

임베디드 과제 2020161123 최선재

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

코드:

```
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)
```

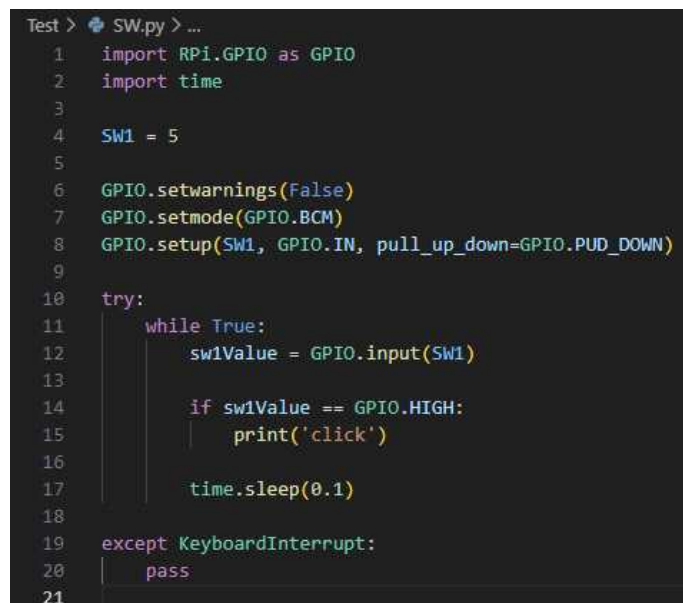
```
        if sw1Value == GPIO.HIGH:
```

```
            print('click')
```

```
        time.sleep(0.1)
```

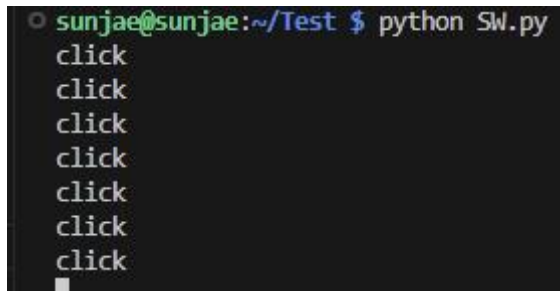
```
except KeyboardInterrupt:
```

```
    pass
```



```
Test > SW.py > ...
1  import RPi.GPIO as GPIO
2  import time
3
4  SW1 = 5
5
6  GPIO.setwarnings(False)
7  GPIO.setmode(GPIO.BCM)
8  GPIO.setup(SW1, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
9
10 try:
11     while True:
12         sw1Value = GPIO.input(SW1)
13
14         if sw1Value == GPIO.HIGH:
15             print('click')
16
17         time.sleep(0.1)
18
19 except KeyboardInterrupt:
20     pass
21
```

결과:

A terminal window with a black background and green text. The prompt is 'sunjae@sunjae:~/Test \$'. The command 'python Sw.py' has been executed, and the output consists of seven lines, each containing the word 'click'.

2. 몇번 스위치가 눌렸는지 확인이 가능하도록 “click x” 등으로 화면 출력  
코드:

```
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)
```

```
previous_states = [GPIO.LOW] * 4
```

```
try:
```

```
    while True:
```

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

```
            if current_state == GPIO.HIGH and previous_states[i] == GPIO.LOW:
                print('click {}'.format(i+1))
```

```
            previous_states[i] = current_state
```

```
            time.sleep(0.1)
```

```
except KeyboardInterrupt:
```

```
    pass
```

```

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)

previous_states = [GPIO.LOW] * 4

try:
    while True:
        for i, pin in enumerate(switch_pins):
            current_state = GPIO.input(pin)

            if current_state == GPIO.HIGH and previous_states[i] == GPIO.LOW:
                print('click {}'.format(i+1))

                previous_states[i] = current_state

            time.sleep(0.1)

except KeyboardInterrupt:
    pass

```

결과:

```

^Csunjae@sunjae:~/Test $ python Sw2.py
click 2
click 3
click 4
click 1
click 3
click 2
click 4
click 1
^Csunjae@sunjae:~/Test $ 

```

3. 스위치를 눌렀을 때 0->1, 눌렀다 떼었을 때 1->0으로 값이 변경되므로 0->1인 경우만 동작되도록 변경

코드:

```
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:
```

```
        state_SW1 = GPIO.input(SW1)
        state_SW2 = GPIO.input(SW2)
        state_SW3 = GPIO.input(SW3)
        state_SW4 = GPIO.input(SW4)
```

```
        print(f"SW1: {state_SW1}, SW2: {state_SW2}, SW3: {state_SW3}, SW4: {state_SW4}")
```

```
        time.sleep(0.1)
```

```
except KeyboardInterrupt:
    pass
```

```
GPIO.cleanup()
```

```

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:
        state_SW1 = GPIO.input(SW1)
        state_SW2 = GPIO.input(SW2)
        state_SW3 = GPIO.input(SW3)
        state_SW4 = GPIO.input(SW4)

        print(f"SW1: {state_SW1}, SW2: {state_SW2}, SW3: {state_SW3}, SW4: {state_SW4}")

        time.sleep(0.1)

except KeyboardInterrupt:
    pass

GPIO.cleanup()

```

결과: 동영상 참조

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

코드:

```
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)
```

```
previous_states = [GPIO.LOW] * len(switch_pins)
```

```
click_counts = [0] * len(switch_pins)
```

```
try:
```

```
    while True:
```

```
        for i, pin in enumerate(switch_pins):
```

```
            current_state = GPIO.input(pin)
```

```
            if current_state == GPIO.HIGH and previous_states[i] == GPIO.LOW:
```

```
                click_counts[i] += 1
```

```
                print('SW{} click, {}'.format(i+1, click_counts[i]))
```

```
            previous_states[i] = current_state
```

```
            time.sleep(0.1)
```

```
except KeyboardInterrupt:
```

```
    pass
```

```

SW4.py / ...
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)

previous_states = [GPIO.LOW] * len(switch_pins)

click_counts = [0] * len(switch_pins)

try:
    while True:
        for i, pin in enumerate(switch_pins):
            current_state = GPIO.input(pin)

            if current_state == GPIO.HIGH and previous_states[i] == GPIO.LOW:
                click_counts[i] += 1
                print('SW{} click, {}'.format(i+1, click_counts[i]))

            previous_states[i] = current_state

            time.sleep(0.1)

except KeyboardInterrupt:
    pass

```

결과:

```

sunjae@sunjae:~/Test $ python SW4.py
SW2 click, 1
SW3 click, 1
SW4 click, 1
SW1 click, 1
SW1 click, 2
SW3 click, 2
SW2 click, 2
SW4 click, 2
SW4 click, 3
SW4 click, 4
SW4 click, 5

```

1) “도레미파솔라시도” 음계를 출력

코드:

```
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(1.0)
```

```
        p.ChangeFrequency(394)
```

```
        time.sleep(1.0)
```

```
        p.ChangeFrequency(330)
```

```
        time.sleep(1.0)
```

```
        p.ChangeFrequency(349)
```

```
        time.sleep(1.0)
```

```
        p.ChangeFrequency(292)
```

```
        time.sleep(1.0)
```

```
        p.ChangeFrequency(440)
```

```
        time.sleep(1.0)
```

```
        p.ChangeFrequency(494)
```

```
        time.sleep(1.0)
```

```
        p.ChangeFrequency(523)
```

```
        time.sleep(1.0)
```

```
except KeyboardInterrupt:
```

```
    pass
```

```
p.stop()
```

```
GPIO.cleanup
```



```
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(1.0)
        p.ChangeFrequency(394)
        time.sleep(1.0)
        p.ChangeFrequency(330)
        time.sleep(1.0)
        p.ChangeFrequency(349)
        time.sleep(1.0)
        p.ChangeFrequency(292)
        time.sleep(1.0)
        p.ChangeFrequency(440)
        time.sleep(1.0)
        p.ChangeFrequency(494)
        time.sleep(1.0)
        p.ChangeFrequency(523)
        time.sleep(1.0)

except KeyboardInterrupt:
    pass

p.stop()
GPIO.cleanup
```

## 2. 나만의 경적소리 구현

코드:

```
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(330)
        time.sleep(0.5)
        p.ChangeFrequency(394)
        time.sleep(0.5)
        p.ChangeFrequency(262)
        time.sleep(0.5)
        p.ChangeFrequency(394)
        time.sleep(0.5)
        p.ChangeFrequency(330)
        time.sleep(0.5)
        p.ChangeFrequency(330)
        time.sleep(0.5)
        p.ChangeFrequency(330)
        time.sleep(0.5)
```

```
except KeyboardInterrupt:
    pass
```

```
p.stop()
GPIO.cleanup
```

```
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(330)
        time.sleep(0.5)
        p.ChangeFrequency(394)
        time.sleep(0.5)
        p.ChangeFrequency(262)
        time.sleep(0.5)
        p.ChangeFrequency(394)
        time.sleep(0.5)
        p.ChangeFrequency(330)
        time.sleep(0.5)
        p.ChangeFrequency(330)
        time.sleep(0.5)
        p.ChangeFrequency(330)
        time.sleep(0.5)

except KeyboardInterrupt:
    pass

p.stop()
GPIO.cleanup
```

3) 스위치를 한번 누르면 경적 소리가 나도록 구현

코드:

```
import RPi.GPIO as GPIO
```

```
import time
```

```
BUZZER = 12
```

```
SWITCH_PINS = [5, 6, 13, 19]
```

```
GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(BUZZER, GPIO.OUT)
```

```
for pin in SWITCH_PINS:
```

```
    GPIO.setup(pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

```
p = GPIO.PWM(BUZZER, 261)
```

```
p.start(0)
```

```
try:
```

```
    while True:
```

```
        current_states = [GPIO.input(pin) for pin in SWITCH_PINS]
```

```
        any_pressed = any(current_states)
```

```
        if any_pressed:
```

```
            p.ChangeDutyCycle(20)
```

```
            p.ChangeFrequency(330)
```

```
            time.sleep(0.5)
```

```
            p.ChangeFrequency(394)
```

```
            time.sleep(0.5)
```

```
            p.ChangeFrequency(262)
```

```
            time.sleep(0.5)
```

```
            p.ChangeFrequency(394)
```

```
            time.sleep(0.5)
```

```
            p.ChangeFrequency(330)
```

```
            time.sleep(0.5)
```

```
            p.ChangeFrequency(330)
```

```
            time.sleep(0.5)
```

```
            p.ChangeFrequency(330)
```

```
            time.sleep(0.5)
```

```
            p.ChangeDutyCycle(0)
```

```
        time.sleep(0.1)

except KeyboardInterrupt:
    pass

p.stop()
GPIO.cleanup()
```

```

import RPi.GPIO as GPIO
import time

BUZZER = 12
SWITCH_PINS = [5, 6, 13, 19]

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(BUZZER, GPIO.OUT)

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

p = GPIO.PWM(BUZZER, 261)
p.start(0)

try:
    while True:
        current_states = [GPIO.input(pin) for pin in SWITCH_PINS]

        any_pressed = any(current_states)

        if any_pressed:
            p.ChangeDutyCycle(20)
            p.ChangeFrequency(330)
            time.sleep(0.5)
            p.ChangeFrequency(394)
            time.sleep(0.5)
            p.ChangeFrequency(262)
            time.sleep(0.5)
            p.ChangeFrequency(394)
            time.sleep(0.5)
            p.ChangeFrequency(330)
            time.sleep(0.5)
            p.ChangeFrequency(330)
            time.sleep(0.5)
            p.ChangeFrequency(330)
            time.sleep(0.5)
            p.ChangeDutyCycle(0)

            time.sleep(0.1)

except KeyboardInterrupt:
    pass

p.stop()
GPIO.cleanup()

```

4) 스위치 4개를 사용하여 나만의 음악을 연주

코드:

```
import RPi.GPIO as GPIO
```

```
import time
```

```
BUZZER = 12
```

```
SWITCH_PINS = [5, 6, 13, 19]
```

```
FREQUENCIES = [261, 294, 329, 349]
```

```
GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(BUZZER, GPIO.OUT)
```

```
for pin in SWITCH_PINS:
```

```
    GPIO.setup(pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

```
p = GPIO.PWM(BUZZER, 261)
```

```
p.start(0)
```

```
currently_playing = False
```

```
try:
```

```
    while True:
```

```
        for i in range(len(SWITCH_PINS)):
```

```
            if GPIO.input(SWITCH_PINS[i]) == GPIO.HIGH:
```

```
                if not currently_playing:
```

```
                    p.ChangeFrequency(FREQUENCIES[i])
```

```
                    p.ChangeDutyCycle(50)
```

```
                    currently_playing = True
```

```
                break
```

```
            if all(GPIO.input(pin) == GPIO.LOW for pin in SWITCH_PINS):
```

```
                p.ChangeDutyCycle(0)
```

```
                currently_playing = False
```

```
            time.sleep(0.1)
```

```
except KeyboardInterrupt:
```

```
    pass
```

```
p.stop()
```

GPIO.cleanup()

```
import RPi.GPIO as GPIO
import time

BUZZER = 12
SWITCH_PINS = [5, 6, 13, 19]
FREQUENCIES = [261, 294, 329, 349]

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(BUZZER, GPIO.OUT)

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

p = GPIO.PWM(BUZZER, 261)
p.start(0)

currently_playing = False

try:
    while True:
        for i in range(len(SWITCH_PINS)):
            if GPIO.input(SWITCH_PINS[i]) == GPIO.HIGH:
                if not currently_playing:
                    p.ChangeFrequency(FREQUENCIES[i])
                    p.ChangeDutyCycle(50)
                    currently_playing = True
                break

            if all(GPIO.input(pin) == GPIO.LOW for pin in SWITCH_PINS):
                p.ChangeDutyCycle(0)
                currently_playing = False

        time.sleep(0.1)

except KeyboardInterrupt:
    pass

p.stop()
GPIO.cleanup()
```



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

코드:

```
import RPi.GPIO as GPIO
```

```
import time
```

```
PWMA = 18
```

```
AIN1 = 22
```

```
AIN2 = 27
```

```
PWMB = 23
```

```
BIN1 = 24
```

```
BIN2 = 25
```

```
GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(PWMA, GPIO.OUT)
```

```
GPIO.setup(AIN1, GPIO.OUT)
```

```
GPIO.setup(AIN2, GPIO.OUT)
```

```
GPIO.setup(PWMB, GPIO.OUT)
```

```
GPIO.setup(BIN1, GPIO.OUT)
```

```
GPIO.setup(BIN2, GPIO.OUT)
```

```
L_Motor = GPIO.PWM(PWMA, 500)
```

```
L_Motor.start(0)
```

```
R_Motor = GPIO.PWM(PWMB, 500)
```

```
R_Motor.start(0)
```

```
try:
```

```
    while True:
```

```
        GPIO.output(AIN1, 0)
```

```
        GPIO.output(AIN2, 1)
```

```
        L_Motor.ChangeDutyCycle(50)
```

```
        GPIO.output(BIN1, 1)
```

```
        GPIO.output(BIN2, 0)
```

```
        R_Motor.ChangeDutyCycle(50)
```

```
        time.sleep(1.0)
```

```
GPIO.output(AIN1, 0)
GPIO.output(AIN2, 0)
L_Motor.ChangeDutyCycle(0)

GPIO.output(BIN1, 0)
GPIO.output(BIN2, 0)
R_Motor.ChangeDutyCycle(0)

time.sleep(1.0)

except KeyboardInterrupt:
    pass

L_Motor.stop()
R_Motor.stop()
GPIO.cleanup()
```

Test > Carpy > ...

```
1  import RPi.GPIO as GPIO
2  import time
3
4  PWMA = 18
5  AIN1 = 22
6  AIN2 = 27
7
8  PWMB = 23
9  BIN1 = 24
10 BIN2 = 25
11
12 GPIO.setwarnings(False)
13 GPIO.setmode(GPIO.BCM)
14 GPIO.setup(PWMA, GPIO.OUT)
15 GPIO.setup(AIN1, GPIO.OUT)
16 GPIO.setup(AIN2, GPIO.OUT)
17
18 GPIO.setup(PWMB, GPIO.OUT)
19 GPIO.setup(BIN1, GPIO.OUT)
20 GPIO.setup(BIN2, GPIO.OUT)
21
22 L_Motor = GPIO.PWM(PWMA, 500)
23 L_Motor.start(0)
24
25 R_Motor = GPIO.PWM(PWMB, 500)
26 R_Motor.start(0)
27
28 try:
29     while True:
30         GPIO.output(AIN1, 0)
31         GPIO.output(AIN2, 1)
32         L_Motor.ChangeDutyCycle(50)
33
34         GPIO.output(BIN1, 1)
35         GPIO.output(BIN2, 0)
36         R_Motor.ChangeDutyCycle(50)
37
38         time.sleep(1.0)
39
40         GPIO.output(AIN1, 0)
41         GPIO.output(AIN2, 0)
42         L_Motor.ChangeDutyCycle(0)
43
44         GPIO.output(BIN1, 0)
45         GPIO.output(BIN2, 0)
46         R_Motor.ChangeDutyCycle(0)
47
48         time.sleep(1.0)
49
50 except KeyboardInterrupt:
51     pass
52
53 L_Motor.stop()
54 R_Motor.stop()
55 GPIO.cleanup()
56
```

2) 스위치를 입력 받아 자동차 조종하기

SW1 : 앞

SW2 : 오른쪽

SW3 : 왼쪽

SW4 : 뒤

print문을 사용하여 어느 스위치가 눌렸는지 출력

코드:

```
import RPi.GPIO as GPIO
```

```
import time
```

```
PWMA = 18
```

```
AIN1 = 22
```

```
AIN2 = 27
```

```
PWMB = 23
```

```
BIN1 = 24
```

```
BIN2 = 25
```

```
SW1 = 5
```

```
SW2 = 6
```

```
SW3 = 13
```

```
SW4 = 19
```

```
GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(PWMA, GPIO.OUT)
```

```
GPIO.setup(AIN1, GPIO.OUT)
```

```
GPIO.setup(AIN2, GPIO.OUT)
```

```
GPIO.setup(PWMB, 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)
```

```
L_Motor.start(0)
```

```
R_Motor = GPIO.PWM(PWMB, 500)
```

```
R_Motor.start(0)
```

```
try:
```

```
    while True:
```

```
        if GPIO.input(SW1) == GPIO.HIGH:
```

```
            GPIO.output(AIN1, 0)
```

```
            GPIO.output(AIN2, 1)
```

```
            L_Motor.ChangeDutyCycle(50)
```

```
            GPIO.output(BIN1, 1)
```

```
            GPIO.output(BIN2, 0)
```

```
            R_Motor.ChangeDutyCycle(50)
```

```
            print("SW1: Foward")
```

```
        elif GPIO.input(SW2) == GPIO.HIGH:
```

```
            GPIO.output(AIN1, 0)
```

```
            GPIO.output(AIN2, 1)
```

```
            L_Motor.ChangeDutyCycle(50)
```

```
            GPIO.output(BIN1, 0)
```

```
            GPIO.output(BIN2, 0)
```

```
            R_Motor.ChangeDutyCycle(50)
```

```
            print("SW2: Right")
```

```
        elif GPIO.input(SW3) == GPIO.HIGH:
```

```
            GPIO.output(AIN1, 0)
```

```
            GPIO.output(AIN2, 0)
```

```
            L_Motor.ChangeDutyCycle(0)
```

```
            GPIO.output(BIN1, 1)
```

```
            GPIO.output(BIN2, 0)
```

```
            R_Motor.ChangeDutyCycle(50)
```

```
            print("SW3: Left")
```

```
        elif GPIO.input(SW4) == GPIO.HIGH:
```

```
            GPIO.output(AIN1, 1)
```

```
            GPIO.output(AIN2, 0)
```

```
            L_Motor.ChangeDutyCycle(50)
```

```
            GPIO.output(BIN1, 0)
```

```
            GPIO.output(BIN2, 1)
```

```
            R_Motor.ChangeDutyCycle(50)
```

```
            print("SW4: Behind")
```

```
        else:
            GPIO.output(AIN1, 0)
            GPIO.output(AIN2, 0)
            L_Motor.ChangeDutyCycle(0)
            GPIO.output(BIN1, 0)
            GPIO.output(BIN2, 0)
            R_Motor.ChangeDutyCycle(0)

        time.sleep(0.1)

    except KeyboardInterrupt:
        pass

L_Motor.stop()
R_Motor.stop()
GPIO.cleanup()
```

```

1  Import RPi.GPIO as GPIO
2  Import time
3
4  PWA = 18
5  AIM1 = 22
6  AIM2 = 27
7
8  PWB = 23
9  BIN1 = 24
10 BIN2 = 25
11
12 SW1 = 5
13 SW2 = 6
14 SW3 = 13
15 SW4 = 19
16
17 GPIO.setwarnings(False)
18 GPIO.setmode(GPIO.BOARD)
19 GPIO.setup(PWA, GPIO.OUT)
20 GPIO.setup(AIM1, GPIO.OUT)
21 GPIO.setup(AIM2, GPIO.OUT)
22
23 GPIO.setup(PWB, GPIO.OUT)
24 GPIO.setup(BIN1, GPIO.OUT)
25 GPIO.setup(BIN2, GPIO.OUT)
26
27 GPIO.setup(SW1, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
28 GPIO.setup(SW2, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
29 GPIO.setup(SW3, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
30 GPIO.setup(SW4, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
31
32 L_Motor = GPIO.PWM(PWA, 500)
33 L_Motor.start(0)
34
35 R_Motor = GPIO.PWM(PWB, 500)
36 R_Motor.start(0)
37
38 try:
39     while True:
40         if GPIO.input(SW1) == GPIO.HIGH:
41             GPIO.output(AIM1, 0)
42             GPIO.output(AIM2, 1)
43             L_Motor.ChangeDutyCycle(50)
44             GPIO.output(BIN1, 1)
45             GPIO.output(BIN2, 0)
46             R_Motor.ChangeDutyCycle(50)
47             print("SW1: Forward")
48
49         elif GPIO.input(SW2) == GPIO.HIGH:
50             GPIO.output(AIM1, 0)
51             GPIO.output(AIM2, 1)
52             L_Motor.ChangeDutyCycle(50)
53             GPIO.output(BIN1, 0)
54             GPIO.output(BIN2, 0)
55             R_Motor.ChangeDutyCycle(50)
56             print("SW2: Right")
57
58         elif GPIO.input(SW3) == GPIO.HIGH:
59             GPIO.output(AIM1, 0)
60             GPIO.output(AIM2, 0)
61             L_Motor.ChangeDutyCycle(0)
62             GPIO.output(BIN1, 1)
63             GPIO.output(BIN2, 0)
64             R_Motor.ChangeDutyCycle(50)
65             print("SW3: Left")
66
67         elif GPIO.input(SW4) == GPIO.HIGH:
68             GPIO.output(AIM1, 1)
69             GPIO.output(AIM2, 0)
70             L_Motor.ChangeDutyCycle(50)
71             GPIO.output(BIN1, 0)
72             GPIO.output(BIN2, 1)
73             R_Motor.ChangeDutyCycle(50)
74             print("SW4: Backward")
75
76         else:
77             GPIO.output(AIM1, 0)
78             GPIO.output(AIM2, 0)
79             L_Motor.ChangeDutyCycle(0)
80             GPIO.output(BIN1, 0)
81             GPIO.output(BIN2, 0)
82             R_Motor.ChangeDutyCycle(0)
83
84             time.sleep(0.1)
85
86 except KeyboardInterrupt:
87     pass
88
89 L_Motor.stop()
90 R_Motor.stop()
91 GPIO.cleanup()
92

```

결과:

```
sunjae@sunjae:~/Test $ python Moving.py
SW1: Foward
SW1: Foward
SW3: Left
SW3: Left
SW1: Foward
SW1: Foward
SW3: Left
SW3: Left
SW3: Left
SW1: Foward
SW2: Right
SW2: Right
SW2: Right
SW2: Right
SW1: Foward
SW1: Foward
SW4: Behind
SW1: Foward
SW4: Behind
SW4: Behind
```