Lab 3: Active Buzzer & Ultrasonic Ranging Module

Noah Buchanan
Derek Yocum
CS 4363: IoT Development
University of Arkansas - Fort Smith
Fall 2021

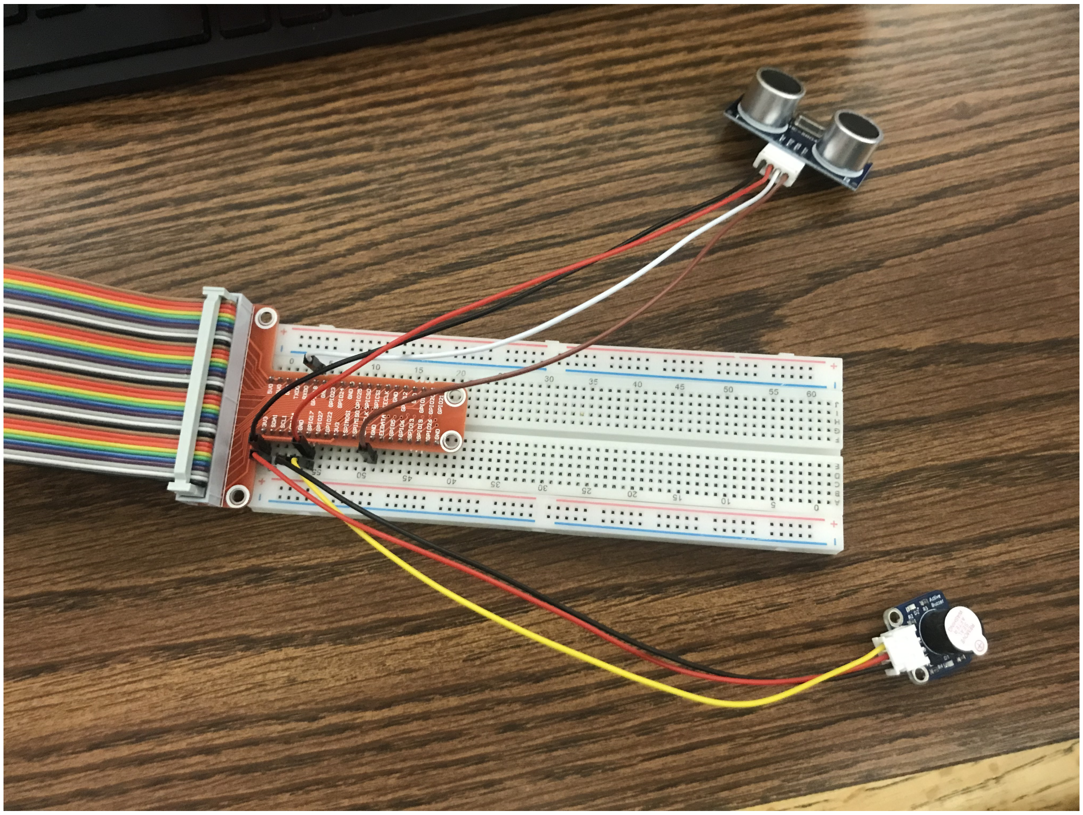
October 7, 2021

Documentation

The only changes made were to the original Ultra Sonic Ranging Modules source code. We took all the necessary logic from the Active Buzzer's source code and utilized it in the Ranging Module's code. The changes are as follows:

1. Copy on(), off(), and beep() directly from the Active Buzzers code.
2. Modify the loop to call beep() when the distance measured is less than 30 cm(It is actually picking something up and not just random echos from around the room).
3. Modify the setup() function to incorporate any initialization that the active buzzer needs from its python file that was not originally included in the Ranging Module's python file.
4. Lastly, modify the destroy() function to turn off the active buzzer properly as well and not just the Ranging Module.

Photo of Breadboard



Source Code

```
#!/usr/bin/env python3

import RPi.GPIO as GPIO
import time

TRIG = 11
ECHO = 12

def setup():
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(ECHO, GPIO.IN)
        GPIO.setup(TRIG, GPIO.OUT)
        GPIO.output(TRIG, GPIO.HIGH)
```

```
def distance():
        GPIO.output(TRIG, 0)
        time.sleep(0.000002)

        GPIO.output(TRIG, 1)
        time.sleep(0.00001)
        GPIO.output(TRIG, 0)

        while GPIO.input(ECHO) == 0:
                a = 0
        time1 = time.time()
        while GPIO.input(ECHO) == 1:
                a = 1
        time2 = time.time()

        during = time2 - time1
        return during * 340 / 2 * 100

def loop():
        while True:
                dis = distance()
                print (int(dis), 'cm')
                print ('')
                if dis < 20:
                        beep(0.5)
                off()
                time.sleep(0.5)
def on():
        GPIO.output(TRIG, GPIO.LOW)

def off():
        GPIO.output(TRIG, GPIO.HIGH)

def beep(x):
        on()
        time.sleep(x)
        off()
        time.sleep(x)

def destroy():
        GPIO.output(TRIG, GPIO.HIGH)
        GPIO.cleanup()

if __name__ == "__main__":
        setup()
        off()
```

```
try:
        loop()
except KeyboardInterrupt:
        destroy()
```