|  |
| --- |
| **버튼 입력 받기 구현** |
| #-\*- coding:utf-8 -\*-  import RPi.GPIO as GPIO  import time  sw = [5, 6, 13, 19] # pin 번호 배열  cnt = [0, 0, 0, 0] # 스위치 별 count 횟수 배열  GPIO.setmode(GPIO.BCM) # GPIO를 BCM 모드로 설정  map(lambda a: GPIO.setup(a, GPIO.IN, pull\_up\_down = GPIO.PUD\_DOWN),sw) # 스위치 세팅  last = [0, 0, 0, 0] # 직전 스위치 값 저장 배열  curr = [0, 0, 0, 0] # 현재 스위치 값 저장 배열  try:  while True:  for i in range(4):  curr[i] = GPIO.input(sw[i]) # 현재 스위치 값 저장  if last[i] != curr[i]: # 현재 값과 직전 값을 비교해서  # 현재 값과 직전값이 다를 때  if curr[i]: # 0->1 경우에만  cnt[i] += 1 # count한다  print('SW{} clicked! count : {}'.format(i+1, cnt[i]))  last[i] = curr[i] # 직전 값을 현재 값으로 update  except KeyboardInterrupt:  pass  GPIO.cleanup() |
|  |
| **부저 음계 출력 구현** |
| #-\*- coding:utf-8 -\*-  import RPi.GPIO as GPIO  import time  sw = [5, 6, 13, 19] # pin 번호 배열  fre = [261, 239, 329, 391] # 스위치 별 출력 주파수  buzzer = 12 # buzzer pin 번호  GPIO.setmode(GPIO.BCM) # GPIO를 BCM 모드로 설정  GPIO.setup(buzzer, GPIO.OUT) # buzzer 세팅  map(lambda a: GPIO.setup(a, GPIO.IN, pull\_up\_down = GPIO.PUD\_DOWN),sw) # 스위치 세팅  p = GPIO.PWM(buzzer, 261)  last = [0, 0, 0, 0] # 직전 스위치 값 저장 배열  curr = [0, 0, 0, 0] # 현재 스위치 값 저장 배열  try:  while True:  for i in range(4):  curr[i] = GPIO.input(sw[i]) # 현재 스위치 값 저장  if last[i] != curr[i]: # 현재 값과 직전 값을 비교해서  # 현재 값과 직전값이 다를 때  if curr[i]: # 0->1 경우에만  p.start(50)  p.ChangeFrequency(fre[i]) # 해당 주파수 출력  else: p.stop()  last[i] = curr[i] # 직전 값을 현재 값으로 update    except KeyboardInterrupt:  pass  GPIO.cleanup() |

|  |
| --- |
| **자동차 움직이기 구현** |
| #-\*- coding:utf-8 -\*-  import RPi.GPIO as GPIO  import time  sw = [5, 6, 13, 19] # pin 번호 배열  PWM = [18, 23] #  Min = [22, 27, 25, 24] # 왼쪽 모터 출력 PWM, IN1, IN2  # 오른쪽 모터 출력 PWM, IN1, IN2  GPIO.setmode(GPIO.BCM) # GPIO를 BCM 모드로 설정  map(lambda a: GPIO.setup(a, GPIO.IN, pull\_up\_down = GPIO.PUD\_DOWN),sw) # 스위치 세팅  map(lambda a: GPIO.setup(a, GPIO.OUT), PWM) # Motor PWM 핀 세팅  map(lambda a: GPIO.setup(a, GPIO.OUT), Min) # Motor 입력 핀 세팅  last = [0, 0, 0, 0] # 직전 스위치 값 저장 배열  curr = [0, 0, 0, 0] # 현재 스위치 값 저장 배열  # 진행 방향에 따라 입력할 RIN1, RIN2, LIN1, LIN2, Rspeed, Lspeed  GO = [0, 1, 0, 1, 50, 50]  RIGHT = [0, 1, 1, 0, 100, 40]  LEFT = [1, 0, 0, 1, 40, 100]  BACK = [1, 0, 1, 0, 50, 50]  STOP = [0, 1, 0, 1, 0, 0]  R\_Motor = GPIO.PWM(PWM[0], 500)  R\_Motor.start(0)  L\_Motor = GPIO.PWM(PWM[1], 500)  L\_Motor.start(0)  # list 입력받아 출력  def PrintMotor(dir):  print(dir)  for i in range(4):  GPIO.output(Min[i], dir[i])  R\_Motor.ChangeDutyCycle(dir[4])  L\_Motor.ChangeDutyCycle(dir[5])  try:  while True:  for i in range(4):  curr[i] = GPIO.input(sw[i]) # 현재 스위치 값 저장  if last[i] != curr[i]: # 현재 값과 직전 값을 비교해서  # 현재 값과 직전값이 다를 때  if curr[i]: # 0->1 경우에만  if i == 0: # 1번 스위치가 눌림  PrintMotor(GO)  elif i == 1: # 2번 스위치가 눌림  PrintMotor(RIGHT)  elif i == 2: # 3번 스위치가 눌림  PrintMotor(LEFT)  elif i == 3: # 4번 스위치가 눌림  PrintMotor(BACK)  else:  PrintMotor(STOP) # 스위치가 눌리지 않으면 정지  last[i] = curr[i] # 직전 값을 현재 값으로 update  except KeyboardInterrupt:  GPIO.cleanup()  pass |