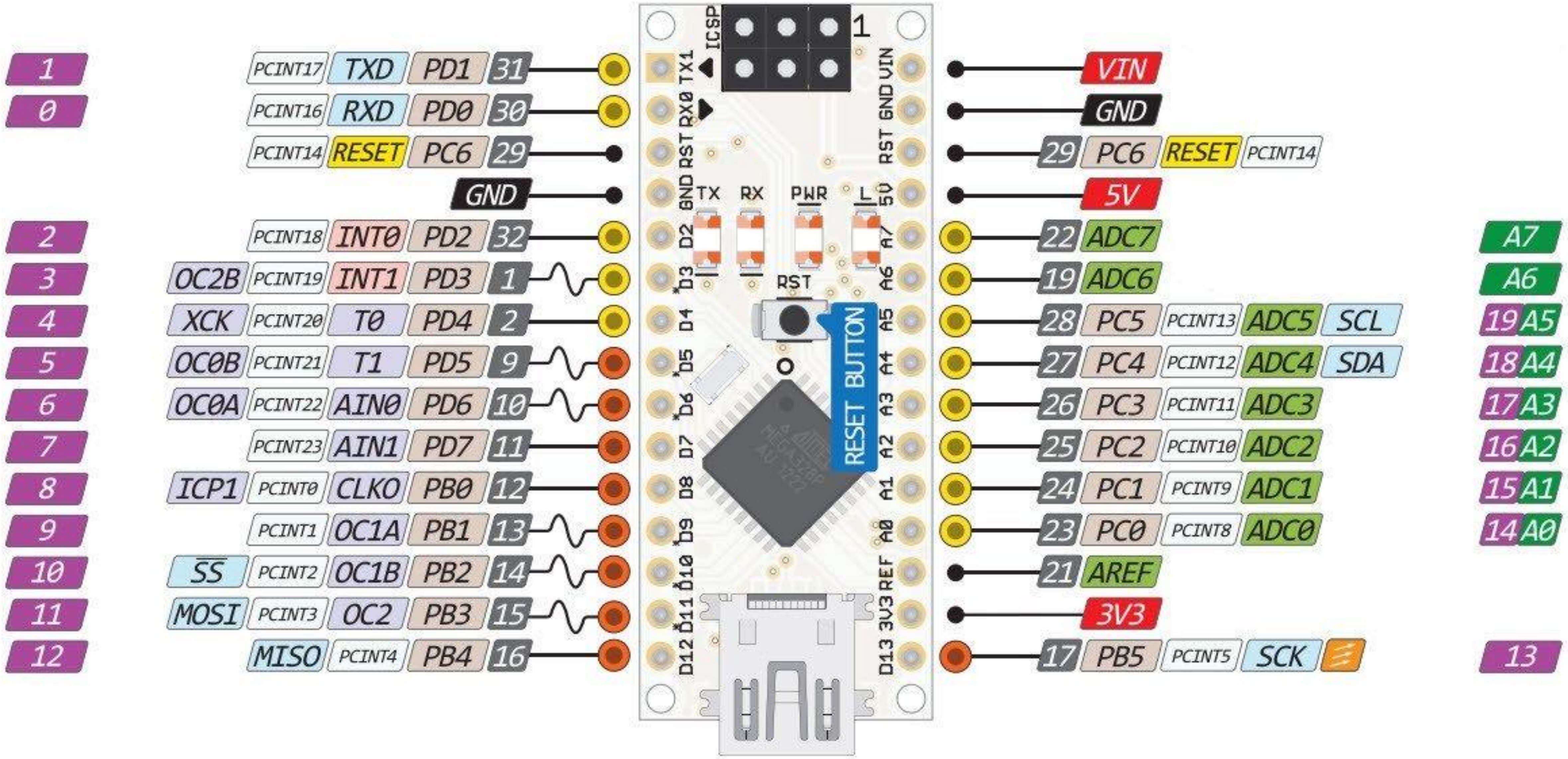
Workshop

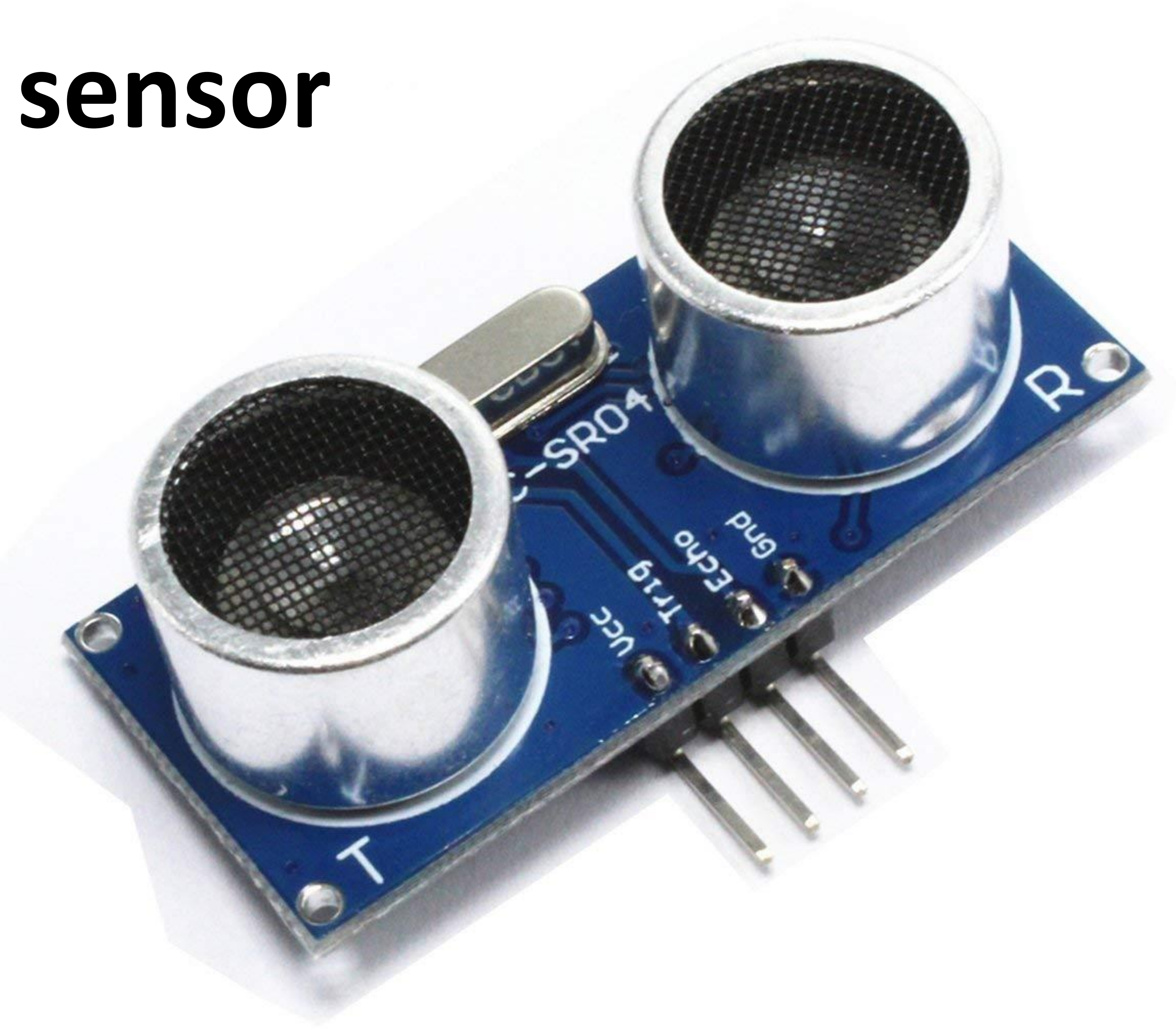
Sensors

Mr Ben Bird

ARDUINO NANO PINOUT



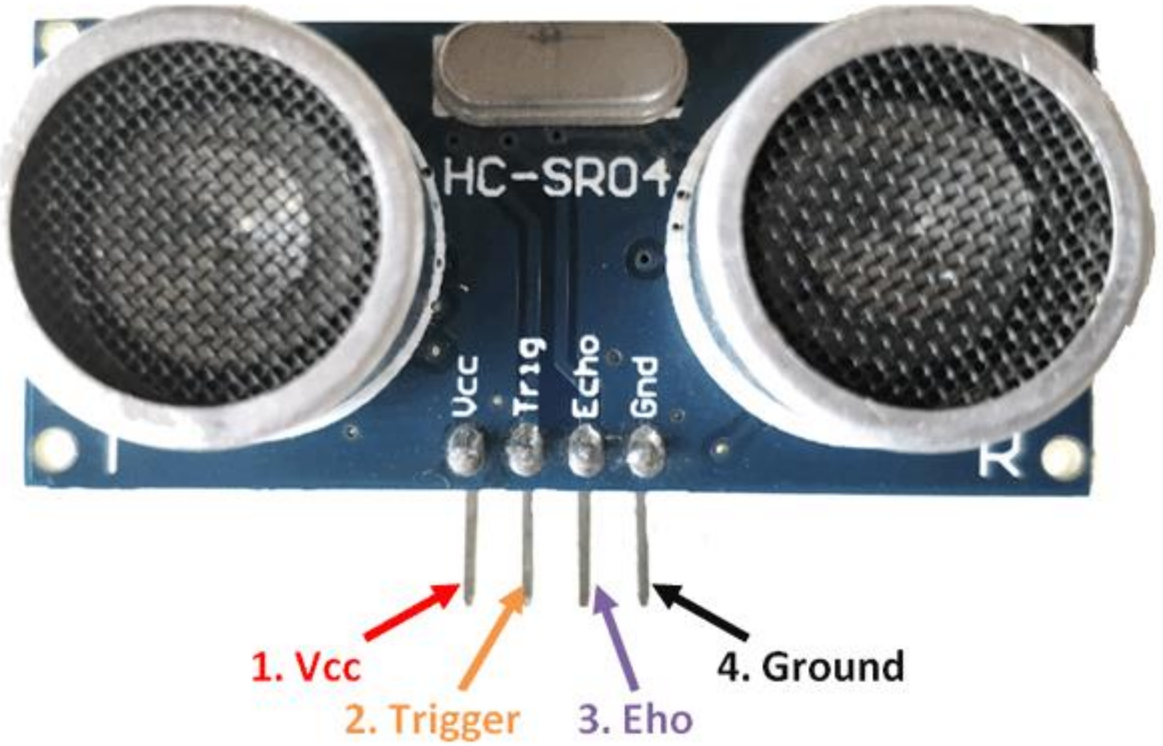
Ultrasonic sensor



https://images-na.ssl-images-amazon.com/images/I/61-2fYKuyKL._SL1200_.jpg

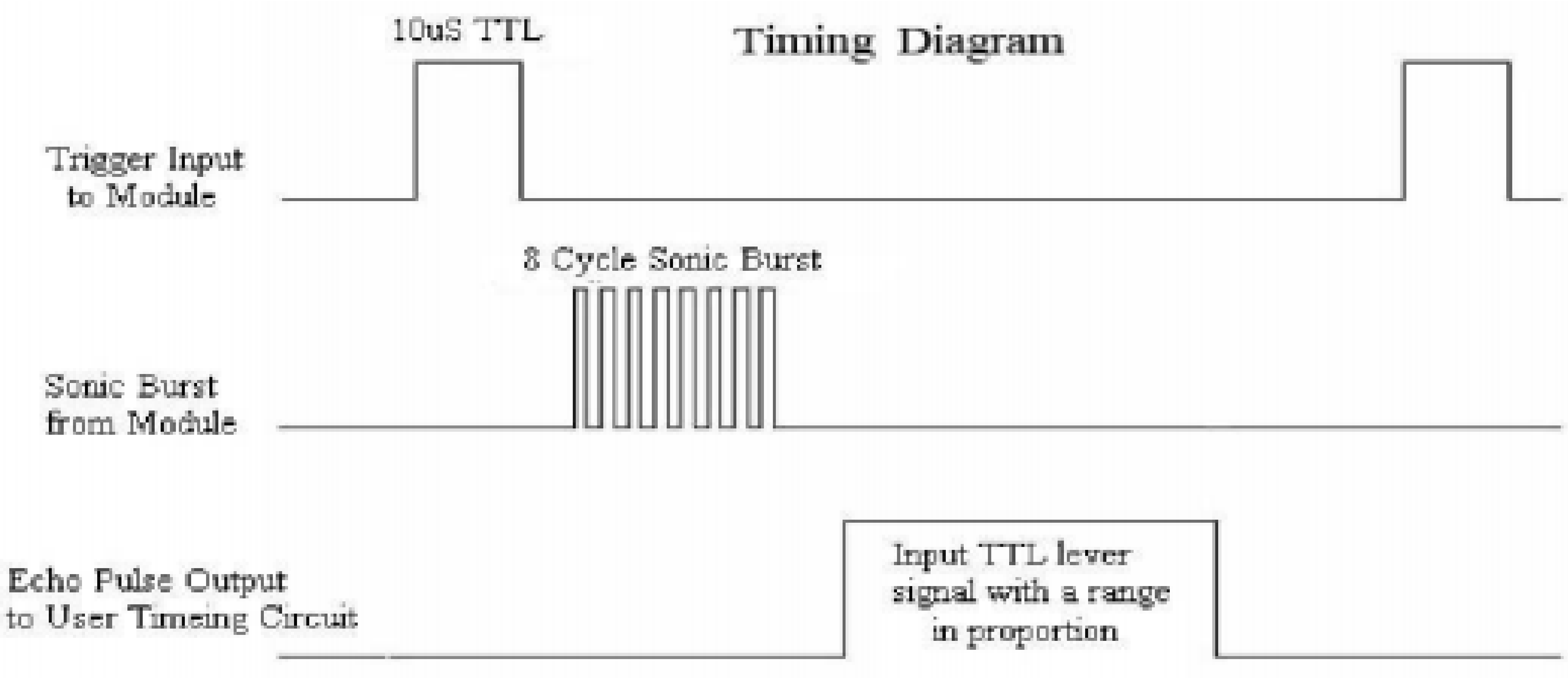
Ultrasonic Sensor features

- Uses ultrasound to find the distance of an object.
- Powered by 3.3V or 5V
- Can be controlled by one pin – trigger and echo can be tied together. Can also be timed using the i2c bus.

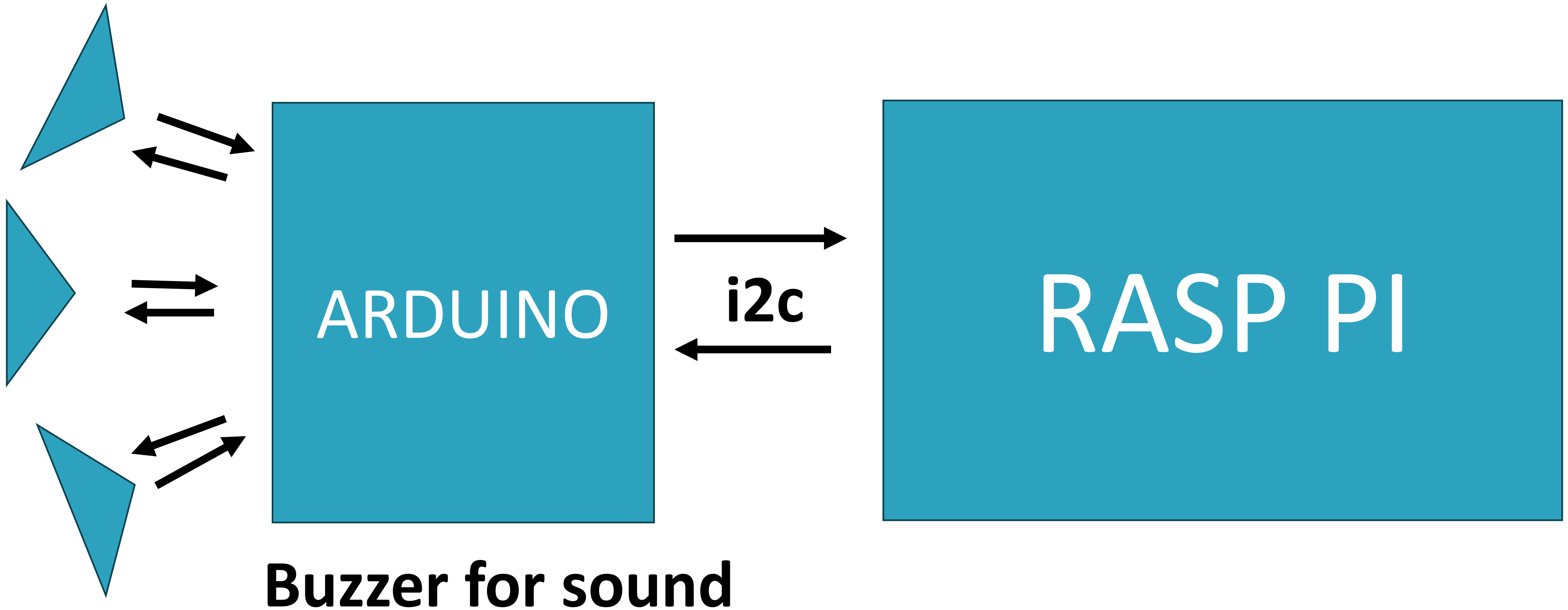


Pin Number	Pin Name	Description
1	Vcc	The Vcc pin powers the sensor, typically with +5V but can go down to 3.
2	Trigger	Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave.
3	Echo	Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor. Pins 2 and 3 are soldered together as the trigger and echo don't happen at the same time
4	Ground	0V or ground

How to use the ultrasound module


10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timeing Circuit
Input TTL lever signal with a range in proportion

It does appear that the i2c bus on the Raspberry PI screen could be used as a virtual i2c, however… the youtube video of the tracker design suggests an issue – it is worth investigating if it works…



Buzzer for sound effect

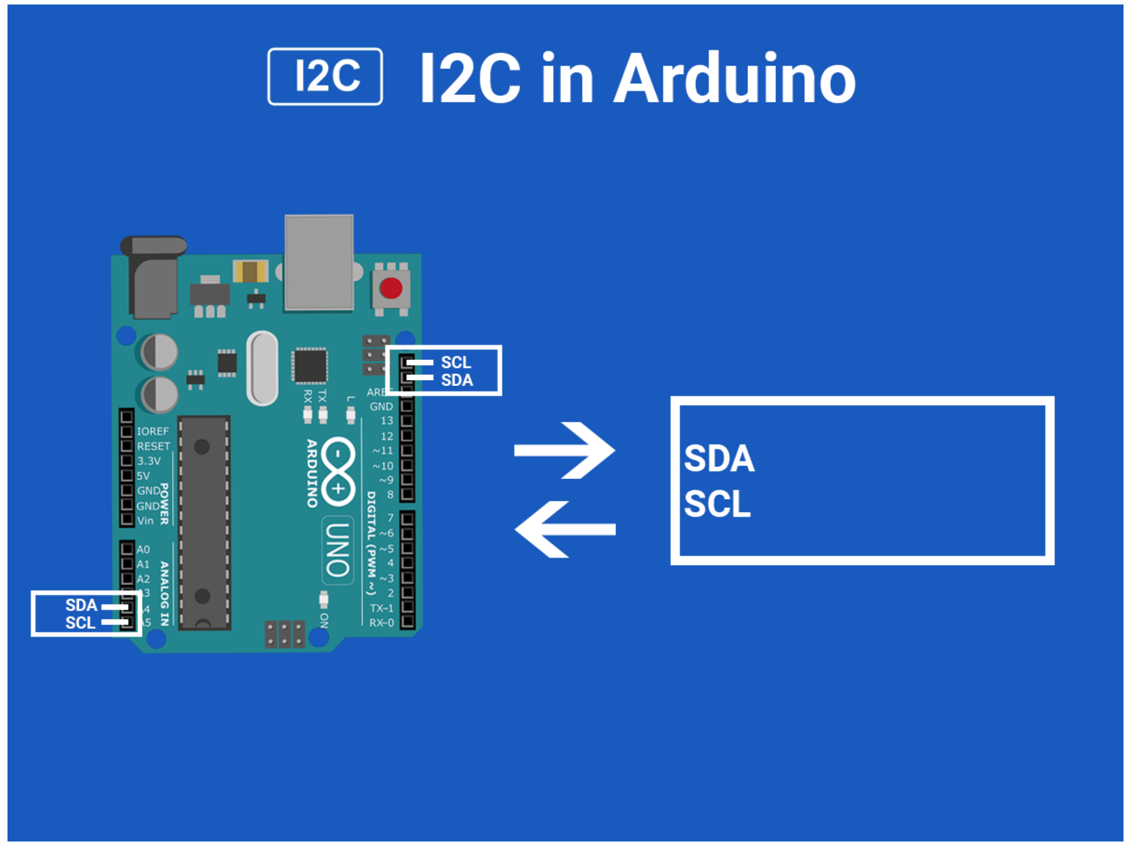
Arduino example “PING”

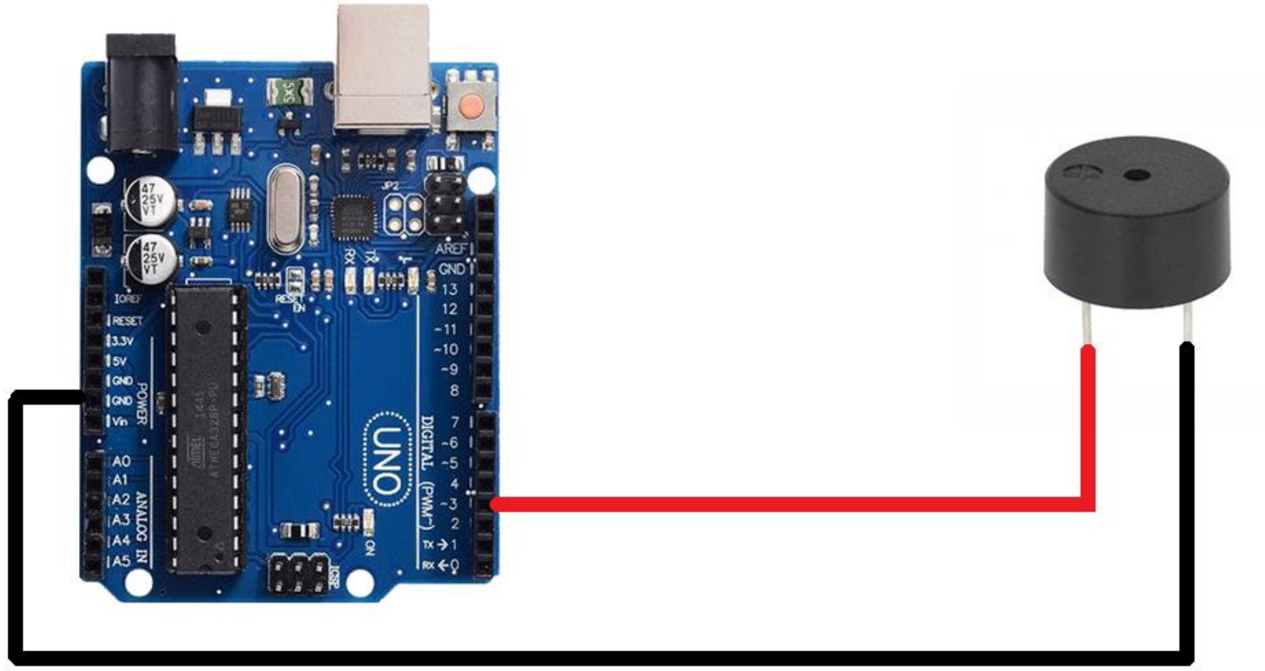
```
// establish variables for duration of the ping, and the distance result
// in inches and centimeters:
long duration, inches, cm;

// The PING))) is triggered by a HIGH pulse of 2 or more microseconds.
// Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
pinMode(pingPin, OUTPUT);
digitalWrite(pingPin, LOW);
delayMicroseconds(2);
digitalWrite(pingPin, HIGH);
delayMicroseconds(5);
digitalWrite(pingPin, LOW);
```

Arduino example “WIRE”

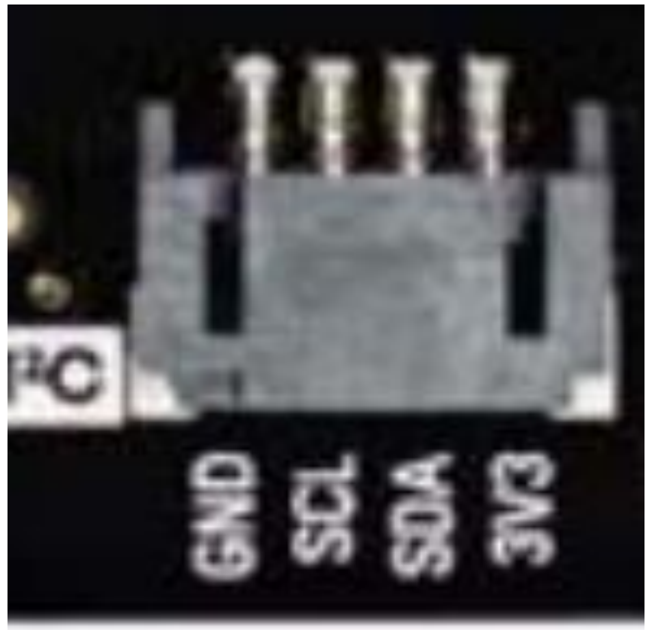
```
  void onRequest() {
  Wire.write(i++);
  Wire.print(" Packets.");
  Serial.println("onRequest");
}
```

```
Int range =1000;
while(1)
{
tone(8, 120, 50);
delay(range);
range--;
noTone(8);
}
```





HyperPixel 4.0 Screen contains an I2C breakout on the back.
This should be used to connect to the Arduino Nano.



GND-> GND on Arduino
SCL -> SCL A5 on Arduino
SDA->SDA A4 on Arduino.
Jumper cables are provided.
NOTE THAT THE I2C PORT ON THE BACK OF THE SCREEN IS A VIRTUAL PORT AND IS /DEV/IC2-11
so is the 11th i2c device on the pi. The screen requires i2c itself and manages that itself.
DO NOT. DO NOT. DO NOT. ENABLE I2C ON THE RASPBERRY PI SETTINGS.

Next steps…..

Building the tracker needs a screen and a processor to display this…..

[Raspberry Pi SPI and I2C Tutorial - SparkFun Learn]

[Pygame Front Page — pygame v2.6.0 documentation]

https://youtu.be/qKiGF54wvsQ?feature=shared

```
/*
  Arduino Slave for Raspberry Pi Master
  i2c_slave_ard.ino
  Connects to Raspberry Pi via I2C

  DroneBot Workshop 2019
  https://dronebotworkshop.com
*/

// Include the Wire library for I2C
#include <Wire.h>

// LED on pin 13
const int ledPin = 13;

void setup() {
  // Join I2C bus as slave with address 8
  Wire.begin(0x8);
  // Call receiveEvent when data received
  Wire.onReceive(receiveEvent);

  // Setup pin 13 as output and turn LED off
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);
}

// Function that executes whenever data is received from master
void receiveEvent(int howMany) {
  while (Wire.available()) { // loop through all but the last
    char c = Wire.read(); // receive byte as a character
    digitalWrite(ledPin, c);
  }
}
void loop() {
  delay(100);
}
```

```
#  Raspberry Pi Master for Arduino Slave
#  i2c_master_pi.py
#  Connects to Arduino via I2C

#  DroneBot Workshop 2019
#  https://dronebotworkshop.com

from smbus import SMBus

addr = 0x8 # bus address
bus = SMBus(11) # indicates /dev/ic2-11 (TFT IS 11)

numb = 1

print ("Enter 1 for ON or 0 for OFF")
while numb == 1:

ledstate = input(">>>>   ")

if ledstate == "1":
bus.write_byte(addr, 0x1) # switch it on
elif ledstate == "0":
bus.write_byte(addr, 0x0) # switch it on
else:
numb = 0
```

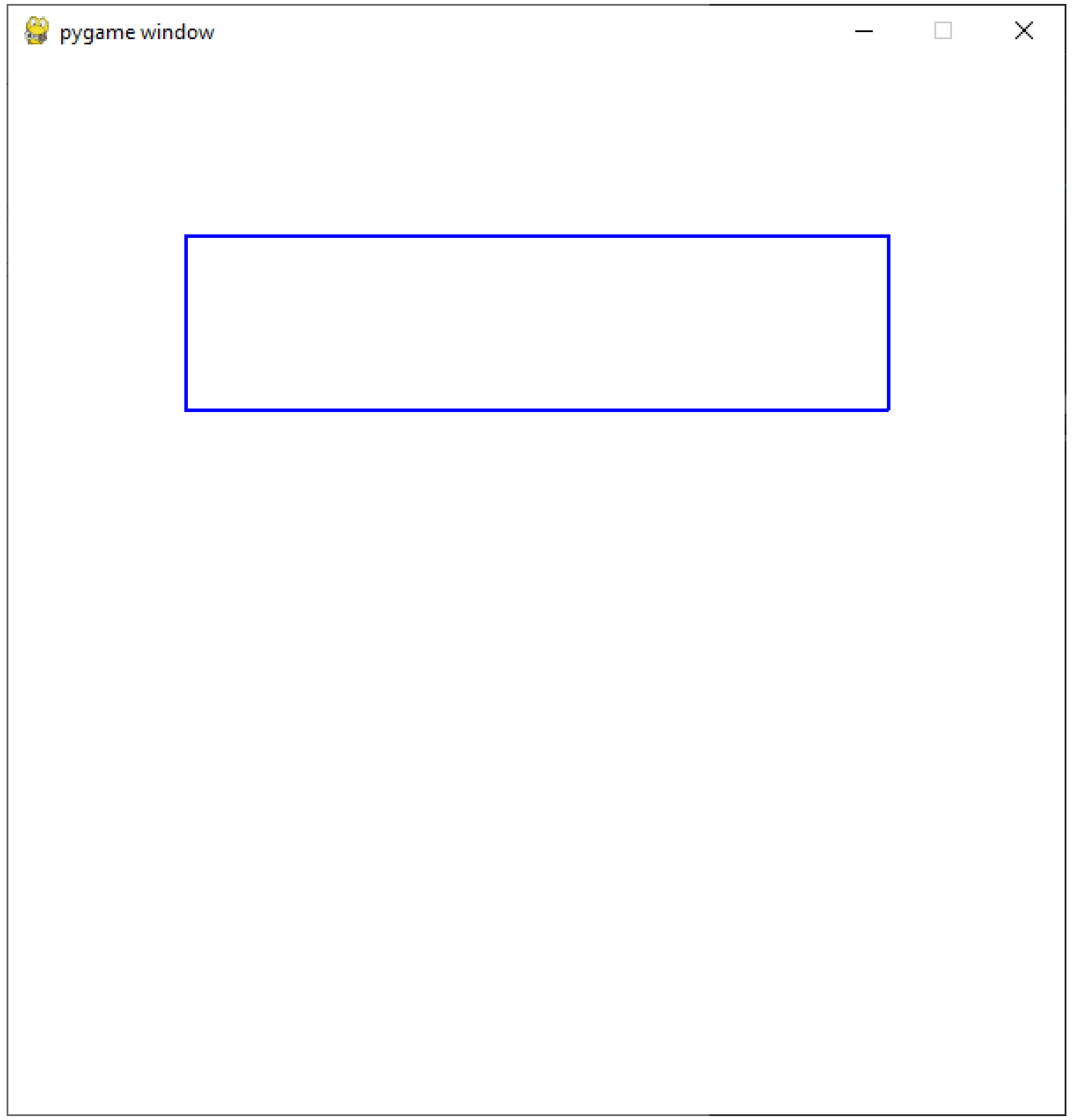
If you need to read a byte from the Arduino…

bus.read_byte(addr)

However the Arduino will not see this as a “Receive” it will see it as a “Request”. Any bus read commands must then have in the Arduino

```
Wire.onRequest(onRequest);

Void onRequest()
{
Code to run when pi wants to
read data.
}
```

```
# Importing pygame module
import pygame
from pygame.locals import *


# initiate pygame and give permission
# to use pygame's functionality.
pygame.init()


# create the display surface object
# of specific dimension.
window = pygame.display.set_mode((600, 600))


# Fill the scree with white color
window.fill((255, 255, 255))


# Using draw.rect module of
# pygame to draw the outlined rectangle
pygame.draw.rect(window, (0, 0, 255),
                 [100, 100, 400, 100], 2)


# Draws the surface object to the screen.
pygame.display.update()
```


pygame window