

San Francisco Bay University

CE450 Fundamentals of Embedded Engineering Lab 4 Buzzer Control

Objectives:

In this lab, students will learn how to control on-board buzzer in Python program on Raspberry Pi board and do hands-on exercise through lab assignments.

Introduction:

One buzzer is available in Sunfounder accessory box. If a control signal generated from one of pins in GPIO port is logic low, it will switch buzzer on through a BJT, otherwise, off.

Equipment:

The equipment you require is as follows:

- Laptop & Raspberry Pi 4B model Board
- SunFounder Super Starter Kit V2.0 for Raspberry Pi

The Laboratory Procedure:

1. Hardware connection

Pin#	NAME		NAME	Pin#		
01	3.3v DC Power	0	DC Power 5v	02		
03	GPIO02 (SDA1, I2C)	00	DC Power 5v	04		V <u>C</u> C
05	GPIO03 (SCL1, I2C)	00	Ground	06		
07	GPIO04 (GPIO_GCLK)	00	(TXD0) GPIO14	08		
09	Ground	00	(RXD0) GPIO15	10		
11	GPIO17 (GPIO_GEN0)	00	(GRIO_GEN1) GPIO18	12	GPIO0 R	
13	GPIO27 (GPIO_GEN2)	00-	Ground	14		'
15	GPIO22 (GPIO_GEN3)	00	(GPIO_GEN4) GPIO23	16		
17	3.3v DC Power	00	(GPIO_GEN5) GPIO24	18		GND
19	GPIO10 (SPI_MOSI)	00	Ground	20		
21	GPIO09 (SPI_MISO)	00	(GPIO_GEN6) GPIO25	22		
23	GPIO11 (SPI_CLK)	00	(SPI_CE0_N) GPIO08	24		
25	Ground	00	(SPI_CE1_N) GPIO07	26		
27	ID_SD (I2C ID EEPROM)	00	(I2C ID EEPROM) ID_SC	28		
29	GPIO05	00	Ground	30		
31	GPIO06	00	GPIO12	32		
33	GPIO13	00	Ground	34		
35	GPIO19	00	GPIO16	36		
37	GPIO26	00	GPIO20	38		
39	Ground	00	GPIO21	40		

2. Control program in Python

```
# Python program
import RPi.GPIO as GPIO
import time
BeepPin = 11
                # pin11
def setup():
         GPIO.setmode(GPIO.BOARD)
                                         # Numbers GPIOs by physical location
         GPIO.setup(BeepPin, GPIO.OUT) # Set BeepPin's mode is output
         GPIO.output(BeepPin, GPIO.HIGH) # Set BeepPin high(+3.3V) to off beep
def loop():
         while True:
                  GPIO.output(BeepPin, GPIO.LOW)
                                                      # Switch on Buzzer
                                                      # 0.1s delay
                  time.sleep(0.1)
                  GPIO.output(BeepPin, GPIO.HIGH)
                  time.sleep(0.1)
def destroy():
         GPIO.output(BeepPin, GPIO.HIGH)
                                            # beep off
         GPIO.cleanup()
                                            # Release resource
print 'Press Ctrl+C to end the program...'
setup()
try:
         1000()
except
        KeyboardInterrupt: # When 'Ctrl+C' is pressed, the child program destroy()
will be executed.
         destroy()
         *Note: Hardware connection reference and running command
         https://learn.sunfounder.com/lesson-6-buzzer/
         https://learn.sunfounder.com/category/super-kit-v3-0-for-raspberry-pi/
```

The Laboratory Assignments:

- 1. Implement the buzzer control based on the above example program
- 2. Add one LED to the buzzer control circuit to make it on when the buzzer is ringing and off if the buzzer is silent

```
# Python program
import RPi.GPIO as GPIO
import time
BeepPin = 11
                # pin11
ledPin = 15
                 # pin15
                           Led one pin to GND, another to GPIO
on = True
off = False
def setup():
                                           # Numbers GPIOs by physical location
          GPIO.setmode(GPIO.BOARD)
          GPIO.setup(BeepPin, GPIO.OUT) # Set BeepPin's mode is output
          GPIO.output(BeepPin, GPIO.HIGH) # Set BeepPin high(+3.3V) to off beep
          GPIO.setup(ledPin, GPIO.OUT)
                                            # set ledPin's mode is output
          GPIO.output(ledPin, GPIO.LOW)
                                             # set ledPin low to off the led
def loop():
          while True:
                   buzzer ctrl(on)
                    time.sleep(0.1)
                   buzzer ctrl(off)
                    time.sleep(0.1)
def buzzer_ctrl(switch):
          if(switch): # switch on buzzer to make both beep and led on
                   GPIO.output(BeepPin, GPIO.LOW) # Switch on Buzzer
GPIO.output(ledPin, GPIO.HIGH) # set ledPin high(+3.3) to on led
          else: # switch off buzzer to make both beep and led off
                    GPIO.output(BeepPin, GPIO.HIGH)
                   GPIO.output(ledPin, GPIO.LOW)
def destroy():
                                              # beep off
          GPIO.output(BeepPin, GPIO.HIGH)
          GPIO.output(ledPin, GPIO.LOW)
                                               # led off
          GPIO.cleanup()
                                               # Release resource
print 'Press Ctrl+C to end the program...'
setup()
try:
          loop()
          KeyboardInterrupt: # When 'Ctrl+C' is pressed, the child program
destroy() will be executed.
          destroy()
```