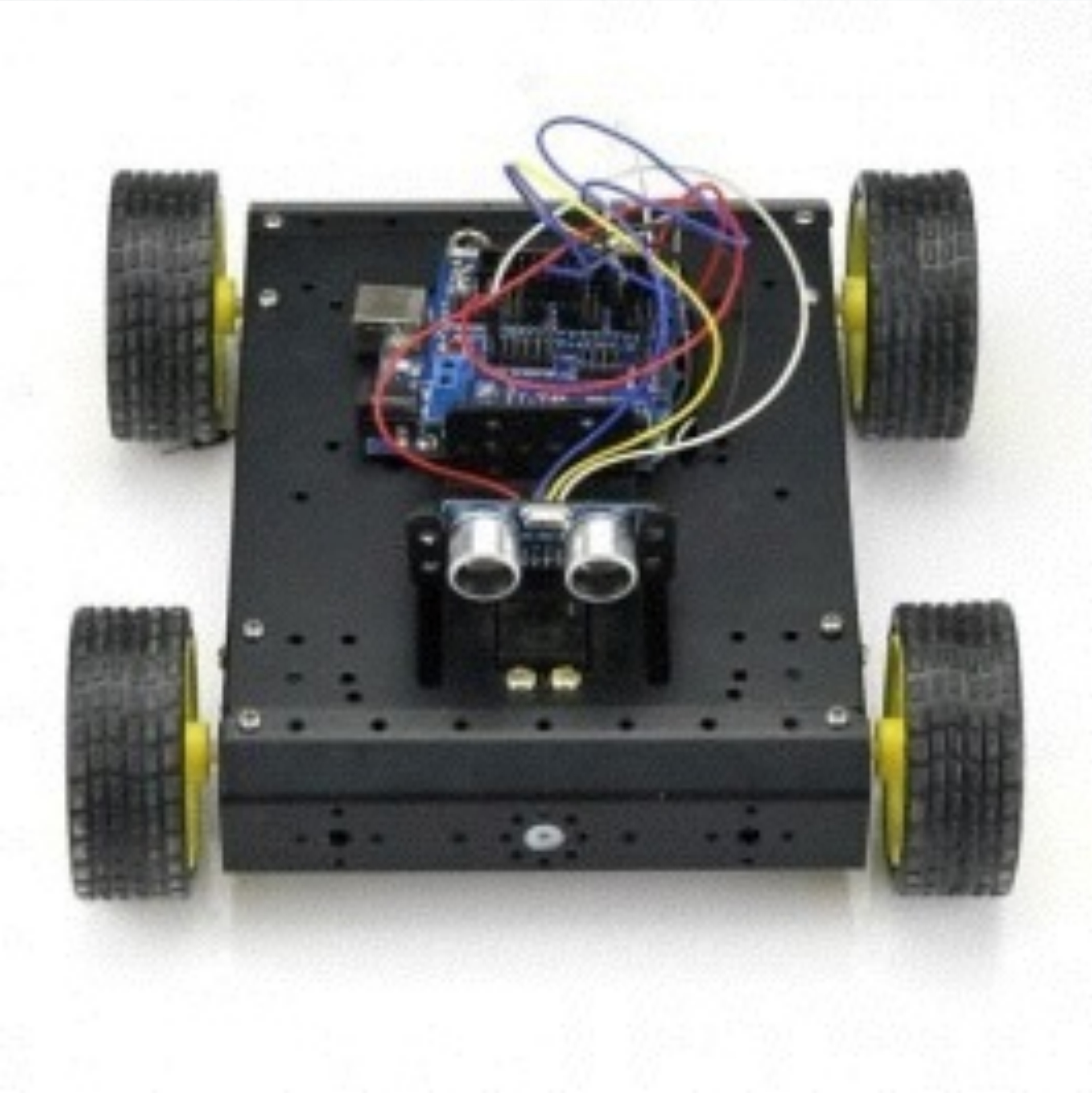
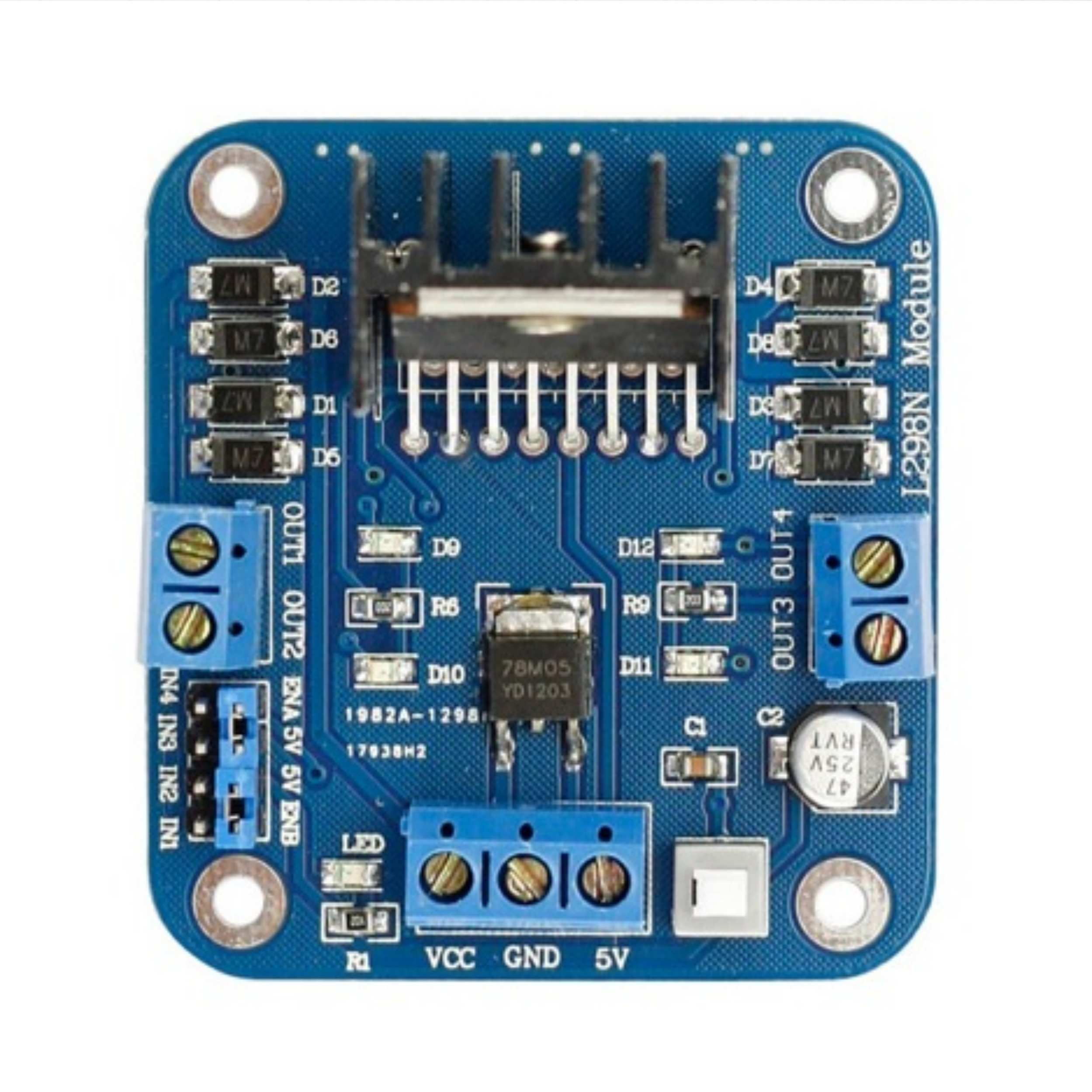




DC Robotics Arduino Motors Workshop

Instructor: Glenn Mossy - gmossy@gmail.com

December 2014

https://github.com/gmossy

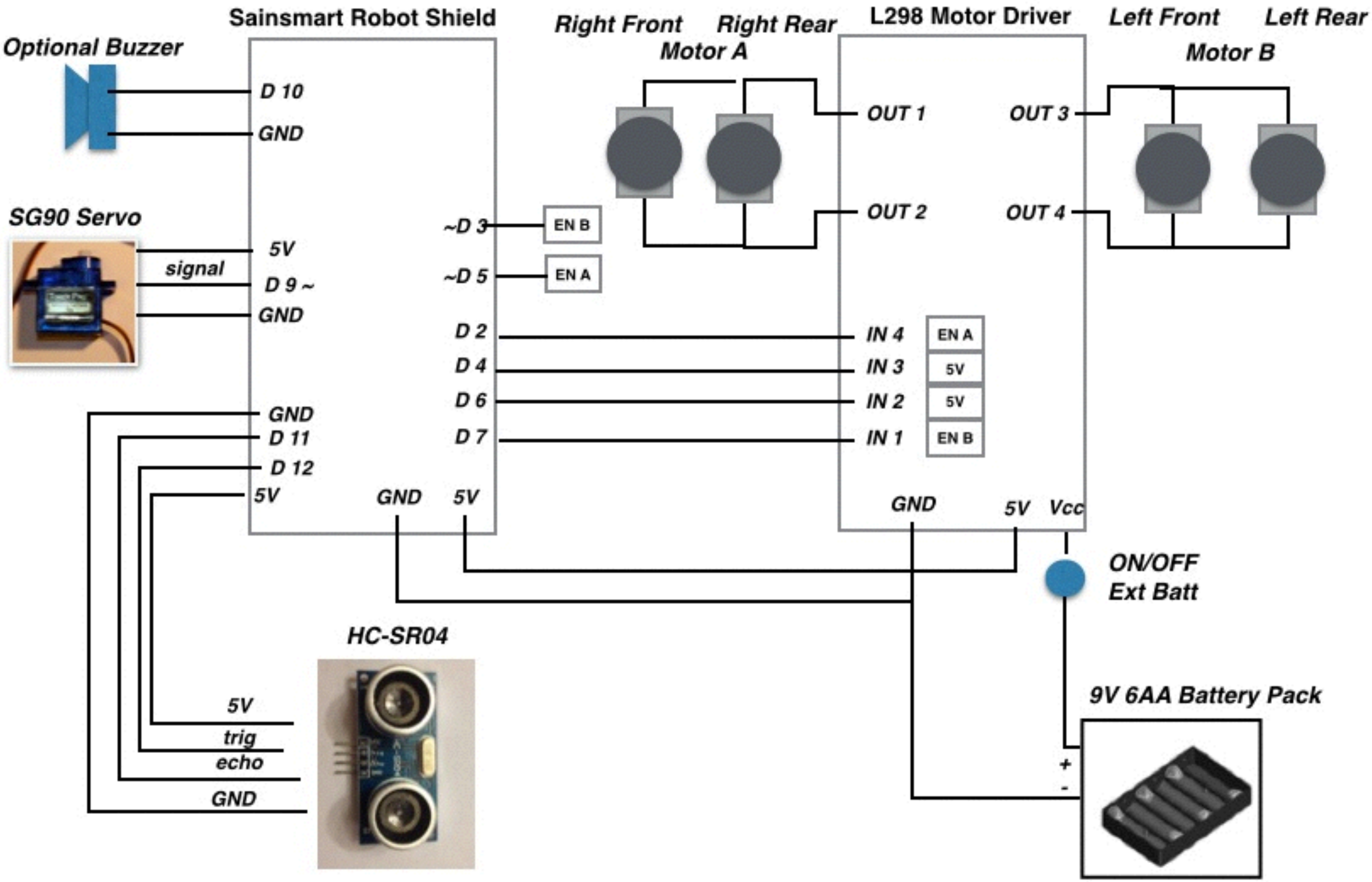
http://www.dcroboticsgroup.com

DC Robotics Group Education Director

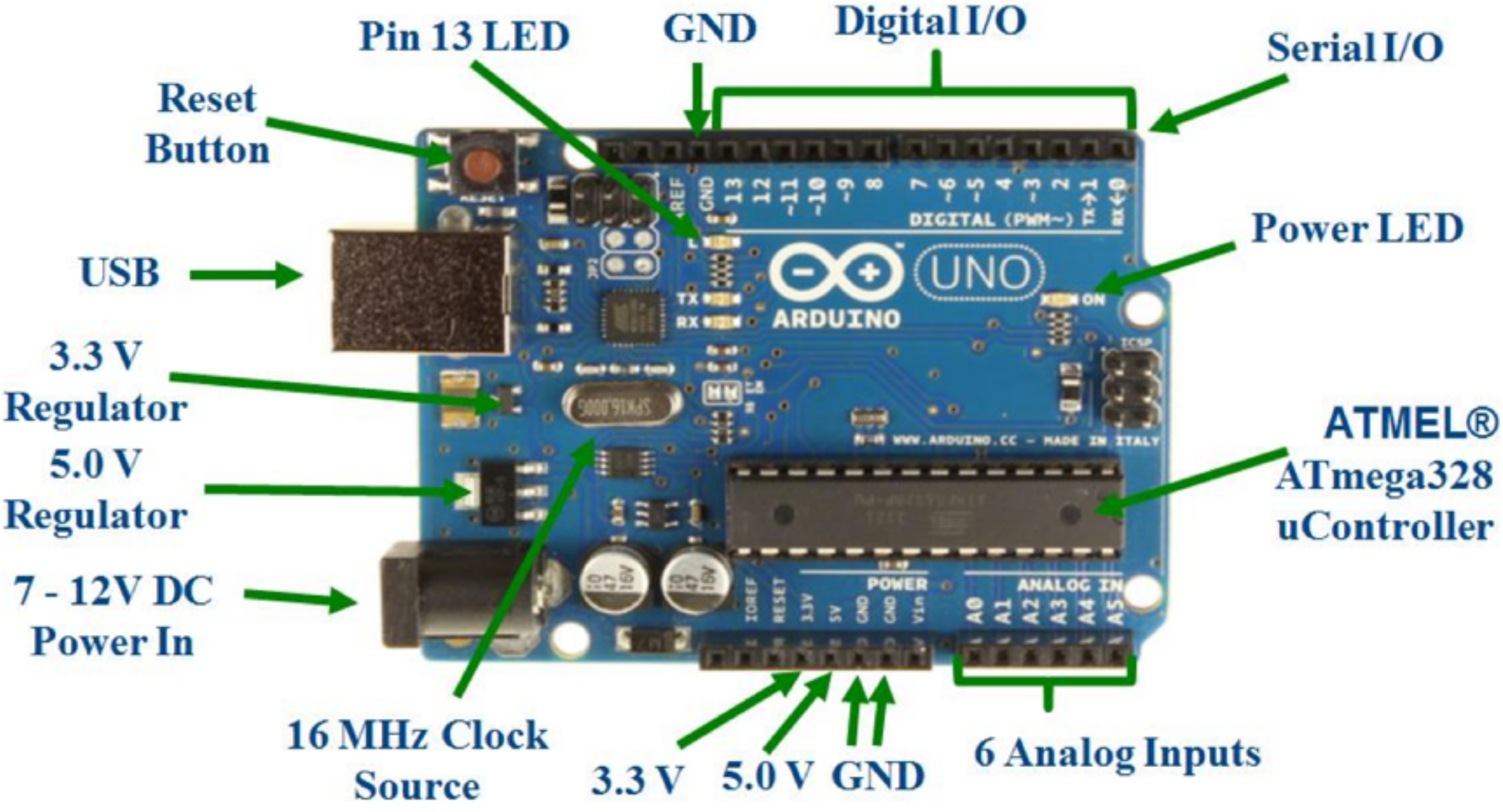
Goals of the Workshop

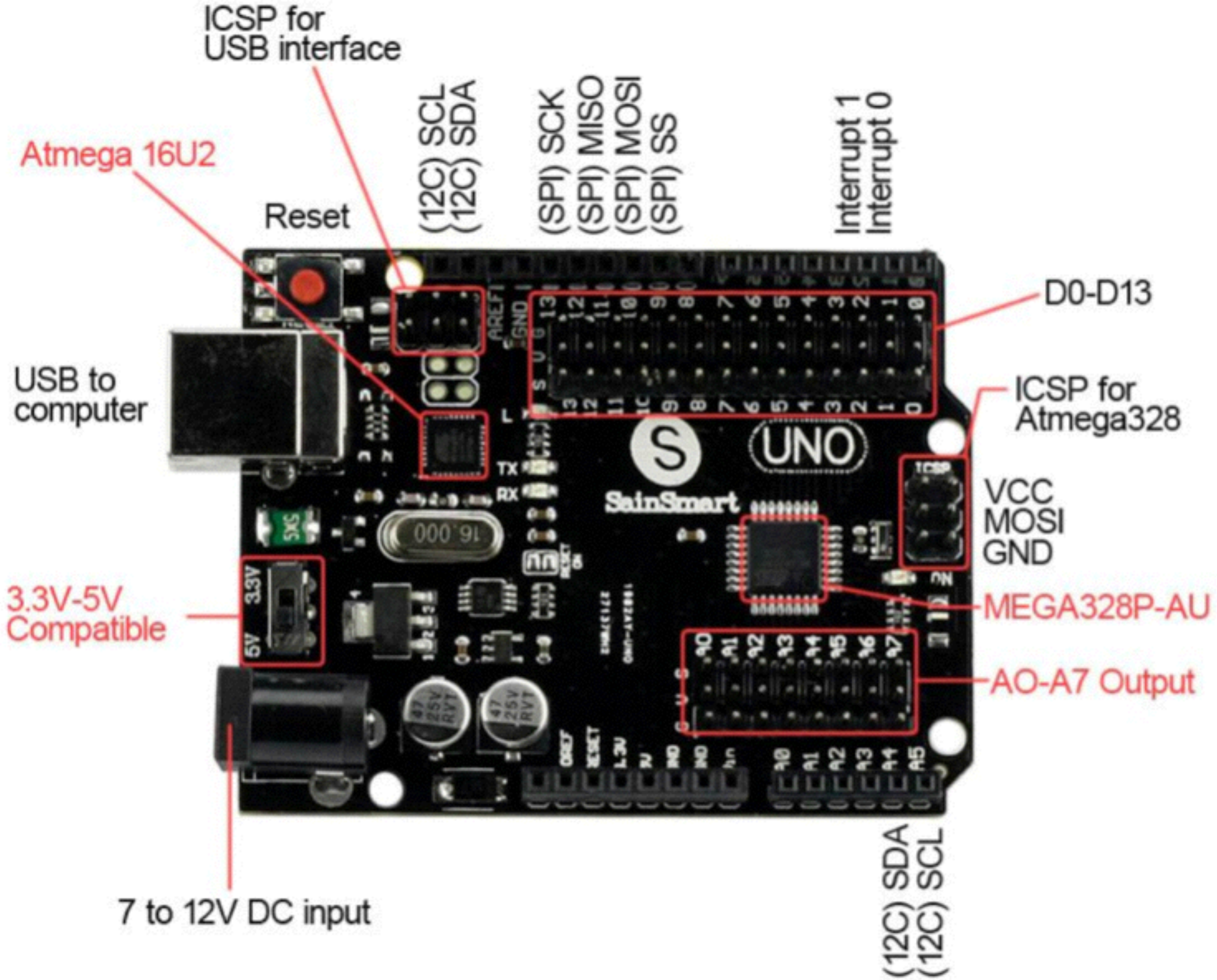
- Arduino Review and Basic Robot Control
- Servo Control
- What are H-Bridges?
- DC Motor Control
- Stepper Motor Control
- Building and Programming the robot

4WD Robot Block Diagram



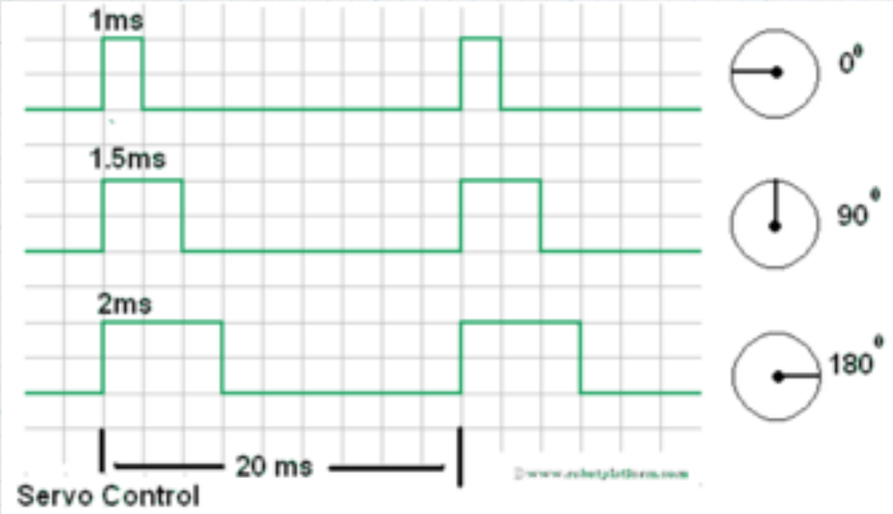
Arduino Uno, R3



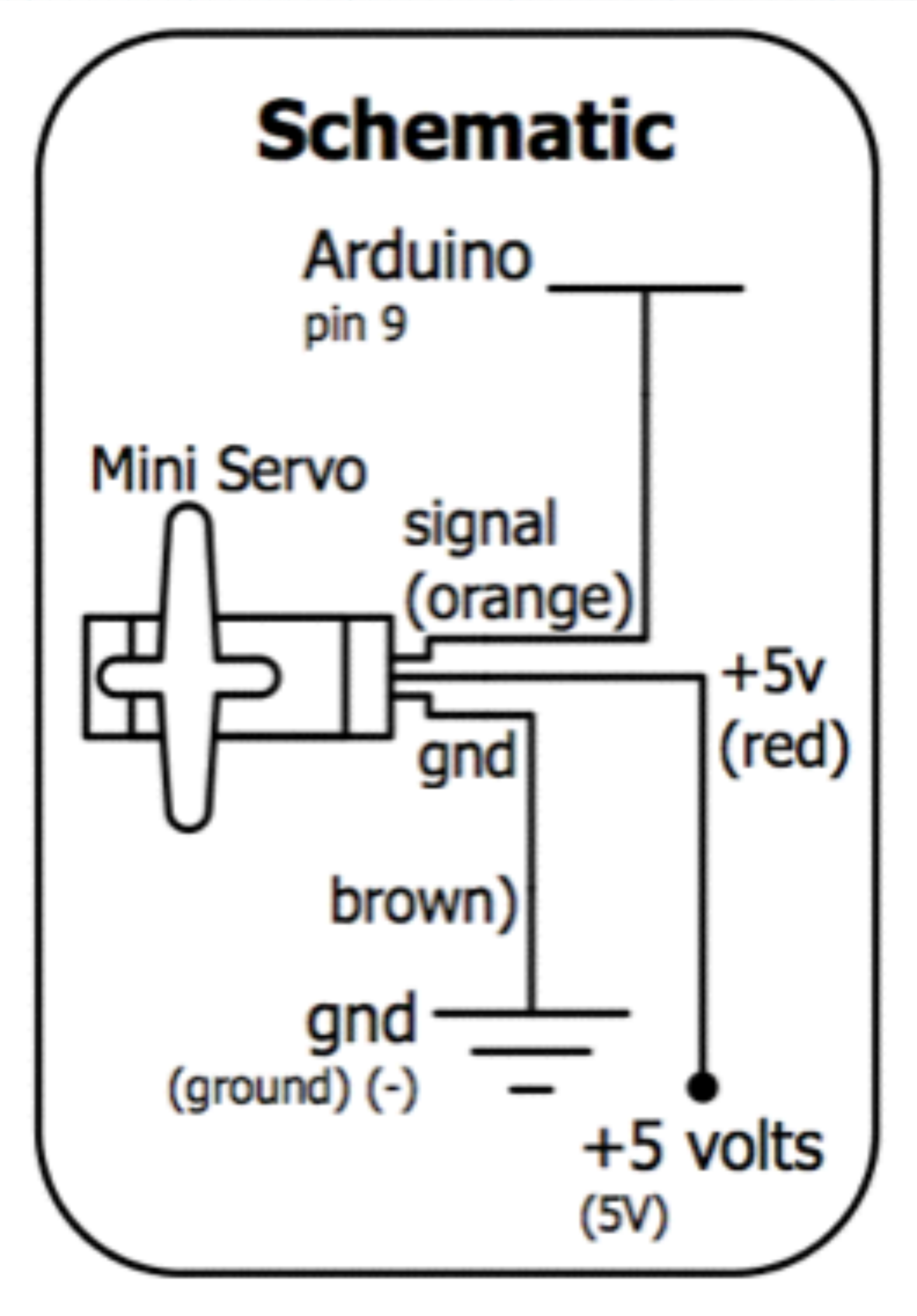

ICSP for USB interface
Atmega 16U2
Reset
(12C) SCL
(12C) SDA
(SPI) SCK
(SPI) MISO
(SPI) MOSI
(SPI) SS
Interrupt 1
Interrupt 0
D0-D13
USB to computer
ICSP for Atmega328
VCC
MOSI
GND
MEGA328P-AU
3.3V-5V Compatible
AO-A7 Output
7 to 12V DC input
(12C) SDA
(12C) SCL
UNO
SainSmart

Servo

A "servo", short for servomotor, is a motor that includes feedback circuitry that allows it to be commanded to move to specific positions. This one is very small, but larger servos are used extensively in robotics to control mechanical arms, hands, etc. You could use it to make a (tiny) robot arm aircraft control surface, or anywhere something needs to be moved to specific positions.



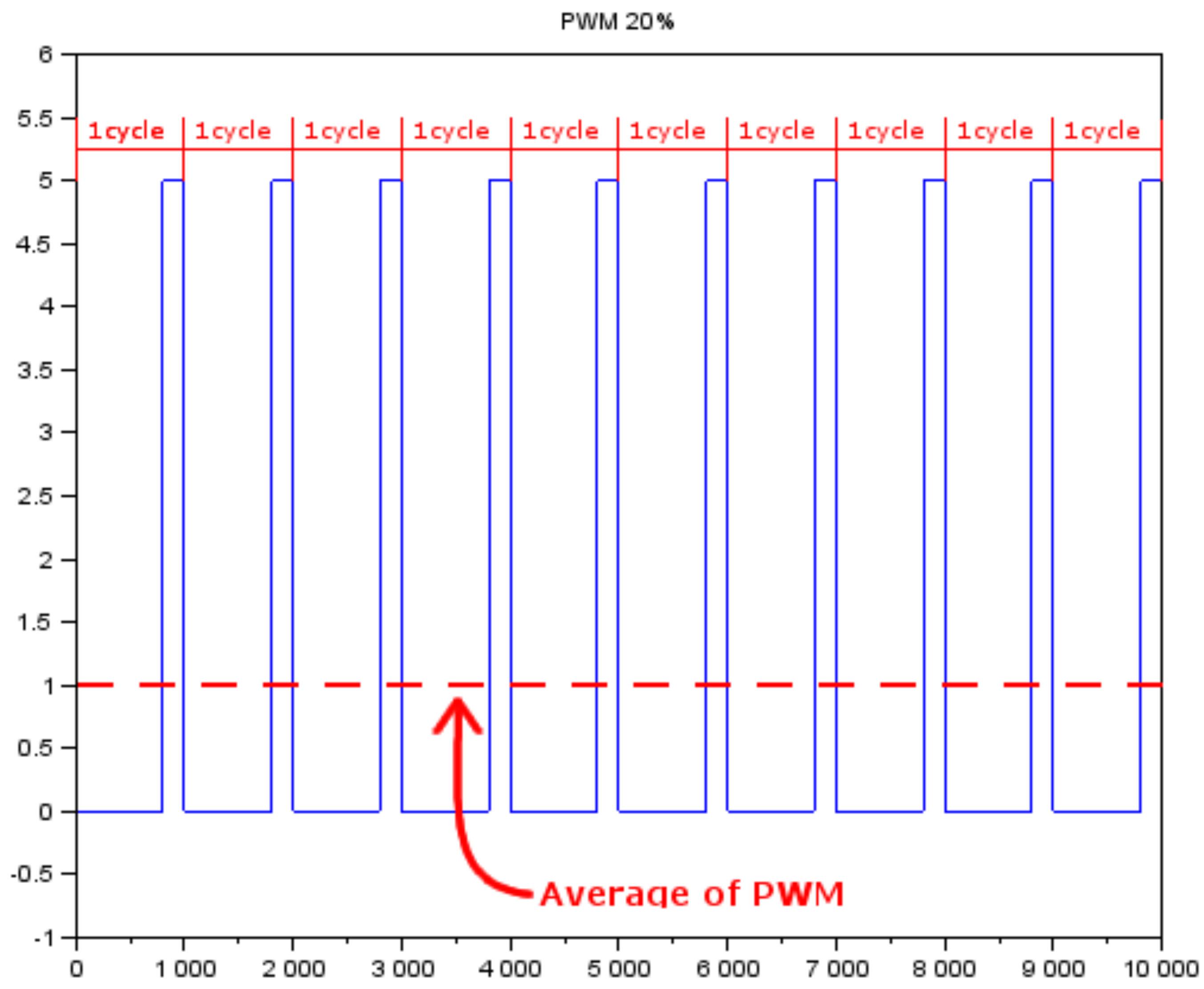
Servos

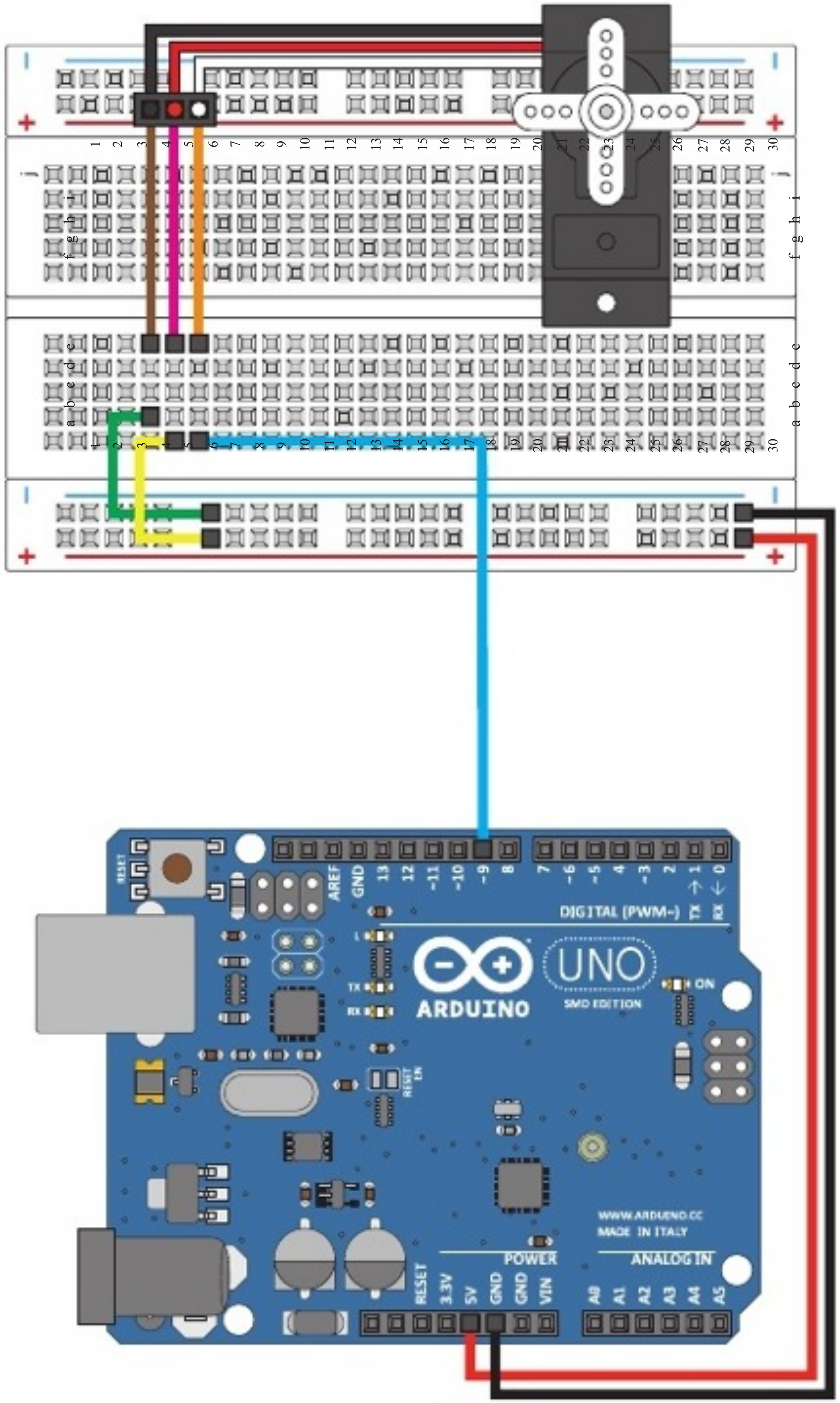


- #include <Servo.h>;
- Servo.attach(9);
- Servo.write(180);

Use servo library

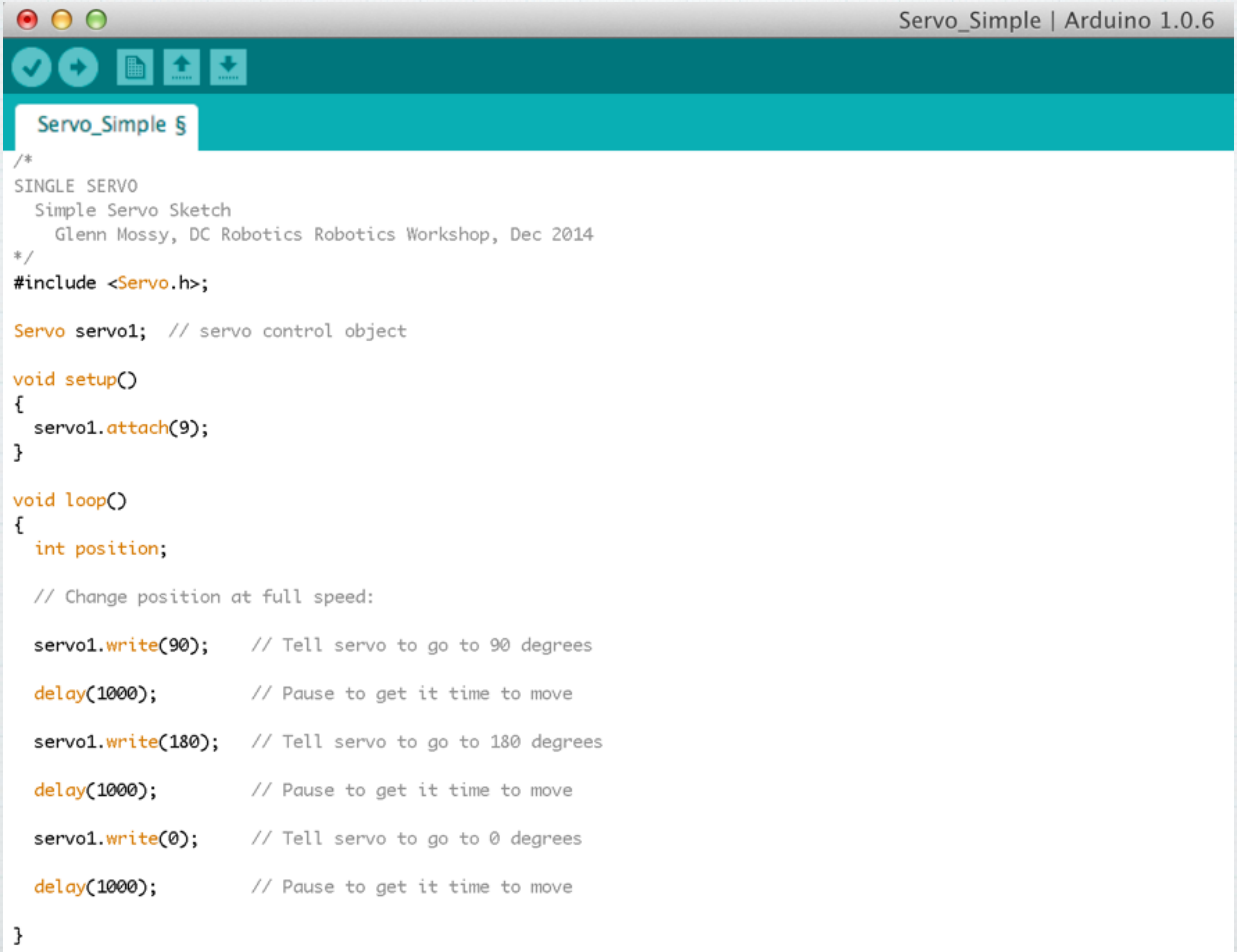
PWM


PWM 20%
1cycle
1cycle
1cycle
1cycle
1cycle
1cycle
1cycle
1cycle
1cycle
1cycle
Average of PWM
6
5.5
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
1 000
2 000
3 000
4 000
5 000
6 000
7 000
8 000
9 000
10 000



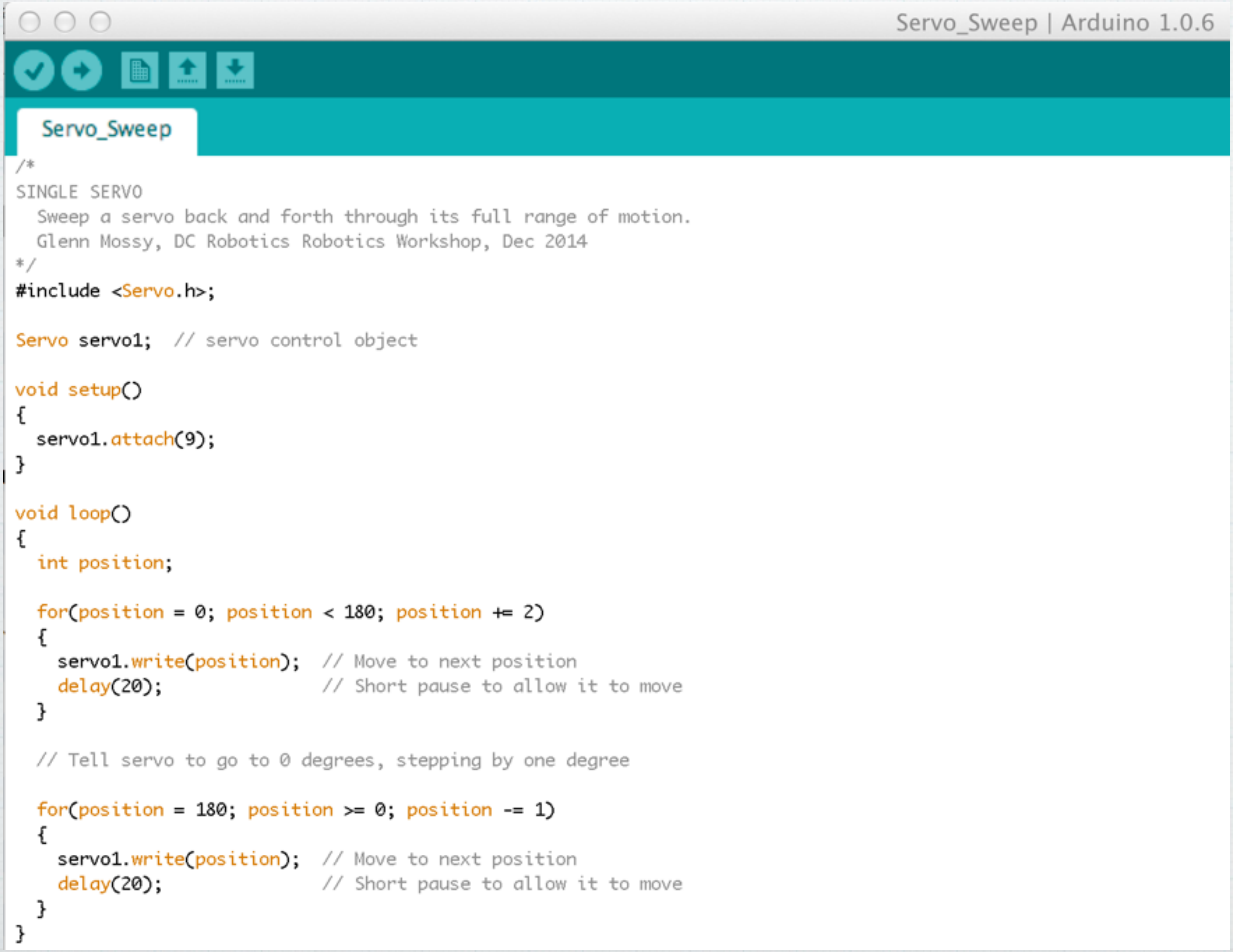
Circuit 8: A Single Servo

Servo Code Simple Example



```
/*
SINGLE SERVO
  Simple Servo Sketch
    Glenn Mossy, DC Robotics Robotics Workshop, Dec 2014
*/
#include <Servo.h>;

Servo servo1;  // servo control object

void setup()
{
  servo1.attach(9);
}

void loop()
{
  int position;

  // Change position at full speed:

  servo1.write(90);     // Tell servo to go to 90 degrees

  delay(1000);          // Pause to get it time to move

  servo1.write(180);    // Tell servo to go to 180 degrees

  delay(1000);          // Pause to get it time to move

  servo1.write(0);      // Tell servo to go to 0 degrees

  delay(1000);          // Pause to get it time to move

}
```

Servo Code Sweep Example

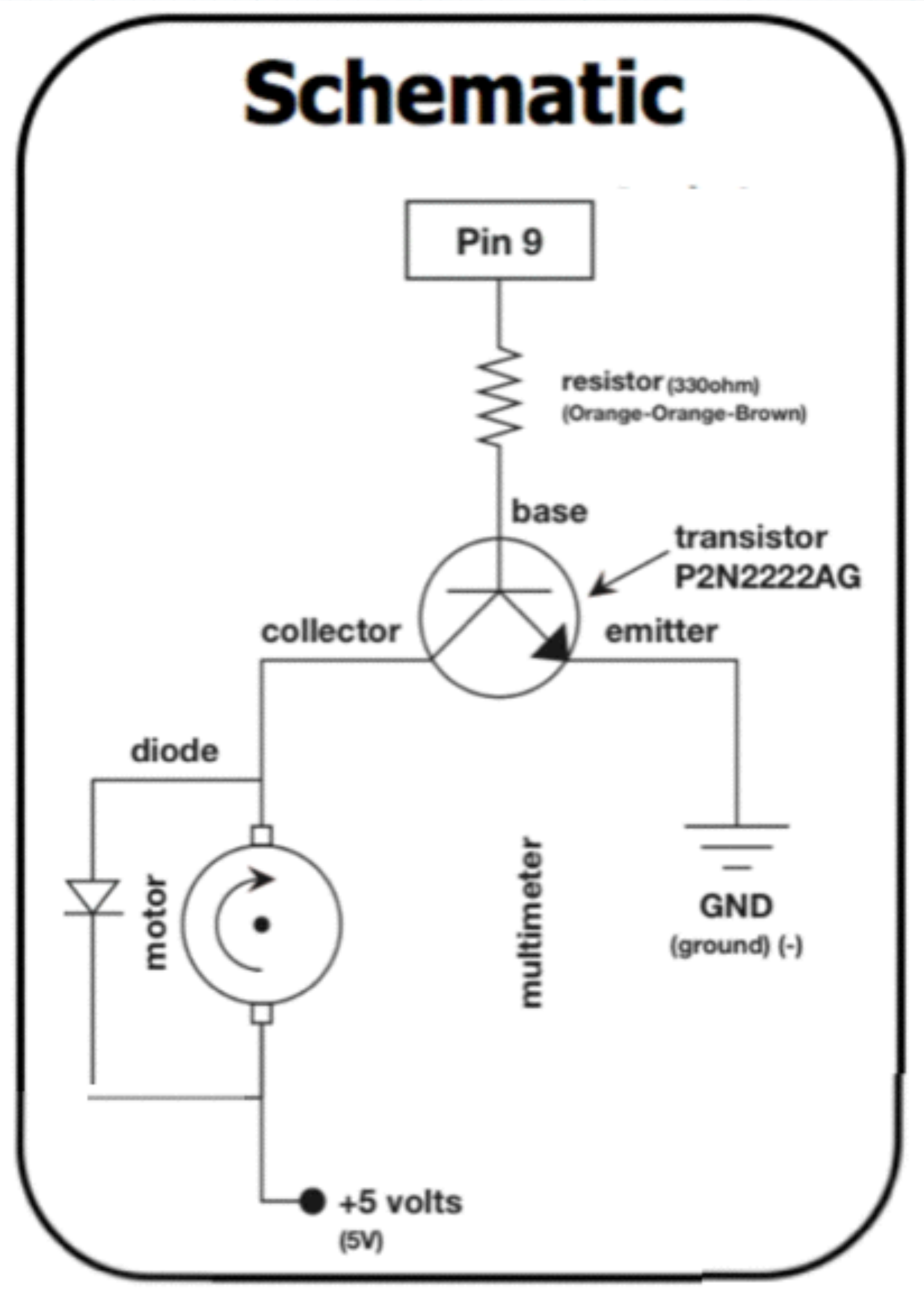


```
/*
SINGLE SERVO
  Sweep a servo back and forth through its full range of motion.
  Glenn Mossy, DC Robotics Robotics Workshop, Dec 2014
*/
#include <Servo.h>;

Servo servo1;  // servo control object

void setup()
{
  servo1.attach(9);
}

void loop()
{
  int position;

  for(position = 0; position < 180; position += 2)
  {
    servo1.write(position);  // Move to next position
    delay(20);               // Short pause to allow it to move
  }

  // Tell servo to go to 0 degrees, stepping by one degree

  for(position = 180; position >= 0; position -= 1)
  {
    servo1.write(position);  // Move to next position
    delay(20);               // Short pause to allow it to move
  }
}
```

Very Basic DC Motor Control

- int (motorPin, OUTPUT);
- digitalWrite(motorPin, HIGH);
- digitalWrite(motorPin, LOW);

Use control

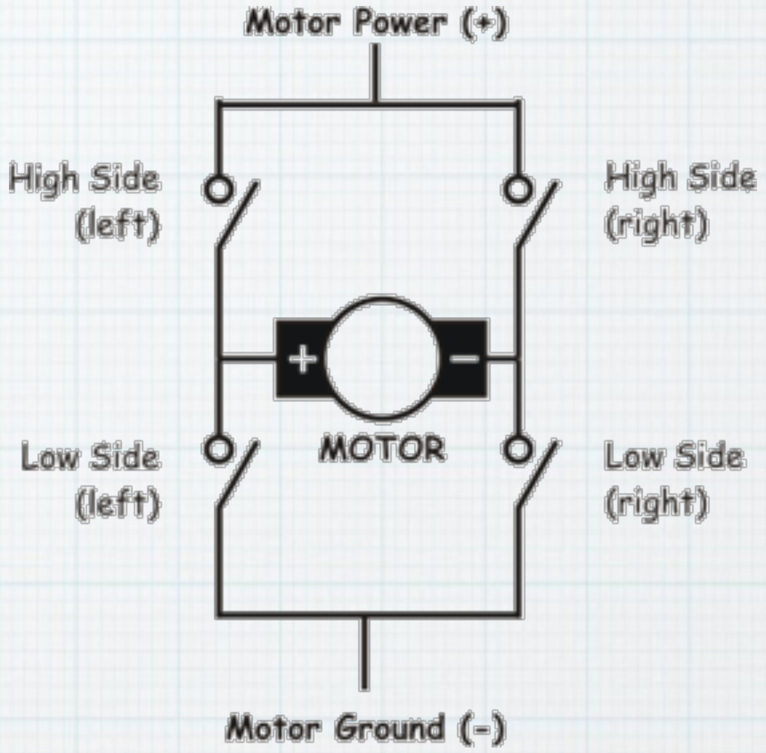


What is an H-Bridge

- How do H-Bridges work
- http://www.tigoe.net/pcomp/code/circuits/motors/stepper-motors/

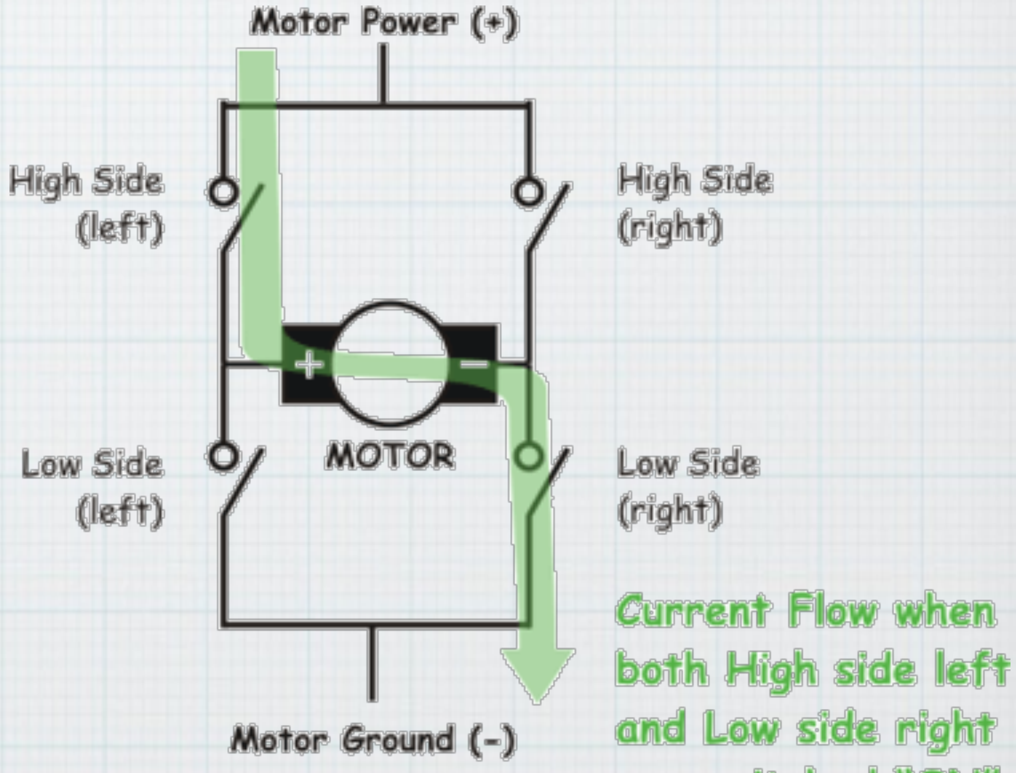
Basic H-Bridge Theory

Sometimes called a "full bridge" the H-bridge is so named because it has four switching elements at the "corners" of the H and the motor forms the cross bar. The basic bridge is shown in the figure to the right.
The switches are turned on in pairs, either high left and lower right, or lower left and high right, but never both switches on the same "side" of the bridge. If both switches on one side of a bridge are turned on it creates a short circuit between the battery plus and battery minus terminals.



To power the motor, you turn on two switches that are diagonally opposed. In the picture to the right, imagine that the high side left and low side right switches are turned on. The current flow is shown in green.

The current flows and the motor begins to turn in a "positive" direction. What happens if you turn on the high side right and low side left switches? Current flows the other direction through the motor and the motor turns in the opposite direction.



Current Flow when both High side left and Low side right are switched "ON"

Basic H-Bridge Theory

One more topic in the basic theory section, quadrants. If each switch can be controlled independently then you can do some interesting things with the bridge. If you built it out of a single DPDT relay, you can really only control forward or reverse. You can build a small truth table that tells you for each of the switch's states, what the bridge will do. As each switch has one of two states, and there are four switches, there are 16 possible states. However, since any state that turns both switches on one side on is "bad" there are in fact only four useful states (the four quadrants) where the transistors are turned on.

High Side	High Side	Lower Left	Lower Right	Quadrant Description
On	Off	Off	On	Motor goes Clockwise
Off	On	On	Off	Motor goes Counter-clockwise
On	On	Off	Off	Motor "brakes" and decelerates
Off	Off	On	On	Motor "brakes" and decelerates

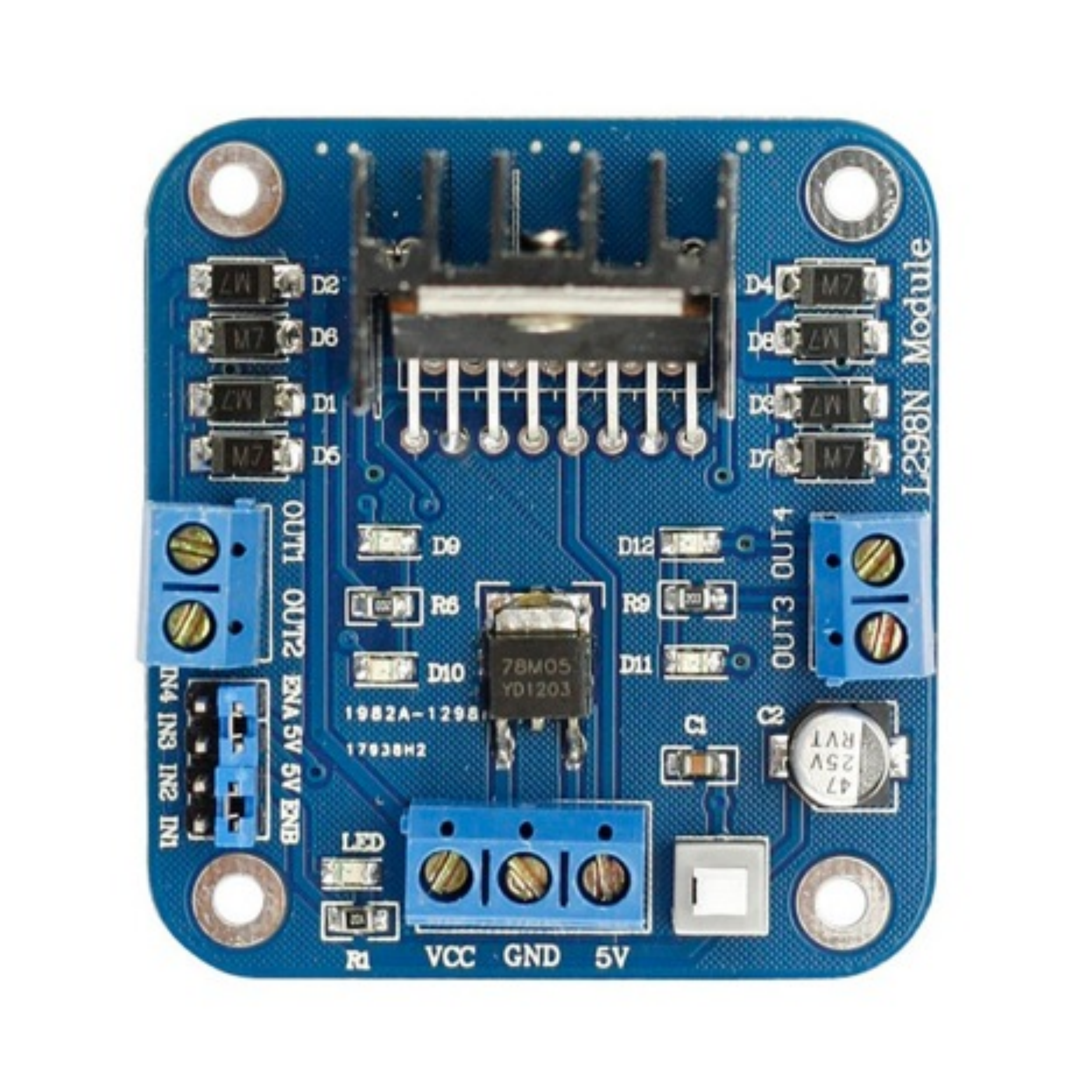
The last two rows describe a maneuver where you "short circuit" the motor which causes the motors generator effect to work against itself. The turning motor generates a voltage which tries to force the motor to turn the opposite direction. This causes the motor to rapidly stop spinning and is called "braking" on a lot of H-bridge designs.

Of course there is also the state where all the transistors are turned off. In this case the motor coasts if it was spinning and does nothing if it was doing nothing.

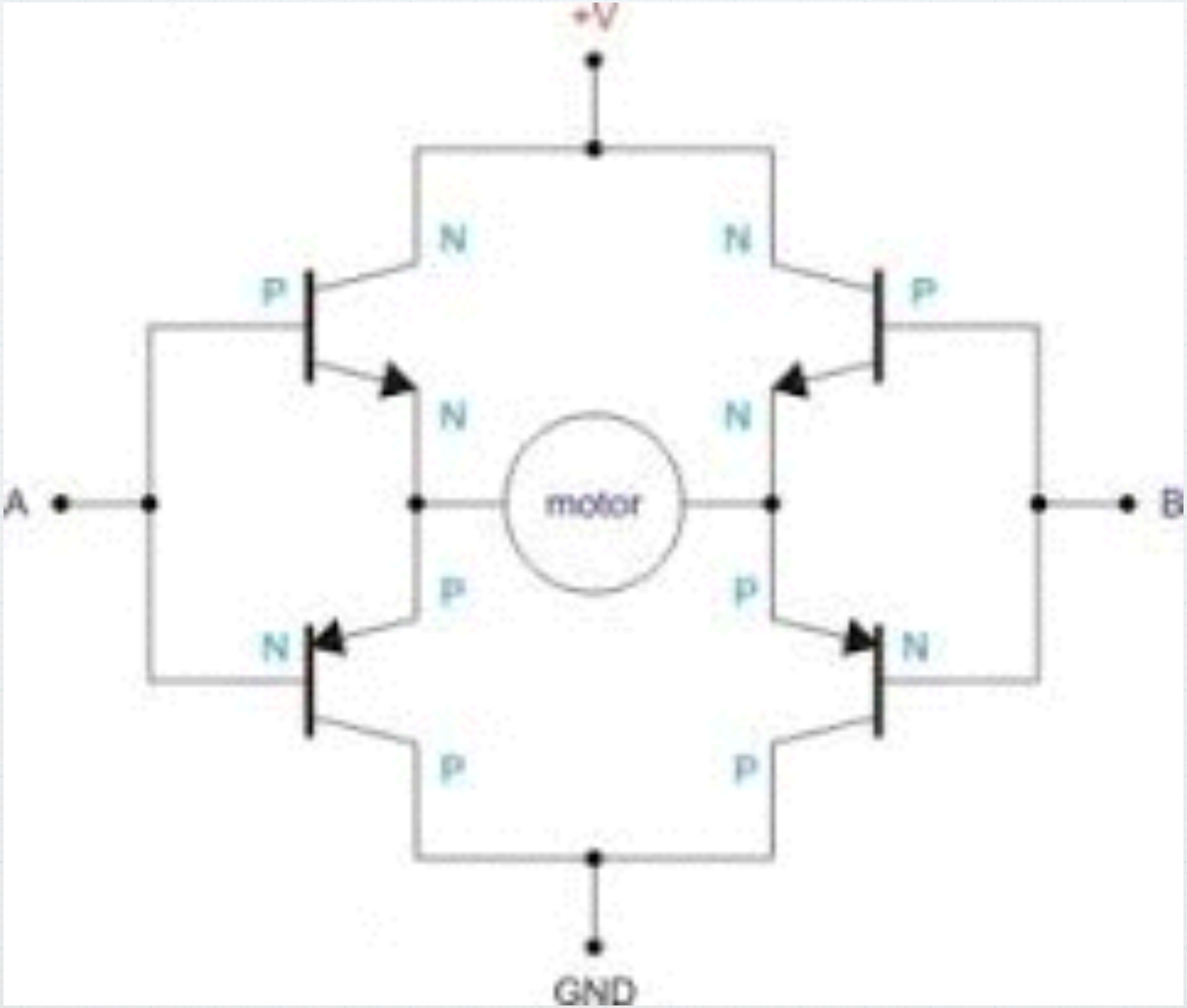
4WD Robot Motor Basic

```
//Setting the wheels to go backward by setting the backward pins to HIGH
void backward(){
  digitalWrite(rightForwardPin, LOW);
  digitalWrite(rightBackwardPin, HIGH);
  digitalWrite(leftForwardPin, LOW);
  digitalWrite(leftBackwardPin, HIGH);

  //use pwm to control motor speed through enable pin
      analogWrite(ENA, duty);

}

//Setting the wheels to go right by setting the rightBackwardPin and leftForwardPin to HIGH
void right(){
  digitalWrite(rightForwardPin, LOW);
  digitalWrite(rightBackwardPin, HIGH);
  digitalWrite(leftForwardPin, HIGH);
  digitalWrite(leftBackwardPin, LOW);

      //use pwm to control motor speed through enable pin
      analogWrite(ENB, duty);

}
```

Implementing H-Bridges

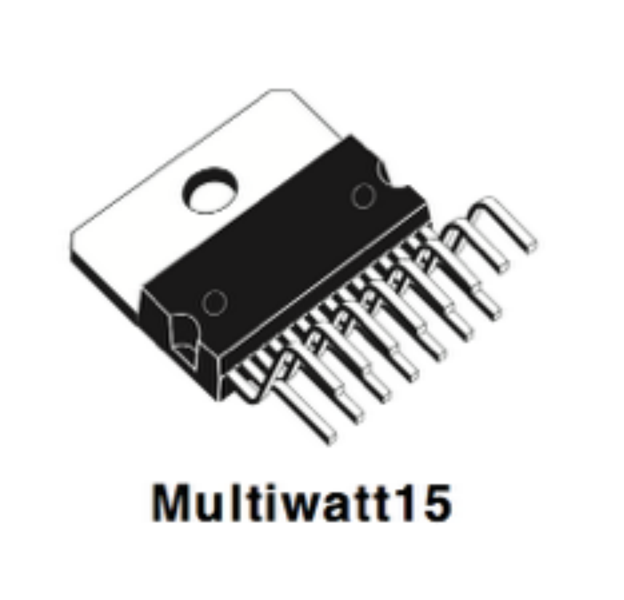


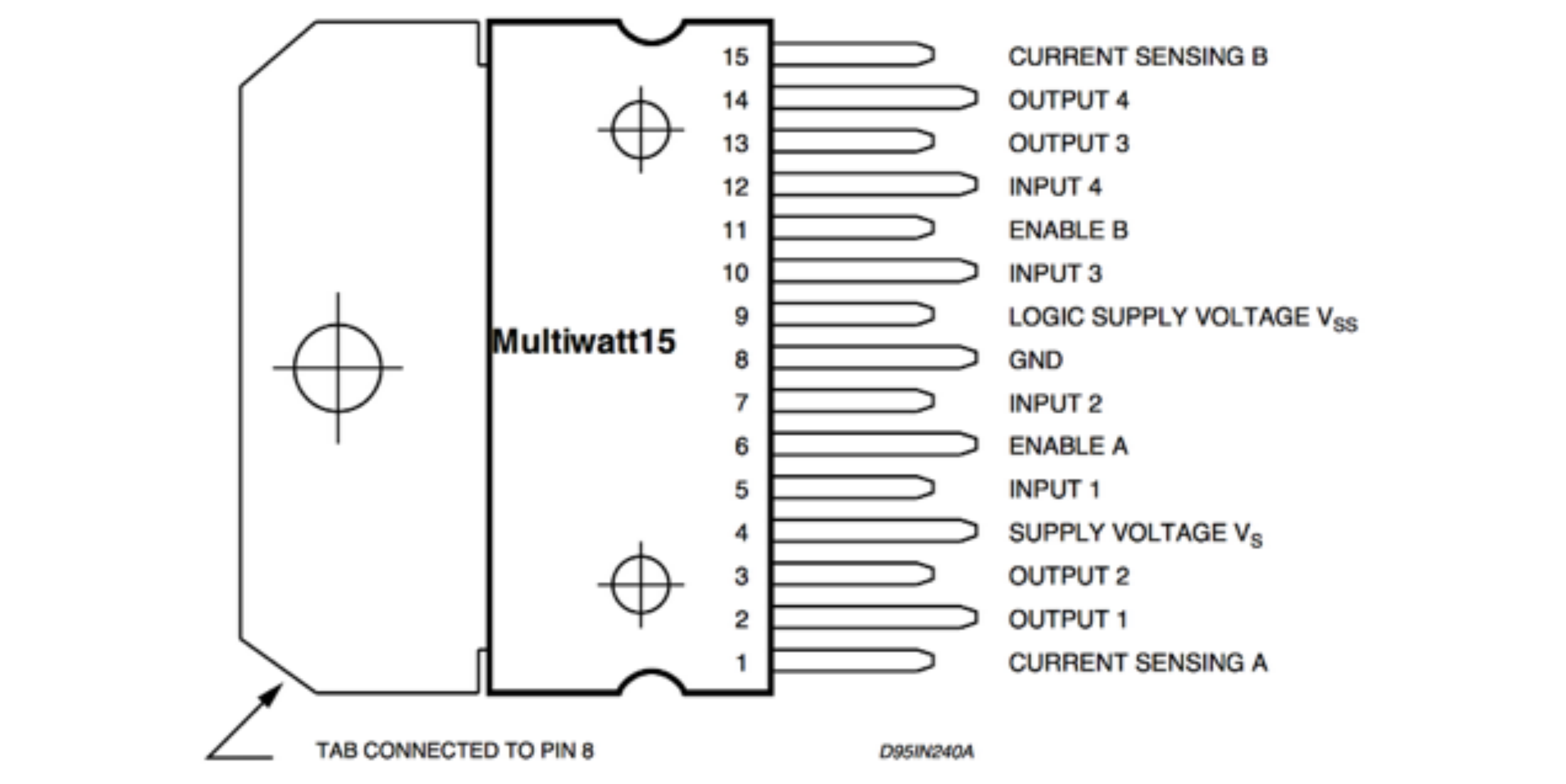
Classic H-Bridge



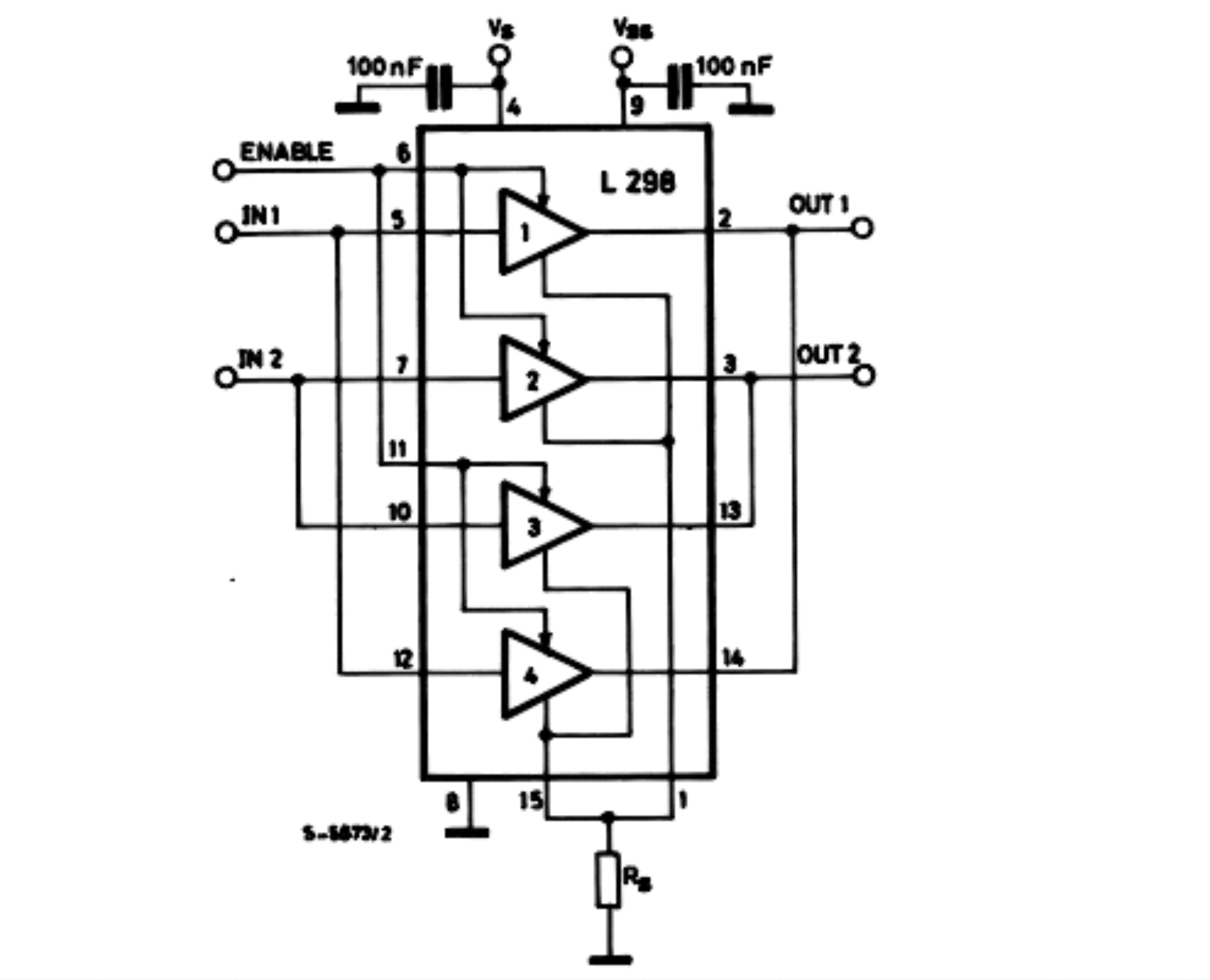
L298 Dual Full-Bridge Driver

The L298 is an integrated monolithic circuit in a 15- lead Multiwatt packages. It is a high voltage, high current dual full-bridge driver de- signed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together. An additional supply input is provided so that the logic works at a lower voltage.

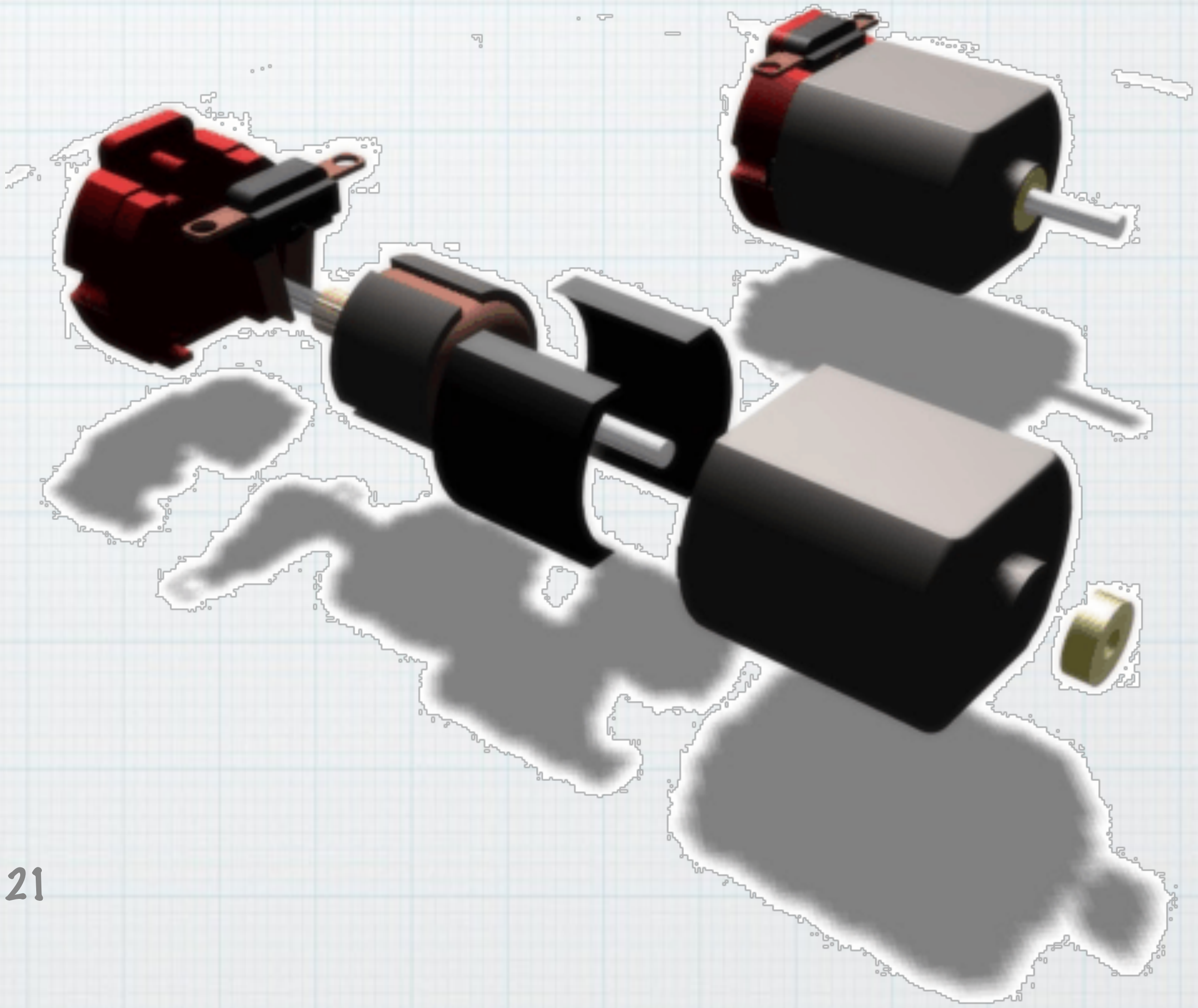




L298 Single Motor

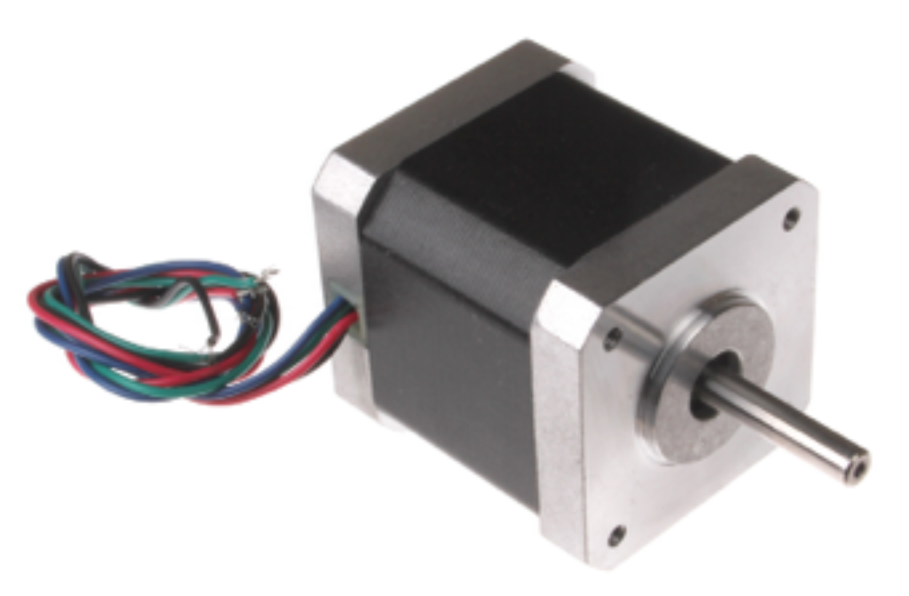


DC - Motors



What is a Stepper Motor

- A **stepper motor** (or step **motor**) is a brushless DC electric **motor** with a series of electromagnetic coils that divides a full rotation into a number of equal steps. The **motor's** position can then be commanded to move and hold at one of these steps without any feedback sensor (an open-loop controller), as long as the **motor** is carefully sized to the application. The center shaft has a series of magnets mounted on it, and the coils surrounding the shaft are alternately given current or not, creating magnetic fields which repulse or attract the magnets on the shaft, causing the motor to rotate.
- This design allows for very precise control of the motor: by proper pulsing, it can be turned in very accurate steps of set degree increments (for example, two-degree increments, half-degree increments, etc.). They are used in printers, disk drives, and other devices where precise positioning of the motor is necessary.
- Two Basic Types:
    - Unipolar Stepper Motors
    - Bipolar Stepper Motors



Stepper Motor (cont)

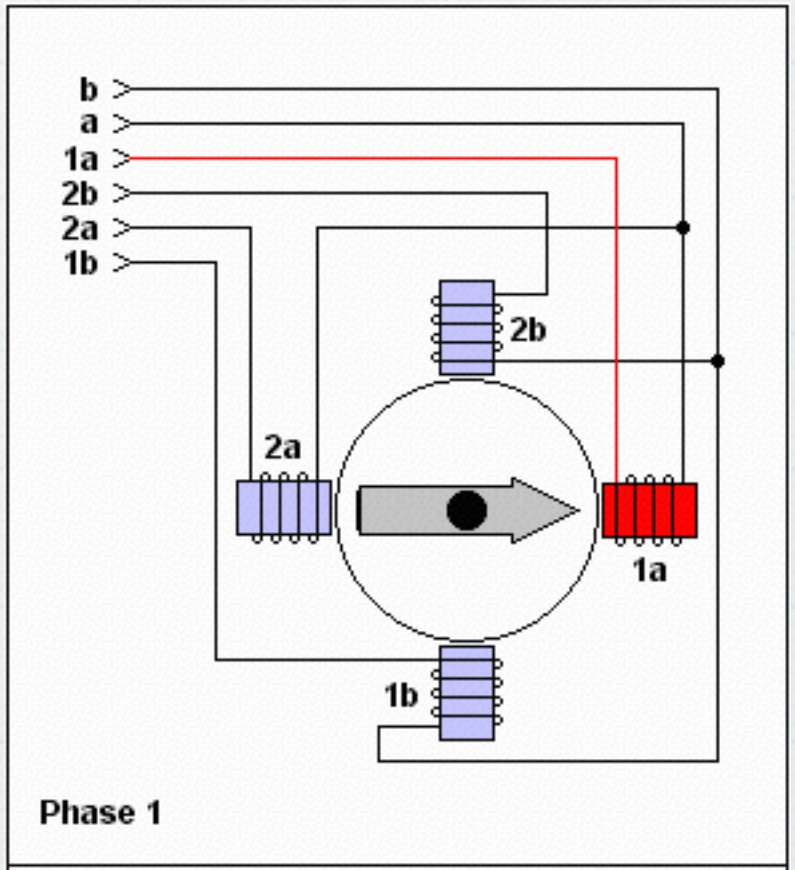
- A stepper motor is a motor controlled by a series of electromagnetic coils. The center shaft has a series of magnets mounted on it, and the coils surrounding the shaft are alternately given current or not, creating magnetic fields which repulse or attract the magnets on the shaft, causing the motor to rotate.
- This design allows for very precise control of the motor: by proper pulsing, it can be turned in very accurate steps of set degree increments (for example, two-degree increments, half-degree increments, etc.). They are used in printers, disk drives, and other devices where precise positioning of the motor is necessary.
- Two Basic Types:

  - Unipolar Stepper Motors - The unipolar stepper motor has five or six wires and four coils (actually two coils divided by center connections on each coil). The center connections of the coils are tied together and used as the power connection. They are called unipolar steppers because power always comes in on this one pole.

  - Bipolar Stepper Motors - The bipolar stepper motor usually has four wires coming out of it. Unlike unipolar steppers, bipolar steppers have no common center connection. They have two independent sets of coils instead.

Stepper Motor (cont)

- The stepper motor is a type of motor that uses the principle of induction to work. Each stepper motor has 2 or more coils surrounding the motor. When electricity flows through a coil a magnetic field is created by the current flow. If the coils are charged in the correct order we can make the motor turn by either pushing or pulling it via magnetism!
- When a coil is active (current flowing through it), a magnetic field is created that pushes the motor a tiny bit. One complete cycle is called a "Step" (every stepper motor will have a unique step size).
- Another very important thing to notice is that each coil requires 2 connections (electric current flows in & then out). That means if a stepper motor has 4 coils, it will require 8 connections. Since we are using a 2 coil stepper motor for this tutorial we will need 4 connections.
- Here is an example of a 4 coil (phase) stepper motor in operation:



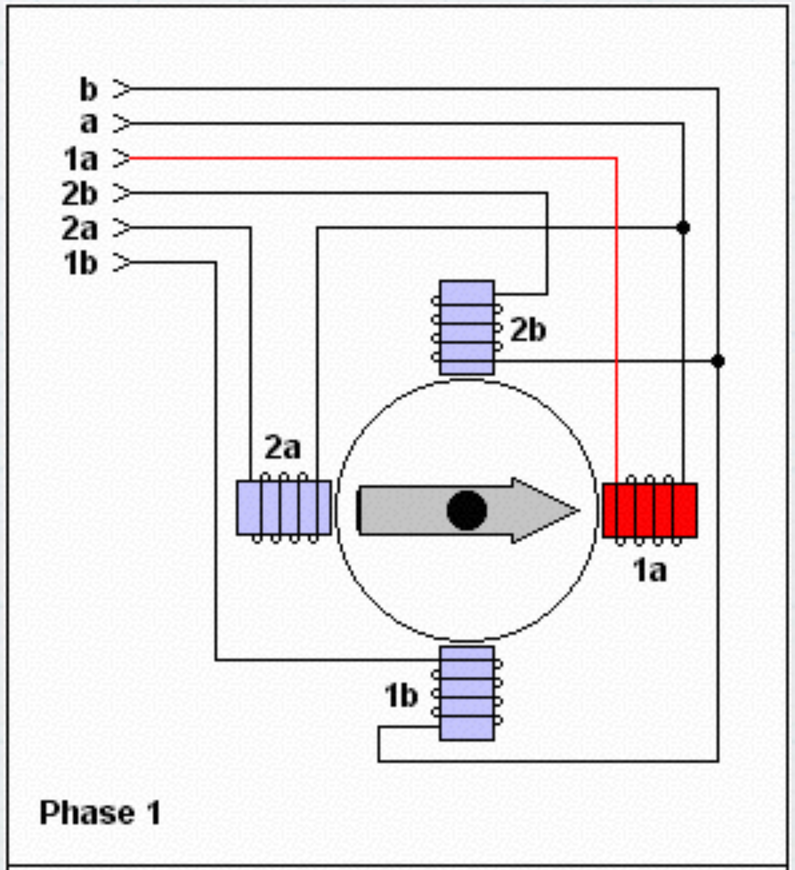
Stepper Motor Operation (Unipolar, Full step)

Stepper Motor (cont)

The Main Algorithm

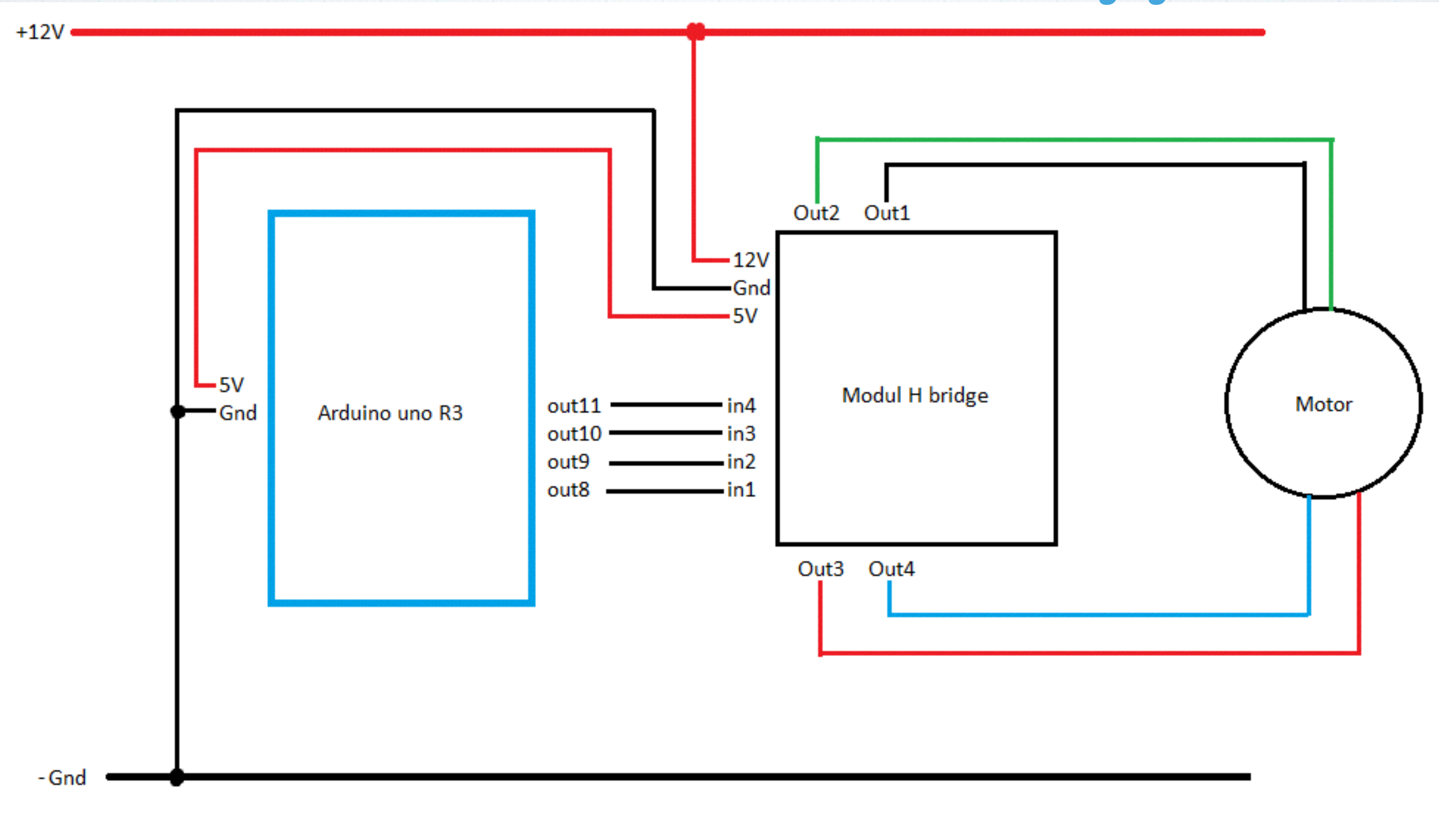
The main algorithm used for completing one step is as follows. The complete stepping cycle for a 2-coil stepper motor goes like this:
AB -> AB* -> A*B* -> A*B -> AB

A and B stand for the two coils in the stepper motor and the * or no star represents the direction of current flow.



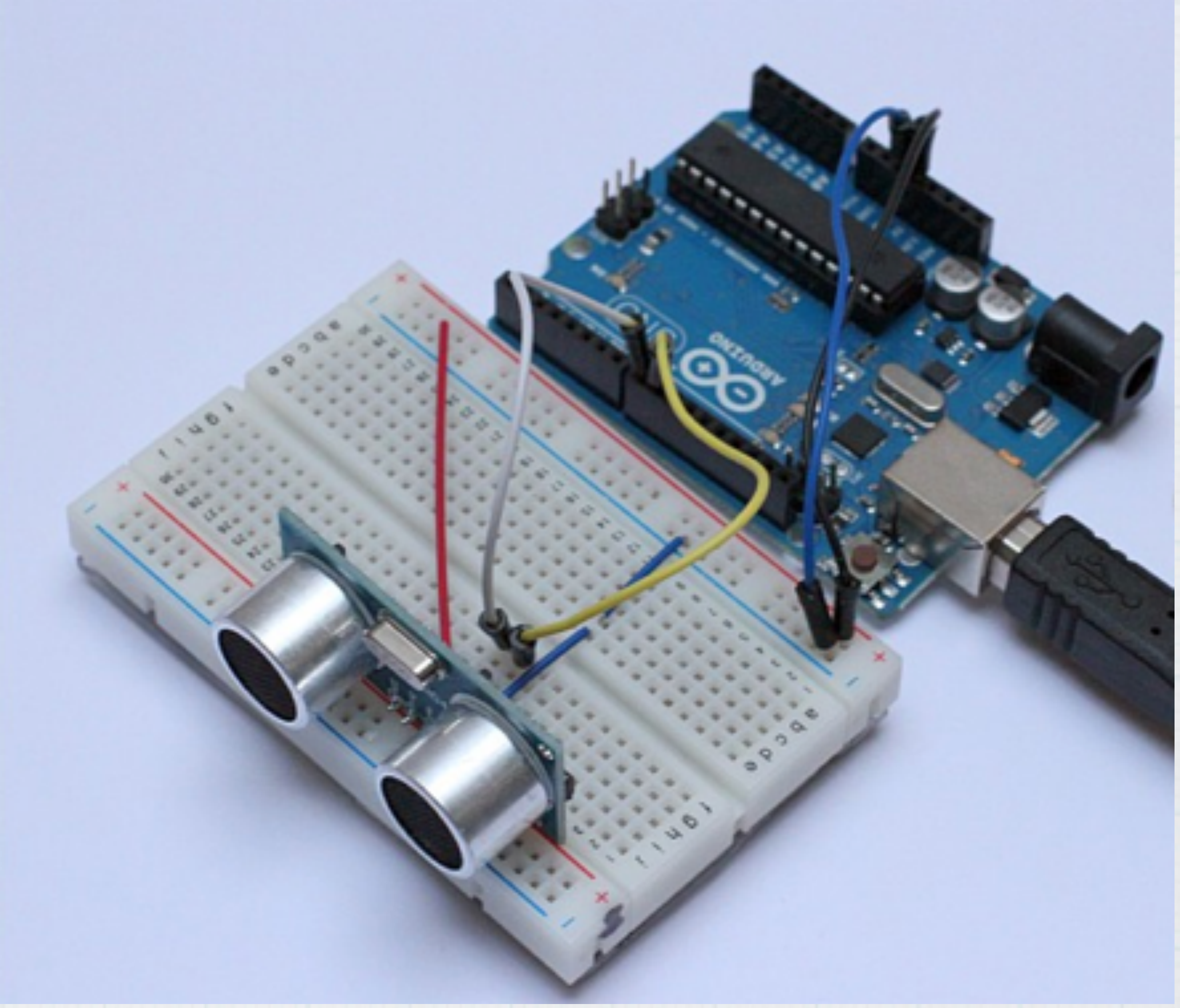
Stepper Motor Operation (Unipolar, Full step)

The L298 With A Stepper

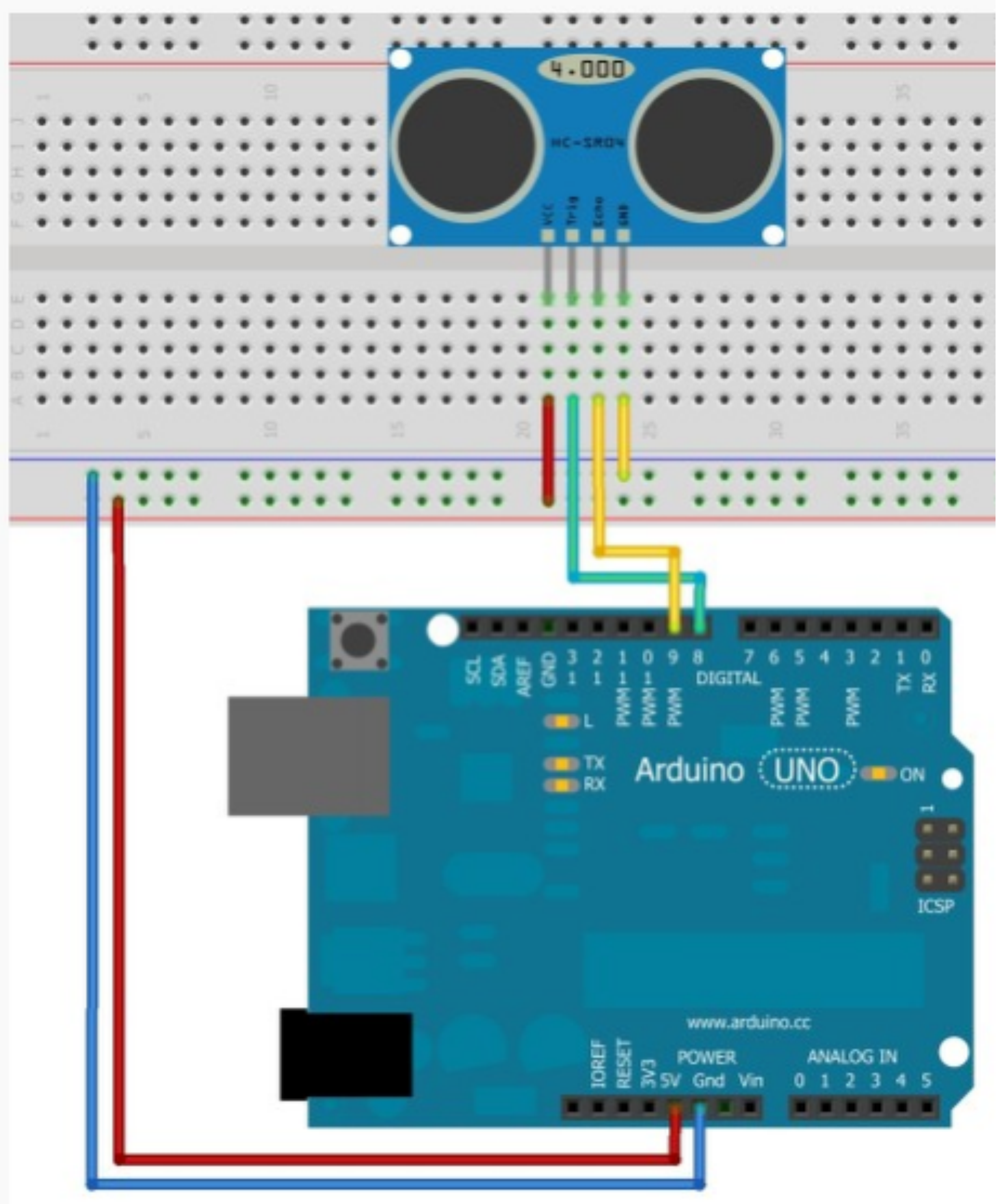


HC-SR04 Ultrasonic Sensor

The way the HC-SR04 works is very simple, you turn the trigger pin high and then low, which initiates a trigger pulse, then wait to receive the echo pulse on another pin called the echo pin. The length divided by 2 of the echo pulse is then calculated as the distance an object is in front of the sensor that is reflecting the pulse. The speed of sound being a constant means that each cm sound will travel xxx cm's.   The range of the SR04 is from 2cm to 400cm within a cone of 15 degrees.

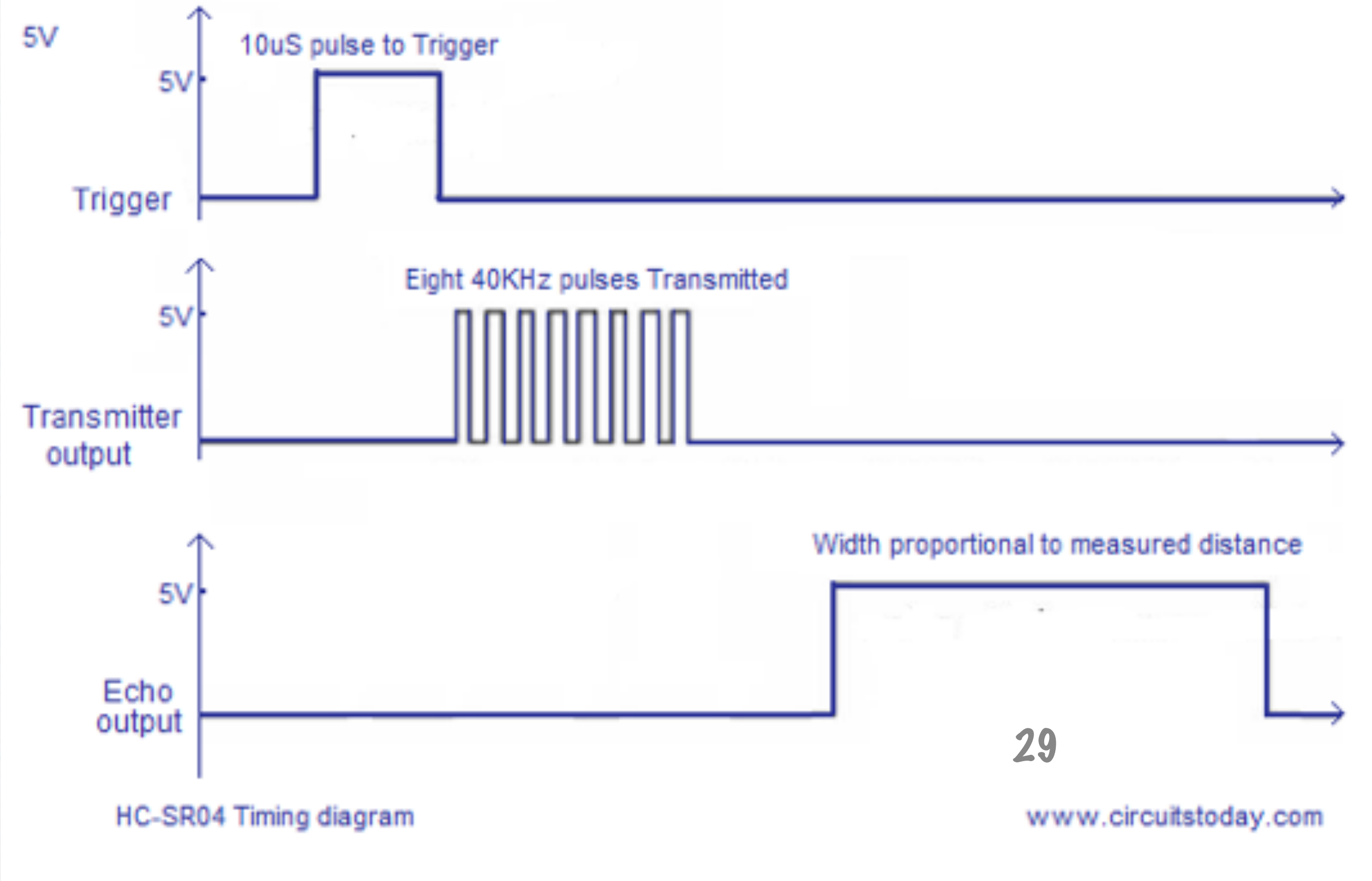


The HC-SR04 Ultrasonic Distance Sensor



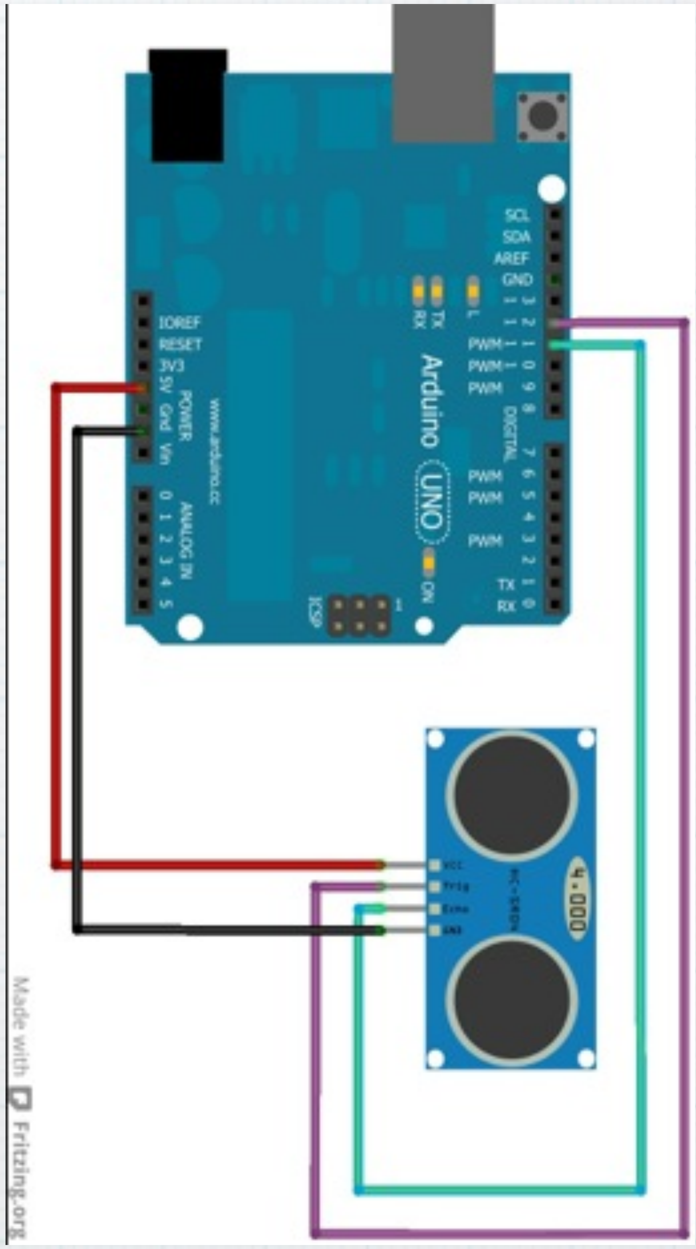
HC-SR04 Ultrasonic Sensor

Can you ping me?

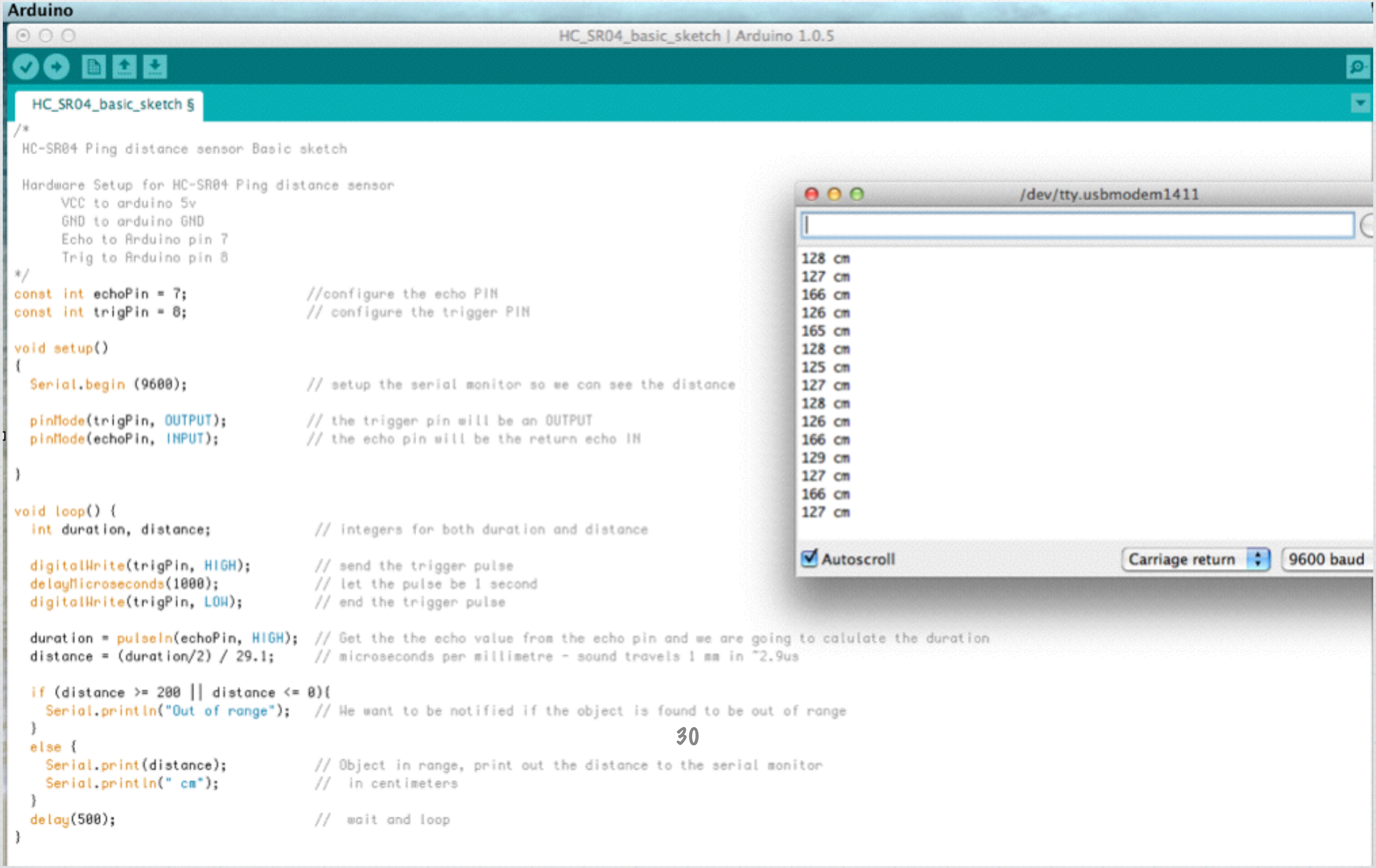

5V
5V
10uS pulse to Trigger
Trigger
5V
Eight 40KHz pulses Transmitted
Transmitter output
5V
Width proportional to measured distance
Echo output

HC-SR04 Timing diagram

www.circuitstoday.com



HC-SR04 Basic Sketch



```
/*
 HC-SR04 Ping distance sensor Basic sketch

 Hardware Setup for HC-SR04 Ping distance sensor
      VCC to arduino 5v
      GND to arduino GND
      Echo to Arduino pin 7
      Trig to Arduino pin 8
*/
const int echoPin = 7;                 //configure the echo PIN
const int trigPin = 8;                 // configure the trigger PIN

void setup()
{
  Serial.begin (9600);                 // setup the serial monitor so we can see the distance

  pinMode(trigPin, OUTPUT);            // the trigger pin will be an OUTPUT
  pinMode(echoPin, INPUT);             // the echo pin will be the return echo IN

}

void loop() {
  int duration, distance;               // integers for both duration and distance

  digitalWrite(trigPin, HIGH);          // send the trigger pulse
  delayMicroseconds(1000);              // let the pulse be 1 second
  digitalWrite(trigPin, LOW);           // end the trigger pulse

  duration = pulseIn(echoPin, HIGH);    // Get the the echo value from the echo pin and we are going to calulate the duration
  distance = (duration/2) / 29.1;       // microseconds per millimetre - sound travels 1 mm in ~2.9us

  if (distance >= 200 || distance <= 0){
    Serial.println("Out of range");     // We want to be notified if the object is found to be out of range
  }
  else {
    Serial.print(distance);             // Object in range, print out the distance to the serial monitor
    Serial.println(" cm");              //  in centimeters
  }
  delay(500);                           //  wait and loop
}
```

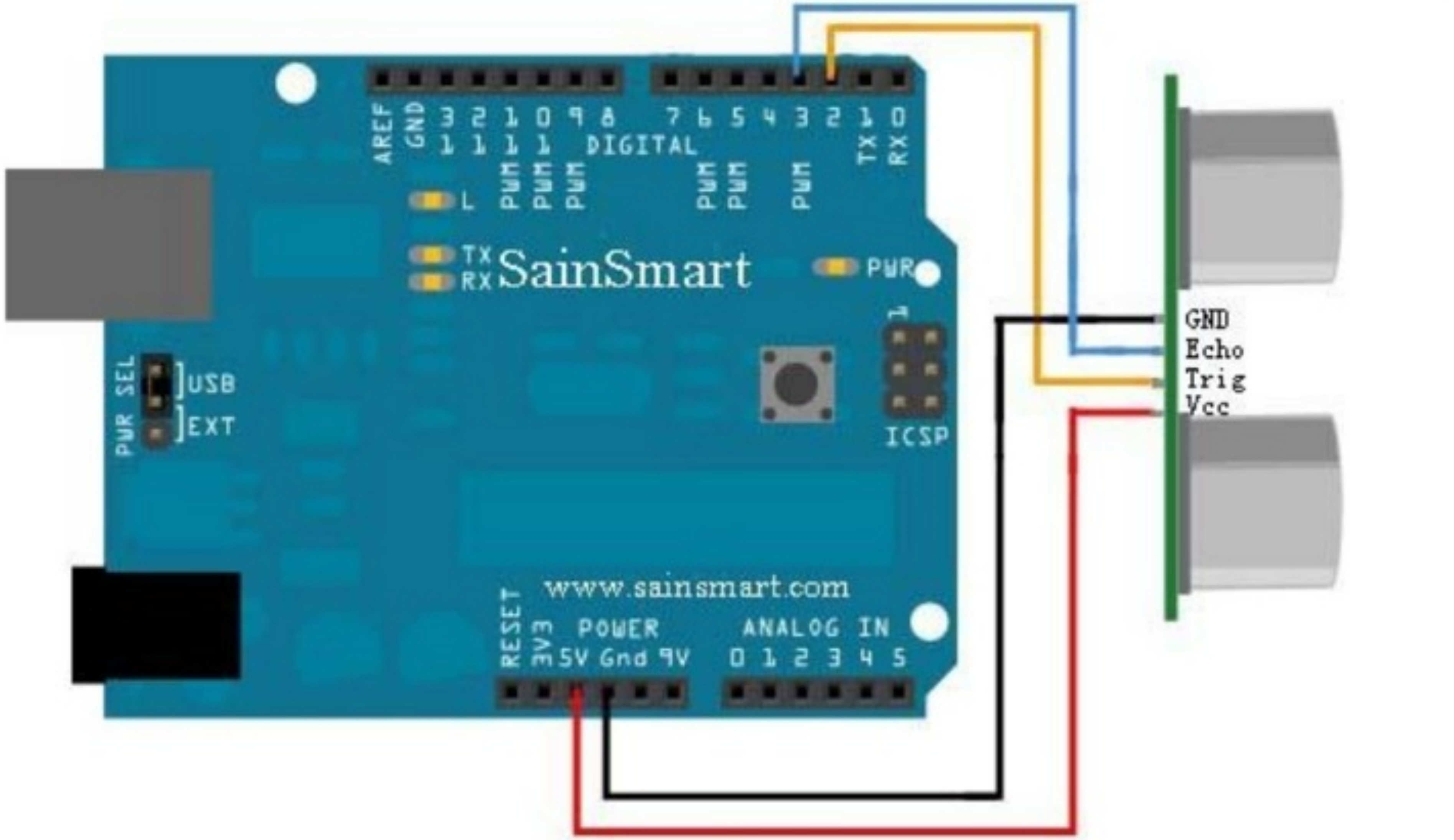
```
128 cm
127 cm
166 cm
126 cm
165 cm
128 cm
125 cm
127 cm
128 cm
126 cm
166 cm
129 cm
127 cm
166 cm
127 cm
```

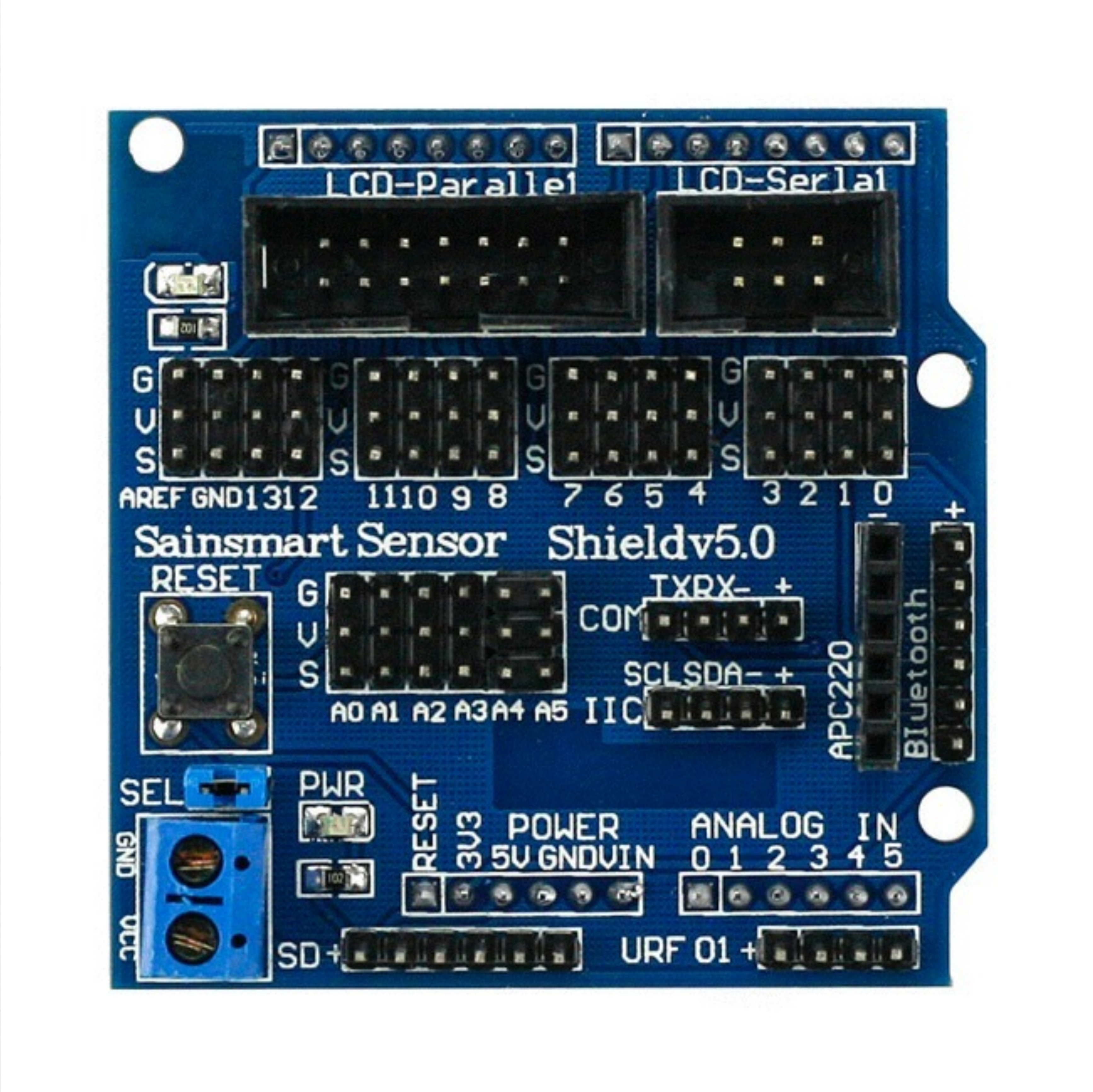
☑ Autoscroll   Carriage return   9600 baud

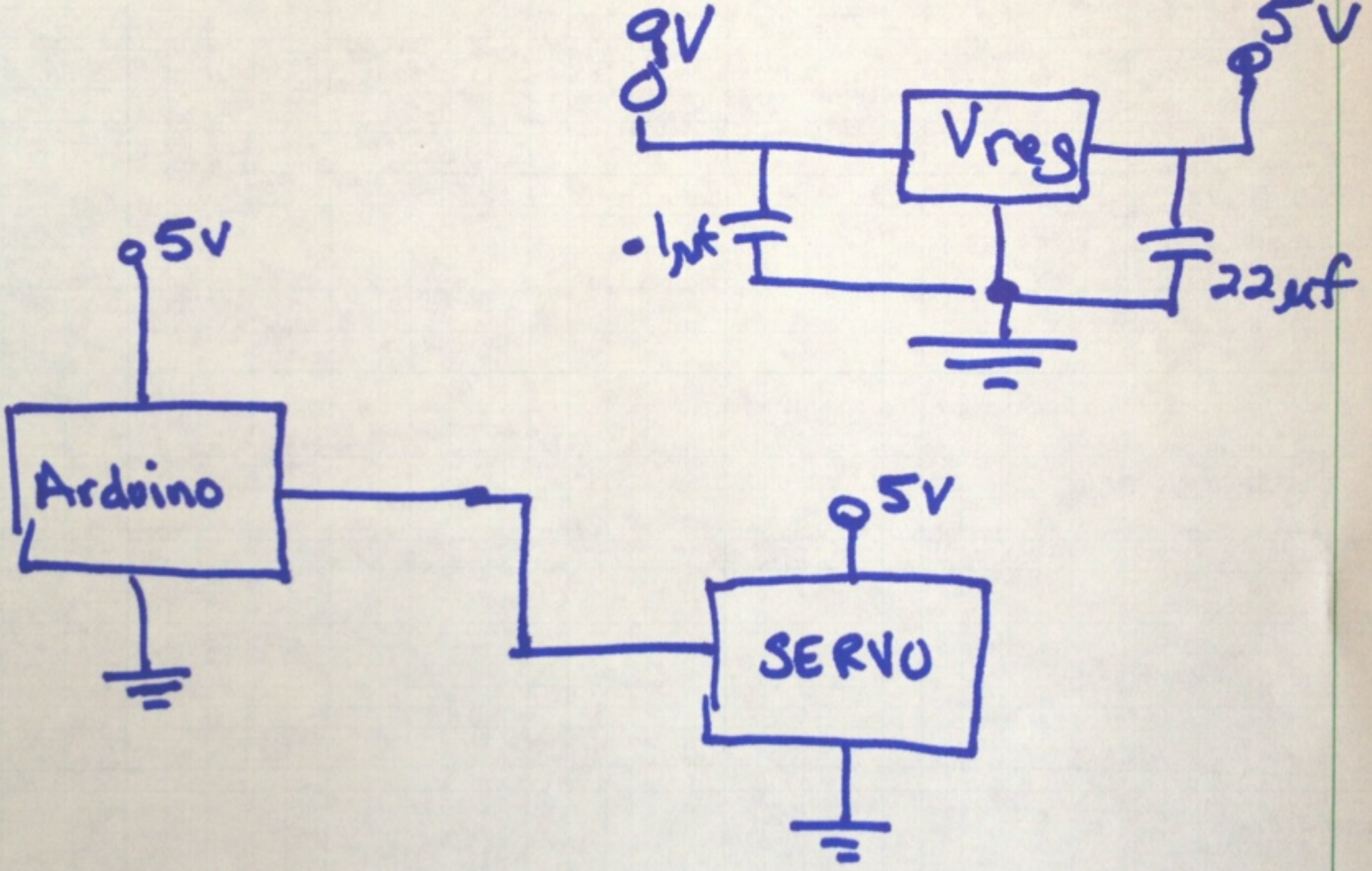
HC-SR04 Method

```
//measure distance, unit “cm”
long MeasuringDistance() {
     // Calculates the Distance in cm
   float cm;                        //define variables for distance sensor
   // ((time)*(Speed of sound))/ toward and backward of object) = Width of Echo pulse, in uS (micro second)
   // How to call the function:    long Distance_cm = Distance(Duration);   // Use function to calculate the distance
long duration;
long adjust = 1.15;                  // Calibration adjustment based on actual measurement test
//pinMode(TrigPin, OUTPUT);
digitalWrite(TrigPin, LOW);
delayMicroseconds(2);
digitalWrite(TrigPin, HIGH);
delayMicroseconds(5);
digitalWrite(TrigPin, LOW);
//pinMode(EchoPin, INPUT);
duration = pulseIn(EchoPin, HIGH);
return duration / 29 / 2 + adjust;       // Actual calculation in centimeters
```

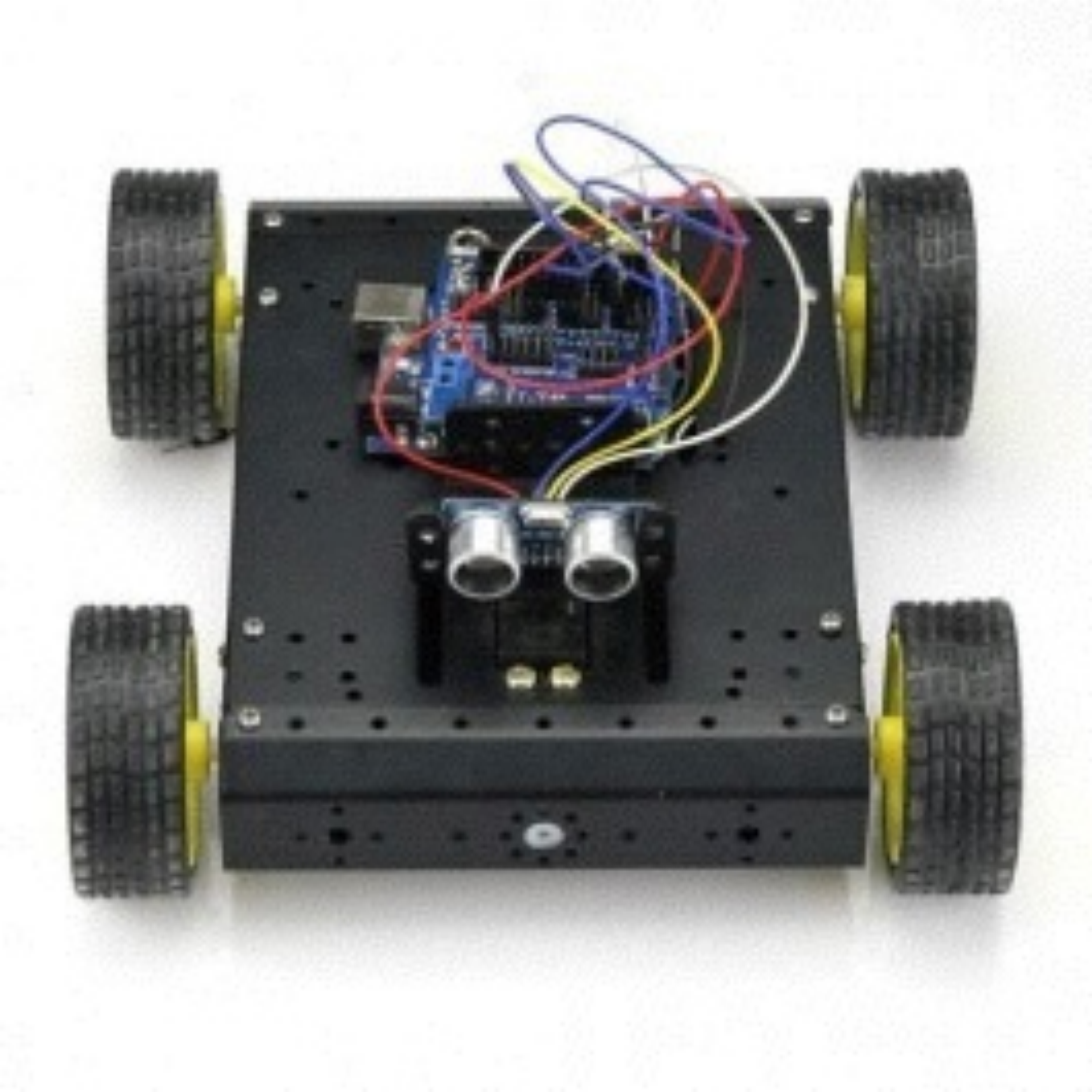
www.sainsmart.com



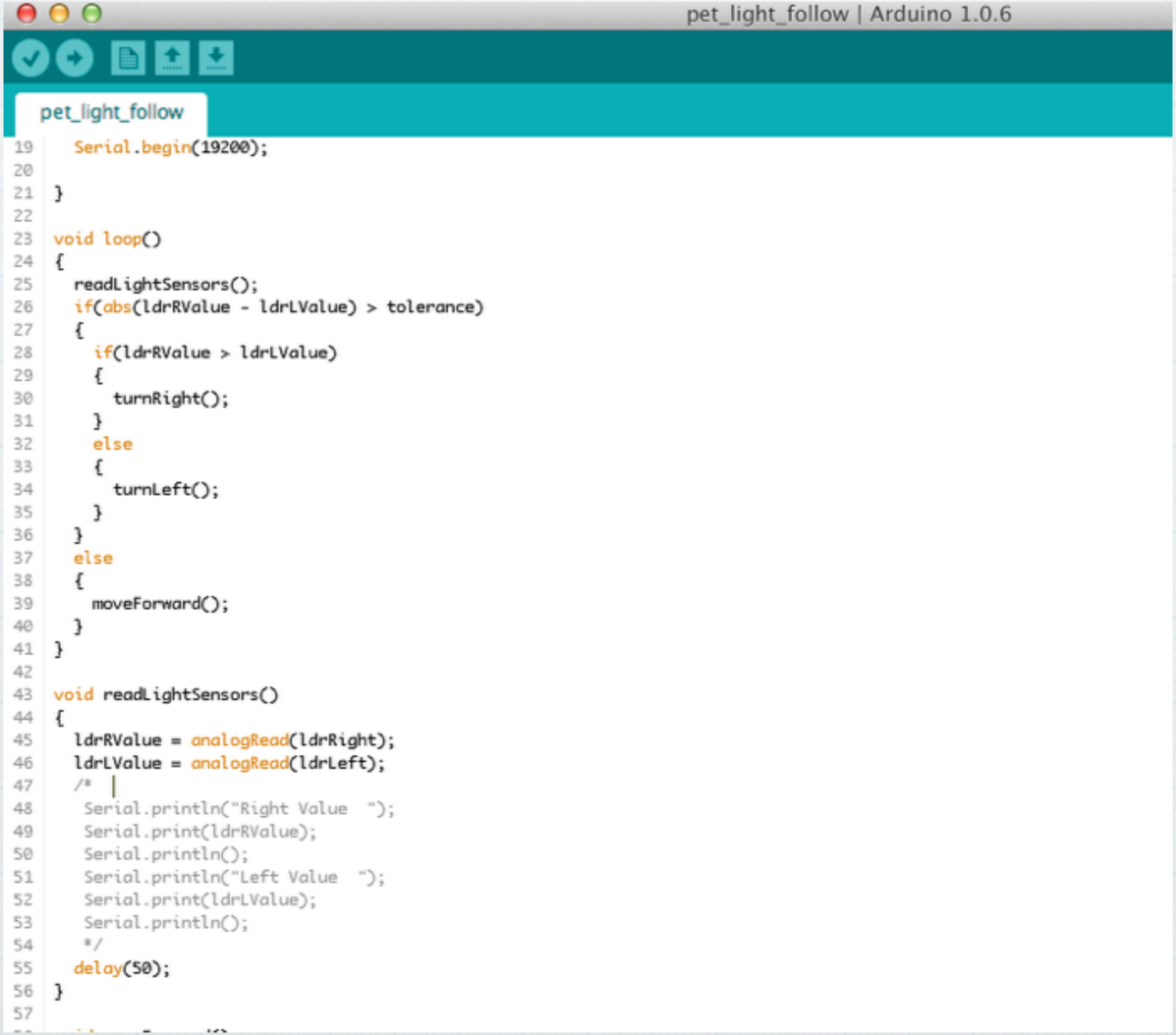

LCD-Parallel
LCD-Serial
G V S
AREF GND 13 12
11 10 9 8
7 6 5 4
3 2 1 0
Sainsmart Sensor Shieldv5.0
RESET
A0 A1 A2 A3 A4 A5
TXRX- +
COM
SCLSDA- +
IIC
- +
APC220
Bluetooth
SEL
PWR
GND
VCC
RESET
3V3
POWER
5V GND VIN
ANALOG IN
0 1 2 3 4 5
SD+
URF 01 +



* More than 2 servos need their own power supply



Light Sensor Example



```
  Serial.begin(19200);

}

void loop()
{
  readLightSensors();
  if(abs(ldrRValue - ldrLValue) > tolerance)
  {
    if(ldrRValue > ldrLValue)
    {
      turnRight();
    }
    else
    {
      turnLeft();
    }
  }
  else
  {
    moveForward();
  }
}

void readLightSensors()
{
  ldrRValue = analogRead(ldrRight);
  ldrLValue = analogRead(ldrLeft);
  /*
   Serial.println("Right Value  ");
   Serial.print(ldrRValue);
   Serial.println();
   Serial.println("Left Value  ");
   Serial.print(ldrLValue);
   Serial.println();
   */
  delay(50);
}
```

Code Respositories

- https://github.com/gmossy/Sainsmart-4WD-Robot

Reference Links

- http://tech-zen.tv/episodes/shows/make-it-work/episodes/stepper-and-dc-motor-control-with-arduino-episode-36
- http://www.mkme.org/index.php/arduino-sainsmart-4wd-robot/
- http://www.mcmanis.com/chuck/robotics/tutorial/h-bridge/images/basic-bridge.gif
- http://www.bristolwatch.com/L298N/L298N_arduino.htm
- http://www.tigoe.net/pcomp/code/circuits/motors/stepper-motors/
- https://www.youtube.com/watch?v=XZLVpfydUdw