
        time.sleep(1)
        p.ChangeFrequency(294)
        time.sleep(1)
        p.ChangeFrequency(262)
        time.sleep(1)
        p.ChangeFrequency(294)
        time.sleep(1)
        p.ChangeFrequency(330)
        time.sleep(1)
        p.ChangeFrequency(330)
        time.sleep(1)
        p.ChangeFrequency(330)
        time.sleep(1)
except KeyboardInterrupt:
    pass
p.stop()
gpio.cleanup()
```

코드해석: p.changeFrequency()로 원하는 계이름을 출력하였습니다. “미레도레미미미”를 출력하여 비행기 노래의 일부를 출력하였습니다.

실행결과는 영상으로 올립니다.

3) 스위치를 한번 누르면 경적 소리가 나도록 구현

다음은 소스코드입니다.

```
import RPi.GPIO as gpio
import time

buzzer = 12
SW1 =  5

gpio.setwarnings(False)
gpio.setmode(gpio.BCM)
gpio.setup(buzzer,gpio.OUT)
gpio.setup(SW1,gpio.IN,pull_up_down = gpio.PUD_DOWN)

p = gpio.PWM(buzzer,261)
try:

    while True:
        sw1Value = gpio.input(SW1)
        if sw1Value == 1:
            p.start(50)
            p.ChangeFrequency(262)
            time.sleep(1)
            p.stop()
            time.sleep(1)

except KeyboardInterrupt:
    pass
p.stop()
gpio.cleanup()
```

코드해석: 스위치 작동하는 방법과 부저 울리는 코드를 혼용하여 구현하였습니다. buzzer 핀인 12번과 스위치 핀인 5번을 정의해주고 gpio.setup()으로 활성화 설정을 완료해준 다음에 if문을 사용하여 스위치 값이 1일 때만 부저가 울리도록 하였습니다.

실행결과는 영상으로 올립니다.

4) 스위치 4개를 이용하여 나만의 음악을 연주

다음은 소스코드입니다.

```
import RPi.GPIO as gpio
import time

buzzer = 12
sw1 = 5
sw2 = 6
sw3 = 13
sw4 = 19
gpio.setwarnings(False)
gpio.setmode(gpio.BCM)
gpio.setup(buzzer,gpio.OUT)
gpio.setup(sw1,gpio.IN,pull_up_down = gpio.PUD_DOWN)
gpio.setup(sw2,gpio.IN,pull_up_down = gpio.PUD_DOWN)
gpio.setup(sw3,gpio.IN,pull_up_down = gpio.PUD_DOWN)
gpio.setup(sw4,gpio.IN,pull_up_down = gpio.PUD_DOWN)

p = gpio.PWM(buzzer,261)
try:

    while True:
        sw1Value = gpio.input(sw1)
        sw2Value = gpio.input(sw2)
        sw3Value = gpio.input(sw3)
        sw4Value = gpio.input(sw4)
        if sw1Value == 1:
            p.start(50)
            p.ChangeFrequency(262)
            time.sleep(0.5)
            p.stop()
            time.sleep(0.5)
        elif sw2Value == 1:
            p.start(50)
            p.ChangeFrequency(294)
            time.sleep(0.5)
```

```
            p.stop()
            time.sleep(0.5)

        elif sw3Value == 1:
            p.start(50)
            p.ChangeFrequency(330)
            time.sleep(0.5)
            p.stop()
            time.sleep(0.5)

        elif sw4Value == 1:
            p.start(50)
            p.ChangeFrequency(349)
            time.sleep(0.5)
            p.stop()
            time.sleep(0.5)

except KeyboardInterrupt:
    pass
p.stop()
gpio.cleanup()
```

코드해석:
if, elif문으로 “도레미파”를 각각 스위치 1,2,3,4에 할당한 다음에 해당 스위치를 누를 때마다 음이 출력되도록 하였습니다.

실행결과는 영상으로 올립니다.

1) 오른쪽 모터부분의 코드를 추가하여 정방향으로 50%로
동작 -> 정지 ->동작 ->정지

다음은 소스코드입니다.

```
import RPi.GPIO as GPIO
import time

PWMA = 18
PWMB = 23
AIN1 = 22
AIN2 = 27
BIN1 = 25
BIN2 = 24

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(PWMA,GPIO.OUT)
GPIO.setup(PWMB,GPIO.OUT)
GPIO.setup(AIN1,GPIO.OUT)
GPIO.setup(AIN2,GPIO.OUT)
GPIO.setup(BIN1,GPIO.OUT)
GPIO.setup(BIN2,GPIO.OUT)

L_Motor = GPIO.PWM(PWMA,500)
R_Motor = GPIO.PWM(PWMB,500)
L_Motor.start(0)
R_Motor.start(0)
try:
    while True:
        GPIO.output(AIN1,0)
        GPIO.output(AIN2,1)
        L_Motor.ChangeDutyCycle(50)
        GPIO.output(BIN1,0)
        GPIO.output(BIN2,1)
        R_Motor.ChangeDutyCycle(50)
        time.sleep(1)
```

```
            GPIO.output(AIN1,0)
            GPIO.output(AIN2,1)
            L_Motor.ChangeDutyCycle(0)
            GPIO.output(BIN1,0)
            GPIO.output(BIN2,1)
            R_Motor.ChangeDutyCycle(0)
            time.sleep(1)

except KeyboardInterrupt:
    pass
GPIO.cleanup()
```

코드해석:
PWMA와 PWMB를 설정해주고 AIN,BIN을 설정해 준 다음에 GPIO.PWM() , start()으로 모터를 활성화시킵니다. 왼쪽과 오른쪽 모터가 같은 속도로 가속하고 정지했다가 가속하는 동작을 반복합니다.

실행결과는 영상으로 올립니다.

2) 스위치를 입력 받아 자동차 조종하기
SW1:앞
SW2:오른쪽
SW3: 왼쪽
SW4: 뒤
print문을 사용하여 어느 스위치가 눌렸는지 출력

다음은 소스코드입니다.

```
import RPi.GPIO as GPIO
import time

PWMA = 18
PWMB = 23
AIN1 = 22
AIN2 = 27
BIN1 = 25
BIN2 = 24
sw1 = 5
sw2 = 6
sw3 = 13
sw4 = 19
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(PWMA,GPIO.OUT)
GPIO.setup(PWMB,GPIO.OUT)
GPIO.setup(AIN1,GPIO.OUT)
GPIO.setup(AIN2,GPIO.OUT)
GPIO.setup(BIN1,GPIO.OUT)
GPIO.setup(BIN2,GPIO.OUT)
GPIO.setup(sw1,GPIO.IN,pull_up_down = GPIO.PUD_DOWN)
GPIO.setup(sw2,GPIO.IN,pull_up_down = GPIO.PUD_DOWN)
GPIO.setup(sw3,GPIO.IN,pull_up_down = GPIO.PUD_DOWN)
GPIO.setup(sw4,GPIO.IN,pull_up_down = GPIO.PUD_DOWN)
L_Motor = GPIO.PWM(PWMA,500)
R_Motor = GPIO.PWM(PWMB,500)
L_Motor.start(0)
```

```
R_Motor.start(0)
try:
    while True:
        sw1Value = GPIO.input(sw1)
        sw2Value = GPIO.input(sw2)
        sw3Value = GPIO.input(sw3)
        sw4Value = GPIO.input(sw4)

        if sw1Value ==1:
            GPIO.output(AIN1,0)
            GPIO.output(AIN2,1)
            L_Motor.ChangeDutyCycle(50)
            GPIO.output(BIN1,0)
            GPIO.output(BIN2,1)
            R_Motor.ChangeDutyCycle(50)
            time.sleep(1)
            print('1번 스위치가 눌렸습니다.')
        elif sw2Value ==1:
            GPIO.output(AIN1,0)
            GPIO.output(AIN2,1)
            L_Motor.ChangeDutyCycle(50)
            time.sleep(1)
            print('2번 스위치가 눌렸습니다.')
        elif sw3Value ==1:
            GPIO.output(BIN1,0)
            GPIO.output(BIN2,1)
            R_Motor.ChangeDutyCycle(50)
            time.sleep(1)
            print('3번 스위치가 눌렸습니다.')
        elif sw4Value ==1:
            GPIO.output(AIN1,1)
            GPIO.output(AIN2,0)
            L_Motor.ChangeDutyCycle(50)
            GPIO.output(BIN1,1)
            GPIO.output(BIN2,0)
            R_Motor.ChangeDutyCycle(50)
            time.sleep(1)
```

```
            print('4번 스위치가 눌렸습니다.')

        GPIO.output(AIN1,0)
        GPIO.output(AIN2,1)
        L_Motor.ChangeDutyCycle(0)
        GPIO.output(BIN1,0)
        GPIO.output(BIN2,1)
        R_Motor.ChangeDutyCycle(0)
        time.sleep(1)

except KeyboardInterrupt:
    pass
GPIO.cleanup()
```

코드해석: 스위치와 모터를 각자 맞는 핀번호를 할당하여 GPIO.setup()으로 활성화 시켜주고 if문과 elif문으로 각각의 스위치가 눌렸을 때 동작을 수행하고 몇 번 스위치가 눌렸는지 print해줍니다.

실행결과는 영상으로 올립니다.