42 ELECTRONICS PRESENTS:

Intro to Robotics

Level B: Working with Sensors & Intermediate Programming

v19325

By Eric Feickert and Julie Feickert

The GPIO pin numbering is set to BCM, each of the pins listed in **`rgb`** is designated as an output, and pin 21 is declared as an input. The try/except format is used to catch keyboard exceptions and a **`GPIO.cleanup`** runs if an exception is encountered.

The **`try`** loop uses a **`while True:`** loop to keep checking the input over and over. If the input on **`GPIO21`** is low or False, then the green element will turn off and red will illuminate. The else condition will trigger if **`GPIO21`** is high or True during a check. In that case the red element will turn off and the green element will turn on. The **`time.sleep`** command is added to slow the loop checks down to avoid unnecessarily checking of the GPIO which can lead to unnecessary load, and heat, on the processor of the Raspberry Pi.

STEP #10

Run the program. The LED will initially be green to indicate that **`GPIO21`** is high or 3.3V coming from the level shifter IC.

Carefully disconnect the jumper wire at location P2-37 and the input will go low, indicated by the LED turning red.

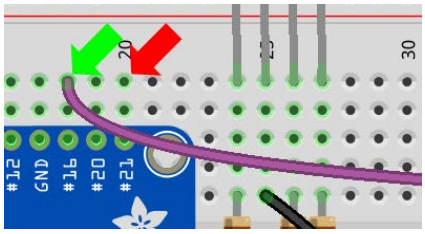
Reconnect the jumper wire to P2-37 and the LED will change back to green to indicate a high is present on the input.

Remove and reinstall the connection at P2-37 a few more times to ensure the program and circuit are acting as expected. If so, feel free to proceed to the next activity where you will reconfigure this circuit to get 5V input from an IR Obstacle sensor.

If the circuit or program is not behaving as outlined above, shut down the Pi and recheck all wiring and the IC's pin 1 orientation. If everything looks good on the circuitry, then proceed to reboot the Pi and double-check your program. Do not continue to the next activity until this program and circuit are behaving as expected.

STEP #3

The ultrasonic.py file that will be used to capture distance readings will be looking for echo signals on GPIO21, but the slide switch is currently in that position. Move the slide switch connection from GPIO21 over to GPIO16:



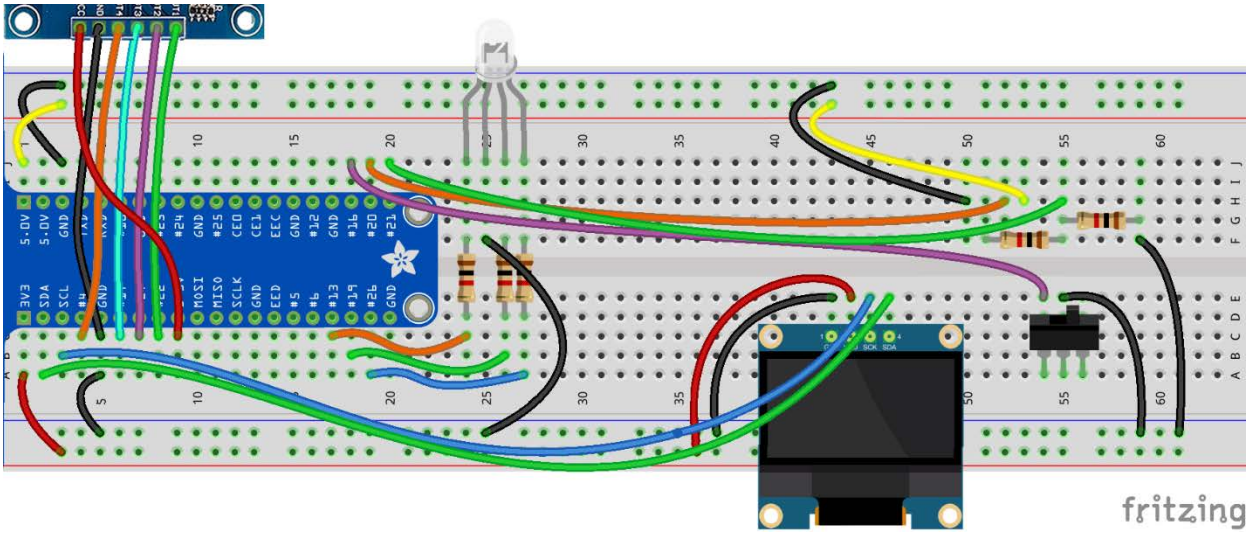
The voltage divider will take care of bringing the Echo line from the ultrasonic range sensor down to a safe level, but you will need to make a few more connections before you can use it for ranges. Make the following four connections between the points listed below:

Short jumper wire – 5V – between H53 and P2-43

Short jumper wire - GND – between H50 and N2-42

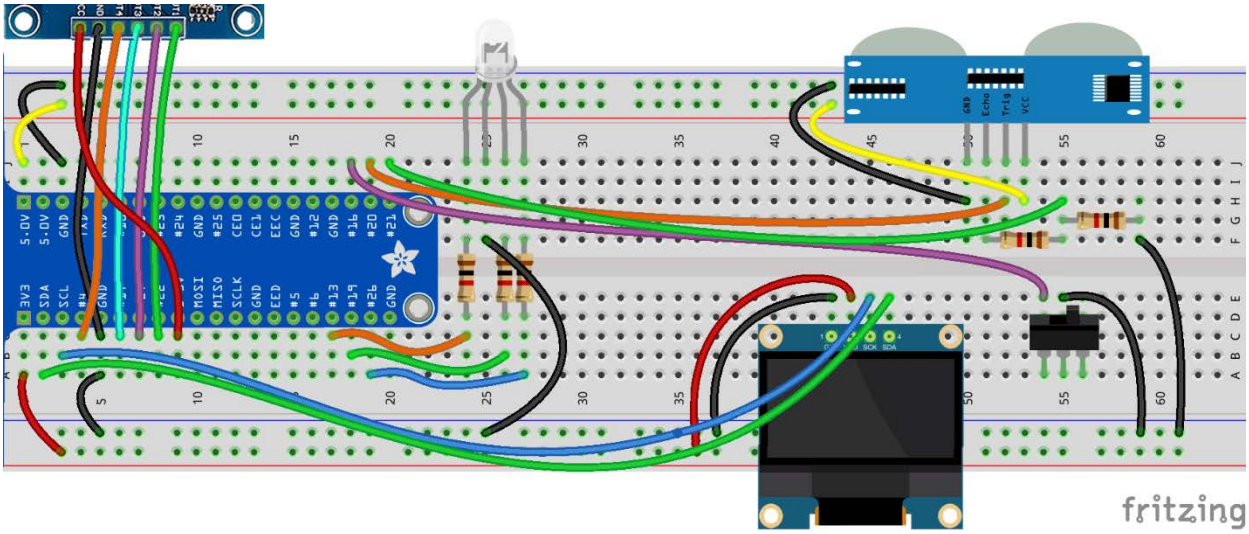
Long jumper wire – Trigger – between H52 and J19

Long jumper wire – Echo – between H55 and J20



STEP #4

The last step is to add the ultrasonic range sensor. Connect it to the breadboard at J50 through J53 with the sensor pointing away from the OLED display. Ensure the sensor is properly oriented and connected to the correct locations, or it could be damaged:



Power the Raspberry Pi on so it can be used to create a program to use with your new circuit.

STEP #2

The first area in the program will be the imports and there are quite a few. You will be using **time**, **RPi.GPIO**, **random**, **ultrasonic**, **Adafruit_SSD1306**, and the **PIL** imports you used for the OLED display. Here are the import lines to add:

```
import time, RPi.GPIO as GPIO, random, ultrasonic, Adafruit_SSD1306
from PIL import Image, ImageDraw, ImageFont
```

STEP #3

Next, you will assign the input and output pins using a lot of variables.

This will help you later if you want to modify this program and move an input or output to another pin. This list will contain pin assignments for the slide switch, RGB elements, and Capacitive Touch inputs.

The lists for **rgb** and **cap** will allow those groups of pins to be configured as inputs and outputs as a group, using only one line each:

```
import time, RPi.GPIO as GPIO, random, ultrasonic, Adafruit_SSD1306
from PIL import Image, ImageDraw, ImageFont

slide = 16
rgb = [13,19,26]
red = 13
green = 19
blue = 26
cap = [22,27,17,4]
cap1 = 22
cap2 = 27
cap3 = 17
cap4 = 4
```

```
import time, RPi.GPIO as GPIO, random, ultrasonic, Adafruit_SSD1306
from PIL import Image, ImageDraw, ImageFont

slide = 16
rgb = [13,19,26]
red = 13
green = 19
blue = 26
cap = [22,27,17,4]
cap1 = 22
cap2 = 27
cap3 = 17
cap4 = 4

GPIO.setmode(GPIO.BCM)
GPIO.setup(slide, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(rgb, GPIO.OUT)
GPIO.setup(cap, GPIO.IN)

disp = Adafruit_SSD1306.SSD1306_128_64(rst=None)
disp.begin()
width = disp.width
height = disp.height
image = Image.new('1', (width, height))
draw = ImageDraw.Draw(image)
font = ImageFont.truetype('/usr/share/fonts/truetype/freefont/FreeSans.ttf',18)

def update_display():
    disp.image(image)
    disp.display()
```

STEP #16

Double check your program and indentation against the code below:

```
import time, RPi.GPIO as GPIO, random, ultrasonic, Adafruit_SSD1306
from PIL import Image, ImageDraw, ImageFont

slide = 16
rgb = [13,19,26]
red = 13
green = 19
blue = 26
cap = [22,27,17,4]
cap1 = 22
cap2 = 27
cap3 = 17
cap4 = 4

GPIO.setmode(GPIO.BCM)
GPIO.setup(slide, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(rgb, GPIO.OUT)
GPIO.setup(cap, GPIO.IN)

disp = Adafruit_SSD1306.SSD1306_128_64(rst=None)
disp.begin()
width = disp.width
height = disp.height
image = Image.new('1', (width, height)) # '1' converts image to 1-bit color
draw = ImageDraw.Draw(image)
font = ImageFont.truetype('/usr/share/fonts/truetype/freefont/FreeSans.ttf',18)

def update_display():
    disp.image(image)
    disp.display()

try:
    while True:
        for skill_selection in range(0,20):
            if GPIO.input(slide) == False:
                draw.rectangle((0,0,width,height), outline=0, fill = 0)
                draw.text((0, 0), "Difficulty?",  font=font, fill=255)
                draw.text((0, 22), "Easy",  font=font, fill=255)
                update_display()
                skill = 2
            else:
                draw.rectangle((0,0,width,height), outline=0, fill = 0)
                draw.text((0, 0), "Difficulty?",  font=font, fill=255)
                draw.text((0, 22), "Hard",  font=font, fill=255)
                update_display()
                skill = 1
            time.sleep(.1)

        target = random.randint(20,100)
```