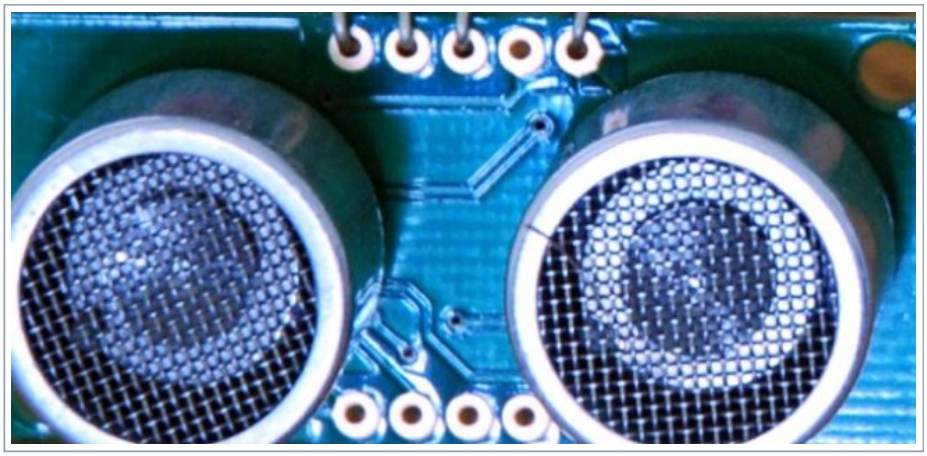
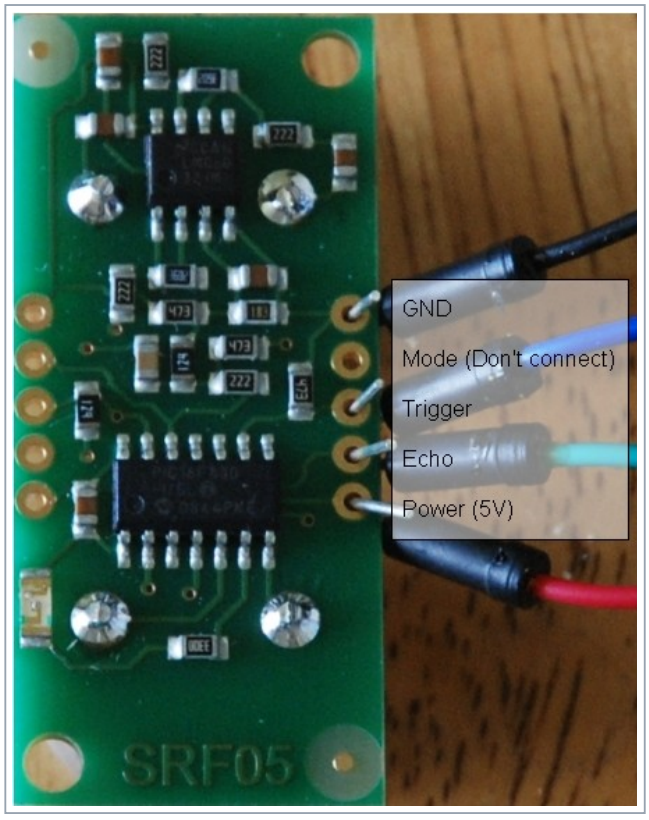


A guide to using the ***SRF05 Distance Sensor*** with ***Arduino*** in order to calculate distances from objects. In this case I'm also altering the output of an LED with PWM according to how close an object is to the sensor. So the nearer you are the brighter the LED.

So if we start with the SRF05, it's an IC that works by sending an ultrasound pulse at around 40Khz. It then waits and listens for the pulse to echo back, calculating the time taken in microseconds (1 microsecond = 1.0×10^{-6} seconds). You can trigger a pulse as fast as 20 times a second and it can determine objects up to 3 metres away and as near as 3cm. It needs a 5V power supply to run.

Adding the SRF05 to the Arduino is very easy, only 4 pins to worry about. Power, Ground, Trigger and Echo. Since it needs 5V and Arduino provides 5V I'm obviously going to use this to power it. Below is a diagram of my SRF05, showing the pins. There are 2 sets of 5 pins, 1 set you can use, the other is for programming the PIC chip so don't touch them!

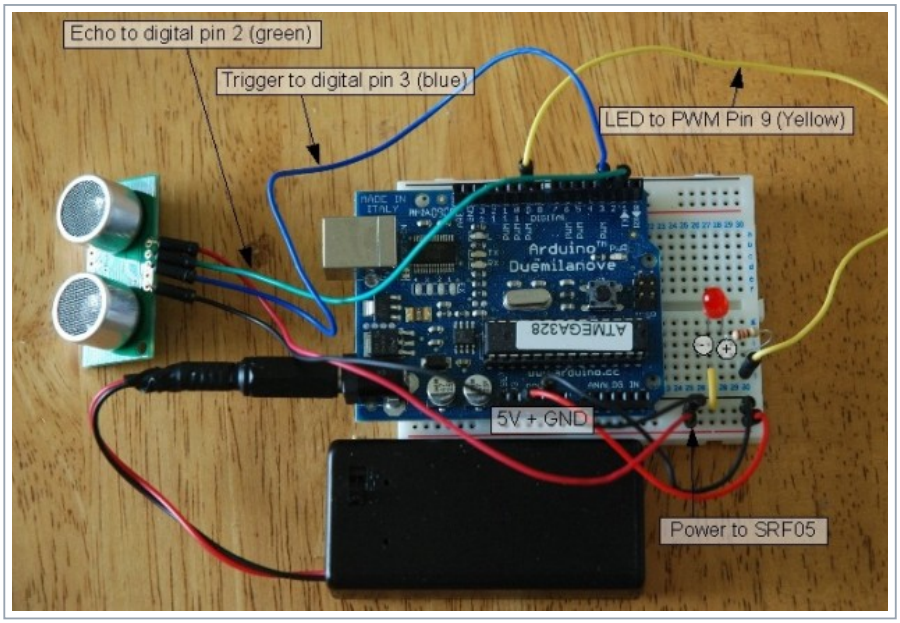


SRF05 Arduino Components

220 Ohm resistor (Red, Red, Brown, Gold)
SRF05 Ultrasonic range finder
LED
Arduino Deumilanove w/ ATMEGA328
Breadboard / Prototyping board
Jumper/ Connector wires
Optional 9V DC power supply or use the USB power for the Arduino

Arduino SRF05 Circuit

Very, very simple circuit, I've used the breadboard to share the GND connection and to add the LED which I could probably have done with out the breadboard. You'll see the most complex thing is the code later on.



SRF05 Arduino Distance Sensor sketch

All the work is done here, I've added code that averages the distance readings to remove some of the jitter in the results as the SRF05 is calculating distances very rapidly and there can be a lot of fluctuation. Also I convert the time in microseconds to distance by dividing the time by 58.

Why 58? Well because if you take the time in microseconds for a pulse to be sent and received e.g. for 1 meter it takes about 5764 microseconds – at least from my wall anyway. If I divide this time by the distance in cm in I will get 57.64 so I just round this up – you can calculate distance in any other unit with this method.

Here I've also decided that for every cm under 255 my LED will get 1 step brighter. I've been lazy here for the sake of the sensors 3 metre range I didn't see the point in making this any more complicated. Otherwise I would calculate the brightness on the percentile of proximity out of total range.

```
// written at: luckylarry.co.uk
```

```
// variables to take x number of readings and then average them
// to remove the jitter/noise from the SRF05 sonar readings

const int numOfReadings = 10;                  // number of readings to take/ i
int readings[numOfReadings];                   // stores the distance readings
int arrayIndex = 0;                            // arrayIndex of the current ite
int total = 0;                                 // stores the cumlative total
int averageDistance = 0;                       // stores the average value

// setup pins and variables for SRF05 sonar device

int echoPin = 2;                               // SRF05 echo pin (digital 2)
int initPin = 3;                               // SRF05 trigger pin (digital 3)
unsigned long pulseTime = 0;                   // stores the pulse in Micro Sec
unsigned long distance = 0;                    // variable for storing the dist

// setup pins/values for LED

int redLEDPin = 9;                             // Red LED, connected to digital
int redLEDValue = 0;                           // stores the value of brightnes

//setup

void setup() {

  pinMode(redLEDPin, OUTPUT);                  // sets pin 9 as output
  pinMode(initPin, OUTPUT);                    // set init pin 3 as output
  pinMode(echoPin, INPUT);                     // set echo pin 2 as input

  // create array loop to iterate over every item in the array

  for (int thisReading = 0; thisReading < numOfReadings; thisReading++) {
readings[thisReading] = 0;
 }
// initialize the serial port, lets you view the
 // distances being pinged if connected to computer
     Serial.begin(9600);
 }

// execute
void loop() {
digitalWrite(initPin, HIGH);                   // send 10 microsecond pulse
delayMicroseconds(10);                  // wait 10 microseconds before turning o
digitalWrite(initPin, LOW);                    // stop sending the pulse
pulseTime = pulseIn(echoPin, HIGH);            // Look for a return pulse, it s
distance = pulseTime/58;                       // Distance = pulse time / 58 to
 total= total - readings[arrayIndex];          // subtract the last distance
readings[arrayIndex] = distance;               // add distance reading to array
total= total + readings[arrayIndex];           // add the reading to the total
arrayIndex = arrayIndex + 1;                   // go to the next item in the ar
// At the end of the array (10 items) then start again
if (arrayIndex >= numOfReadings)  {
    arrayIndex = 0;
  }

  averageDistance = total / numOfReadings;     // calculate the average distanc

  // if the distance is less than 255cm then change the brightness of the LED

  if (averageDistance < 255) {
```

```
    redLEDValue = 255 - averageDistance;        // this means the smaller the di
  }

  analogWrite(redLEDPin, redLEDValue);          // Write current value to LED pi
  Serial.println(averageDistance, DEC);         // print out the average distanc
  delay(100);                                   // wait 100 milli seconds before

}
```

Well this is going to make the sensor for a robot methinks. I'll alter this to control a servo so