임베디드 응용 및 실습 7주차 과제

2020161103 장민준

버튼 입력 받기 구현

1)

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

GPIO.cleanup()

2)

import RPi.GPIO as GPIO

import time

SW1, SW2, SW3, SW4 = 5, 6, 13, 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:

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

            print("click SW1")

            while GPIO.input(SW1) == GPIO.HIGH:

                time.sleep(0.1)

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

            print("click SW2")

            while GPIO.input(SW2) == GPIO.HIGH:

                time.sleep(0.1)

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

            print("click SW3")

            while GPIO.input(SW3) == GPIO.HIGH:

                time.sleep(0.1)

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

            print("click SW4")

            while GPIO.input(SW4) == GPIO.HIGH:

                time.sleep(0.1)

        time.sleep(0.1)

except KeyboardInterrupt:

    pass

GPIO.cleanup()

3)4)

import RPi.GPIO as GPIO

import time

SW1, SW2, SW3, SW4 = 5, 6, 13, 19

click\_counts = [0, 0, 0, 0]

log = []

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:

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

            click\_counts[0] += 1

            log.append(("SW1 click", click\_counts[0]))

            print(log[-1])

            while GPIO.input(SW1) == GPIO.HIGH:

                time.sleep(0.1)

        if GPIO.input(SW2) == GPIO.HIGH:

            click\_counts[1] += 1

            log.append(("SW2 click", click\_counts[1]))

            print(log[-1])

            while GPIO.input(SW2) == GPIO.HIGH:

                time.sleep(0.1)

        if GPIO.input(SW3) == GPIO.HIGH:

            click\_counts[2] += 1

            log.append(("SW3 click", click\_counts[2]))

            print(log[-1])

            while GPIO.input(SW3) == GPIO.HIGH:

                time.sleep(0.1)

        if GPIO.input(SW4) == GPIO.HIGH:

            click\_counts[3] += 1

            log.append(("SW4 click", click\_counts[3]))

            print(log[-1])

            while GPIO.input(SW4) == GPIO.HIGH:

                time.sleep(0.1)

        time.sleep(0.1)

except KeyboardInterrupt:

    pass

GPIO.cleanup()

부저 음계 출력 구현 과제

1)

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, 1)

tones = {

    'C': 262,

    'D': 294,

    'E': 330,

    'F': 349,

    'G': 392,

    'A': 440,

    'B': 494,

    'C\_high': 523

}

melody = ['C', 'D', 'E', 'F', 'G', 'A', 'B', 'C\_high']

try:

    p.start(50)

    for note in melody:

        p.ChangeFrequency(tones[note])

        time.sleep(0.5)

    p.stop()

except KeyboardInterrupt:

    pass

GPIO.cleanup()

2)

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, 1)

tones = {

    'C': 262,

    'D': 294,

    'E': 330,

    'F': 349,

    'G': 392,

    'A': 440,

    'B': 494,

    'C\_high': 523

}

melody = [

    ('F', 0.25), ('G', 0.25),

    ('A', 0.25), ('F', 0.25), ('C\_high', 0.75),

    ('A', 0.25), ('G', 0.5), ('C\_high', 0.5), ('G', 0.5)

]

try:

    p.start(50)

    for note, duration in melody:

        p.ChangeFrequency(tones[note])

        time.sleep(duration)

    p.stop()

except KeyboardInterrupt:

    pass

GPIO.cleanup()

3)

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, 1)

tones = {

    'C': 262,

    'D': 294,

    'E': 330,

    'F': 349,

    'G': 392,

    'A': 440,

    'B': 494,

    'C\_high': 523

}

melody = [

    ('F', 0.25), ('G', 0.25),

    ('A', 0.25), ('F', 0.25), ('C\_high', 0.75),

    ('A', 0.25), ('G', 0.5), ('C\_high', 0.5), ('G', 0.5)

]

def play\_melody():

    p.start(50)

    for note, duration in melody:

        p.ChangeFrequency(tones[note])

        time.sleep(duration)

    p.stop()

try:

    while True:

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

            print("SW1 pressed: playing melody")

            play\_melody()

            time.sleep(0.5)  # debounce delay

except KeyboardInterrupt:

    pass

GPIO.cleanup()

4)

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, 1)

tones = {

    'C': 262,

    'D': 294,

    'E': 330,

    'F': 349,

    'G': 392,

    'A': 440,

    'B': 494,

    'C\_high': 523,

    'D\_high': 587

}

melody = [

    ('E', 0.5), ('E', 0.5), ('A', 0.5), ('G', 0.5),

    ('E', 0.5), ('D', 0.5), ('G', 0.5), ('A', 0.5),

    ('E', 0.5), ('E', 0.5), ('D', 0.5), ('G', 0.5),

    ('A', 0.5), ('G', 0.5), ('E', 0.5), ('D', 0.5)

]

def play\_tone(note, duration):

    p.start(50)

    p.ChangeFrequency(tones[note])

    time.sleep(duration)

    p.stop()

try:

    while True:

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

            print("SW1 pressed: playing melody part 1")

            play\_tone('B', 0.25)

            play\_tone('A', 0.25)

            play\_tone('G', 0.25)

            play\_tone('B', 0.25)

            play\_tone('D\_high', 0.75)

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

            print("SW2 pressed: playing melody part 2")

            play\_tone('B', 0.25)

            play\_tone('A', 0.25)

            play\_tone('G', 0.25)

            play\_tone('B', 0.25)

            play\_tone('A', 0.75)

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

            print("SW3 pressed: playing melody part 3")

            play\_tone('B', 0.25)

            play\_tone('A', 0.25)

            play\_tone('G', 0.25)

            play\_tone('A', 0.25)

            play\_tone('A', 0.25)

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

            print("SW4 pressed: playing melody part 4")

            play\_tone('G', 0.25)

            play\_tone('A', 0.25)

            play\_tone('G', 0.25)

            play\_tone('B', 0.25)

            play\_tone('G', 0.25)

        time.sleep(0.1)  # debounce delay

except KeyboardInterrupt:

    pass

GPIO.cleanup()

자동차 움직이기 구현

1)

import RPi.GPIO as GPIO

import time

PWMA = 18

AIN1 = 22

AIN2 = 27

PWMB = 23

BIN1 = 25

BIN2 = 24

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)

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(1.0)

        L\_Motor.ChangeDutyCycle(0)

        R\_Motor.ChangeDutyCycle(0)

        time.sleep(1.0)

except KeyboardInterrupt:

    pass

GPIO.cleanup()

2)

import RPi.GPIO as GPIO

import time

PWMA = 18

AIN1 = 22

AIN2 = 27

PWMB = 23

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(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)

R\_Motor = GPIO.PWM(PWMB, 500)

L\_Motor.start(0)

R\_Motor.start(0)

def move\_forward():

    GPIO.output(AIN1, 0)

    GPIO.output(AIN2, 1)

    GPIO.output(BIN1, 0)

    GPIO.output(BIN2, 1)

    L\_Motor.ChangeDutyCycle(100)

    R\_Motor.ChangeDutyCycle(100)

def move\_backward():

    GPIO.output(AIN1, 1)

    GPIO.output(AIN2, 0)

    GPIO.output(BIN1, 1)

    GPIO.output(BIN2, 0)

    L\_Motor.ChangeDutyCycle(100)

    R\_Motor.ChangeDutyCycle(100)

def move\_left():

    GPIO.output(AIN1, 0)

    GPIO.output(AIN2, 1)

    GPIO.output(BIN1, 1)

    GPIO.output(BIN2, 0)

    L\_Motor.ChangeDutyCycle(50)

    R\_Motor.ChangeDutyCycle(50)

def move\_right():

    GPIO.output(AIN1, 1)

    GPIO.output(AIN2, 0)

    GPIO.output(BIN1, 0)

    GPIO.output(BIN2, 1)

    L\_Motor.ChangeDutyCycle(50)

    R\_Motor.ChangeDutyCycle(50)

def stop\_motor():

    L\_Motor.ChangeDutyCycle(0)

    R\_Motor.ChangeDutyCycle(0)

try:

    while True:

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

            print("SW1 go")

            move\_forward()

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

            print("SW2 right")

            move\_right()

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

            print("SW3 left")

            move\_left()

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

            print("SW4 backward")

            move\_backward()

        else:

            stop\_motor()

        time.sleep(0.1)

except KeyboardInterrupt:

    pass

GPIO.cleanup()