SWS3009 – Part B
Robotics / TeleOperation

Section 3: Arduino Mega

Outline

- **Arduino Basics**
  - Introduction
  - Programming
  - External components demonstration

ARDUINO BASICS

Arduino Family says Hi!

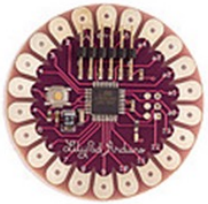
Arduino Uno

Arduino Leonardo

Arduino Ethernet

Arduino Pro

Arduino Mega 2560



Arduino LilyPad

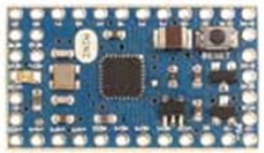
Arduino BT

Arduino Nano

Arduino Mega ADK

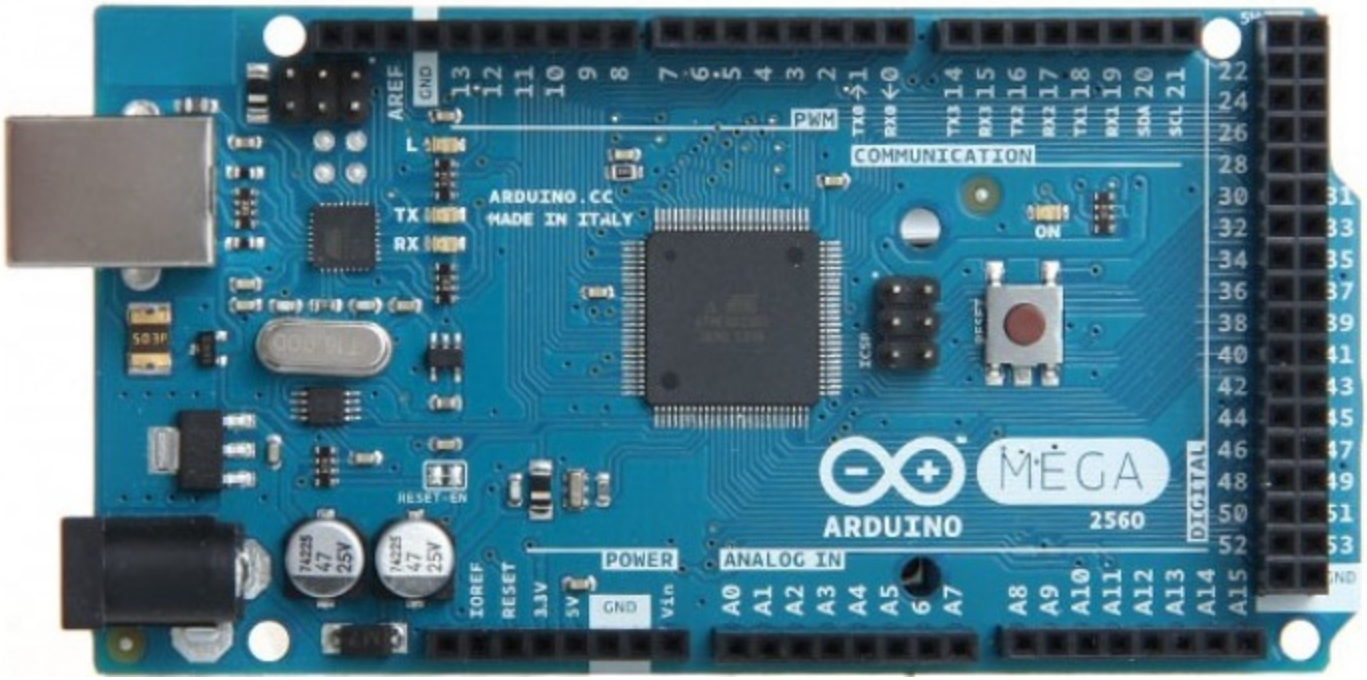
Arduino Fio

USB/Serial Light Adapter



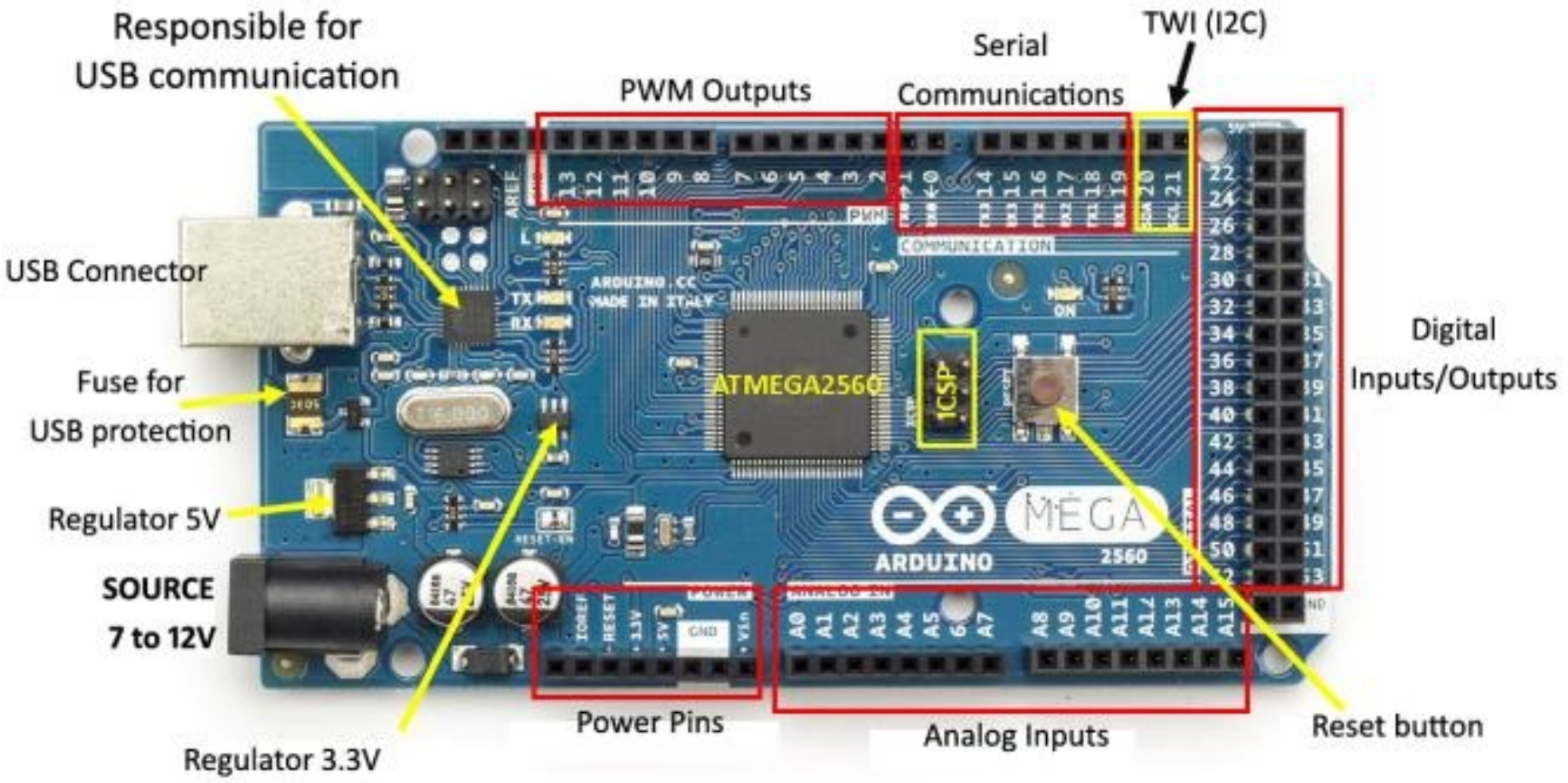
Arduino Mini

Our Arduino….



- **Arduino Mega 2560** (Revision 3)
  - Note that the reset button may be located differently (e.g. corner of the board)

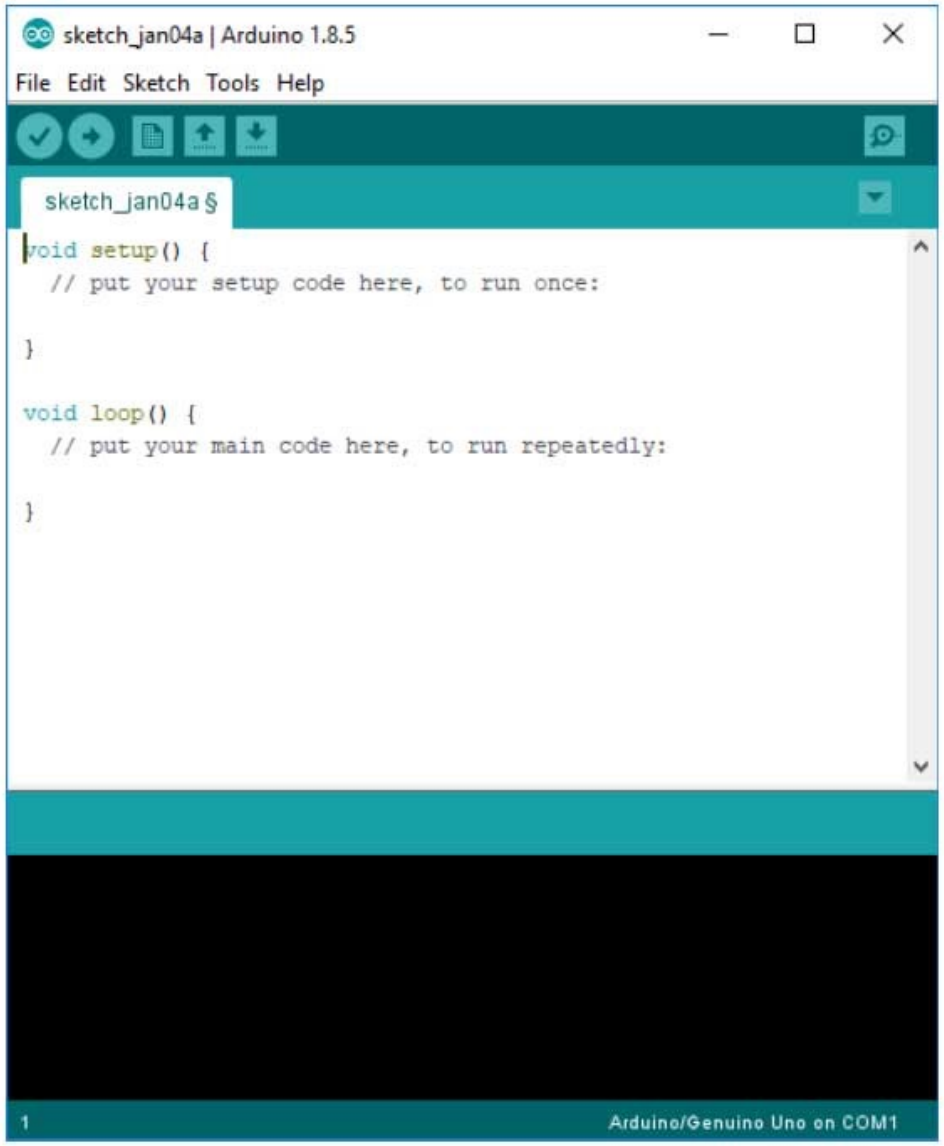
Arduino Mega 2560 - Layout



Arduino Mega: Basic Facts

- Microcontroller Board
  - Based on ATmega2560
- USB connection
  - Alternative Powering Option: Power Jack
- Input / Output Capabilities
  - 54 digital input/output pins
    - 14 can be used for ***PWM*** output (more later)
  - 16 analog inputs
    - Digitized to 10 bits (i.e. 1024 levels)
  - 4 UART (Serial Ports)
  - 16MHz Crystal Oscillator (Clock Signal)

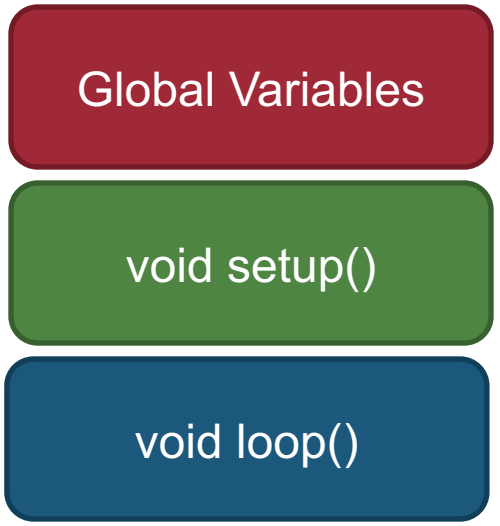
Introducing Arduino IDE



- Arguably the main reason for Arduino's popularity
- Minimalistic but beginner friendly and reasonably powerful

Programming on Arduino

- Programming Language:
  - Subset of **C/C++**
  - Limited set of **C/C++** libraries
  - Additional set of Arduino specific calls
    - Online Reference at https://www.arduino.cc/en/Reference/HomePage
- Program structure:
  - Minimally need the setup() and loop() functions
  - Additional functions can be declared



Blink: An example program

```
void setup() {

   pinMode(13, OUTPUT);

}

void loop() {

  digitalWrite(13, HIGH);
  delay(1000);

  digitalWrite(13, LOW);
  delay(100);

}
```

- **pinMode(pin, mode)**
  - pin = pin number
  - mode = { INPUT, OUTPUT }
- **digitalWrite(pin, value)**
  - pin = pin number
  - value = 0 / 1 (use LOW / HIGH constant!)
- **delay(ms)**
  - ms = millisecond

Question: Let's Blink Differently

- How do we change the blinking patterns as following:

- **A:** Blink twice in a second

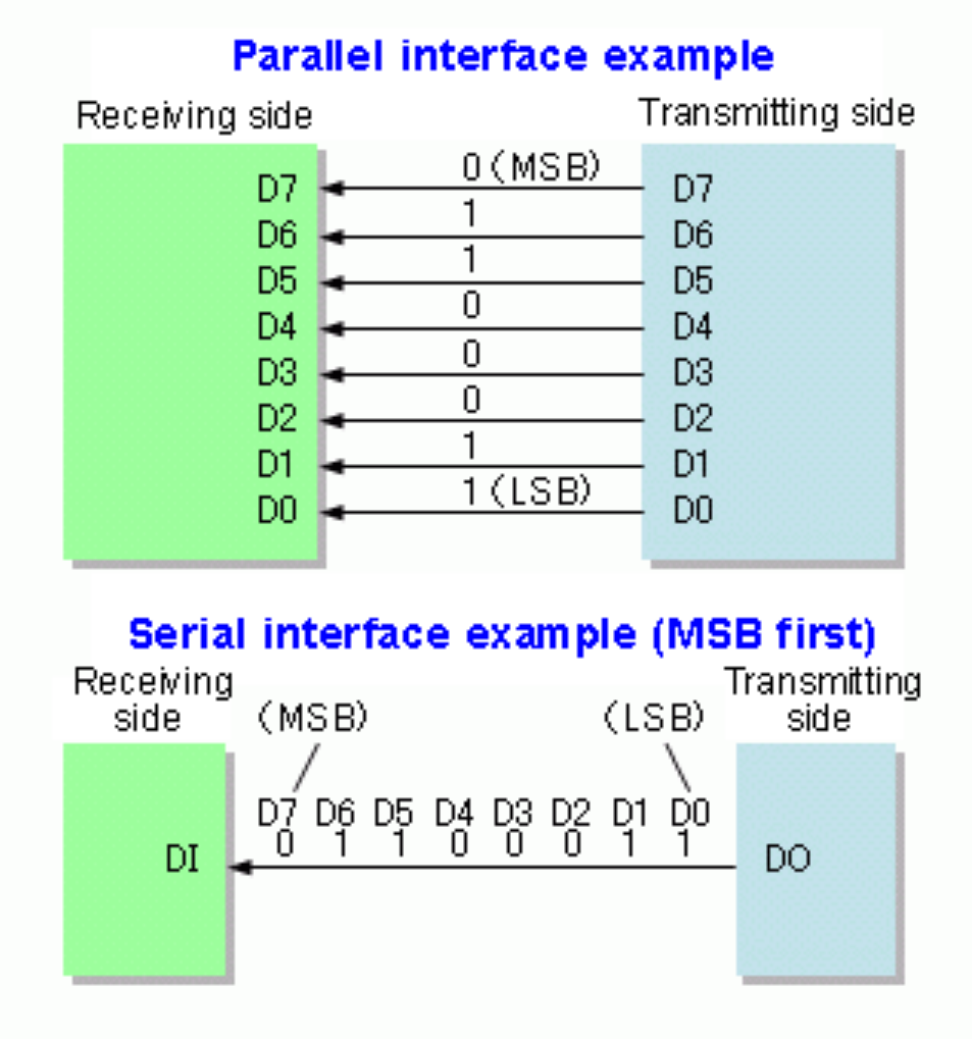
- **B:** Blink short-short-long

- **C:** Blink randomly
  - (Hint: Look for the correct library call in Arduino)

Communication

- Since we cannot easily debug the code on the Arduino directly
  - we will use **basic serial communication** as a "debugging" mechanism

Serial Communication: Idea



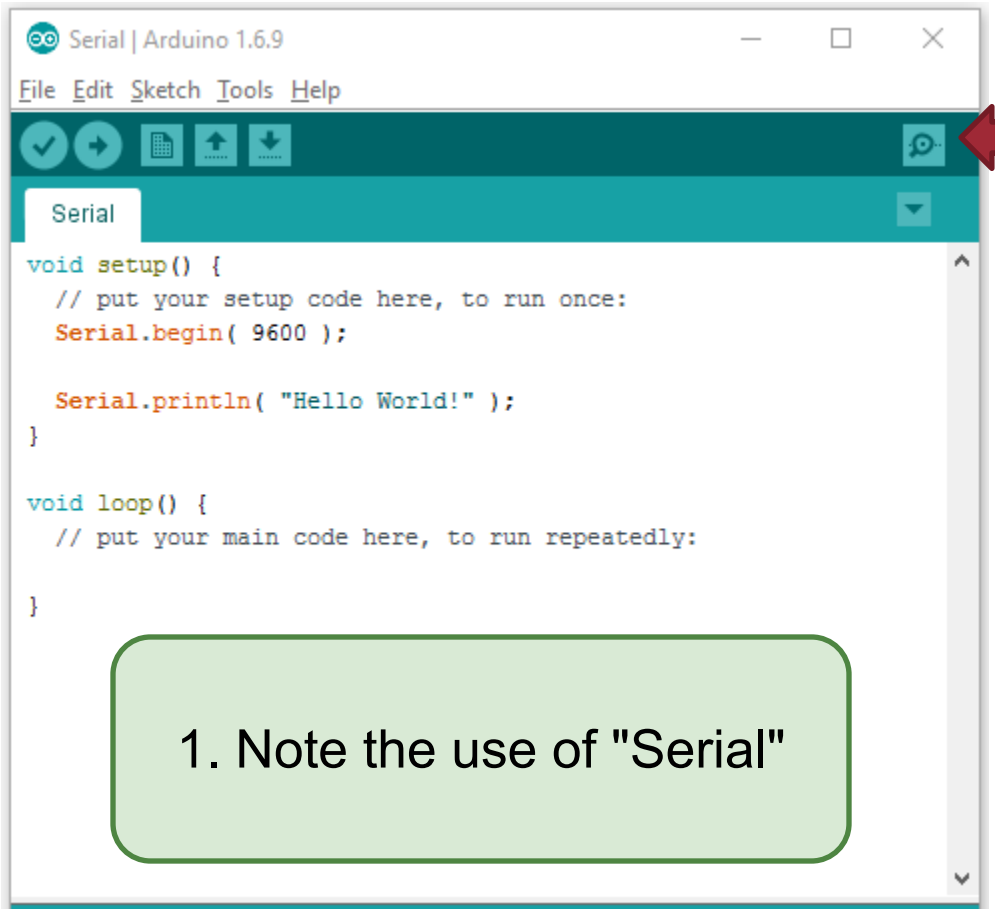
- Serial Communication is a very old idea
- Still commonly supported:
  - Only need **2 wires** for two way communication!

Serial Communication: **Arduino Mega**

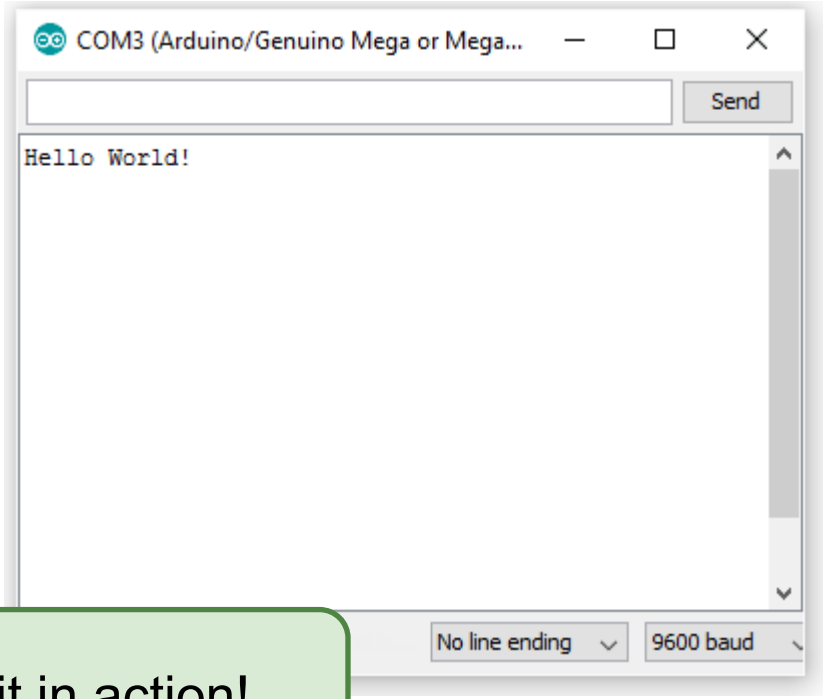
- Arduino Mega support **4 sets of serial communication** (two ways each):
  - **Serial 0**: Through the USB connection or Pin 0 (Receive RX) + Pin 1 (Transmit TX)
  - **Serial 1**: Pin 19 (RX) + Pin 18 (TX),
  - **Serial 2**: Pin 17 (RX) + Pin 16 (TX)
  - **Serial 3**: Pin 15 (RX) + Pin 14 (TX)

- **Be careful:** Check the operating voltage of the device you want to communicate with:
  - Arduino Mega operates at **5 volt**

Serial Communication: Code



2. Click (or press Ctrl-Shift-M") to open the serial monitor



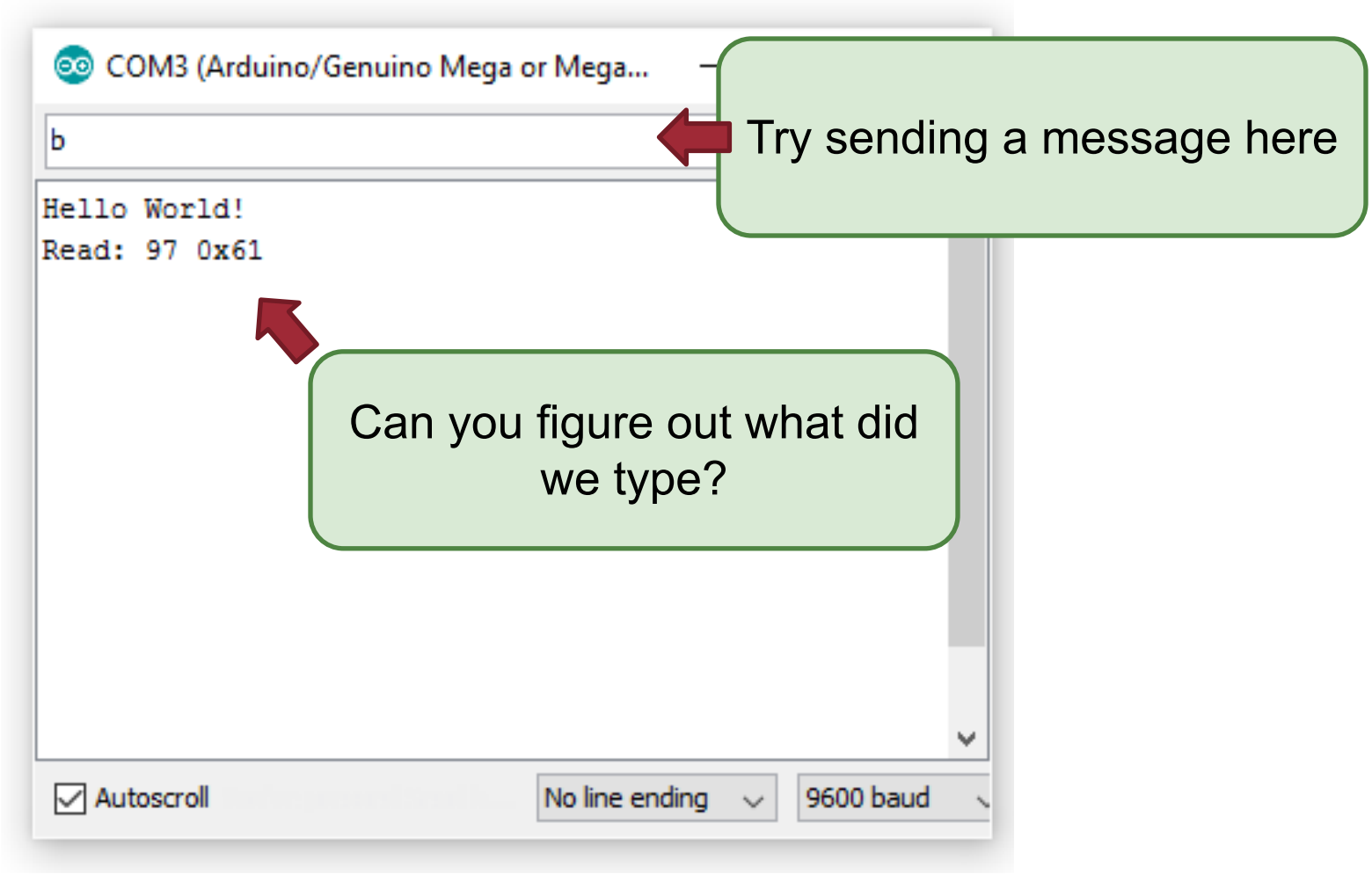
1. Note the use of "Serial"

3. Let's see it in action!

Serial Communication: Code 2

```
void setup() {

  Serial.begin( 9600 );
  Serial.println( "Hello World!" );

}

void loop() {

  if (Serial.available()) {
        int inByte = Serial.read();
        Serial.print("Read: ");
        Serial.print(inByte);
        Serial.print(" 0x");
        Serial.println(inByte, HEX);
    }

}
```

Serial Communication: Code 2



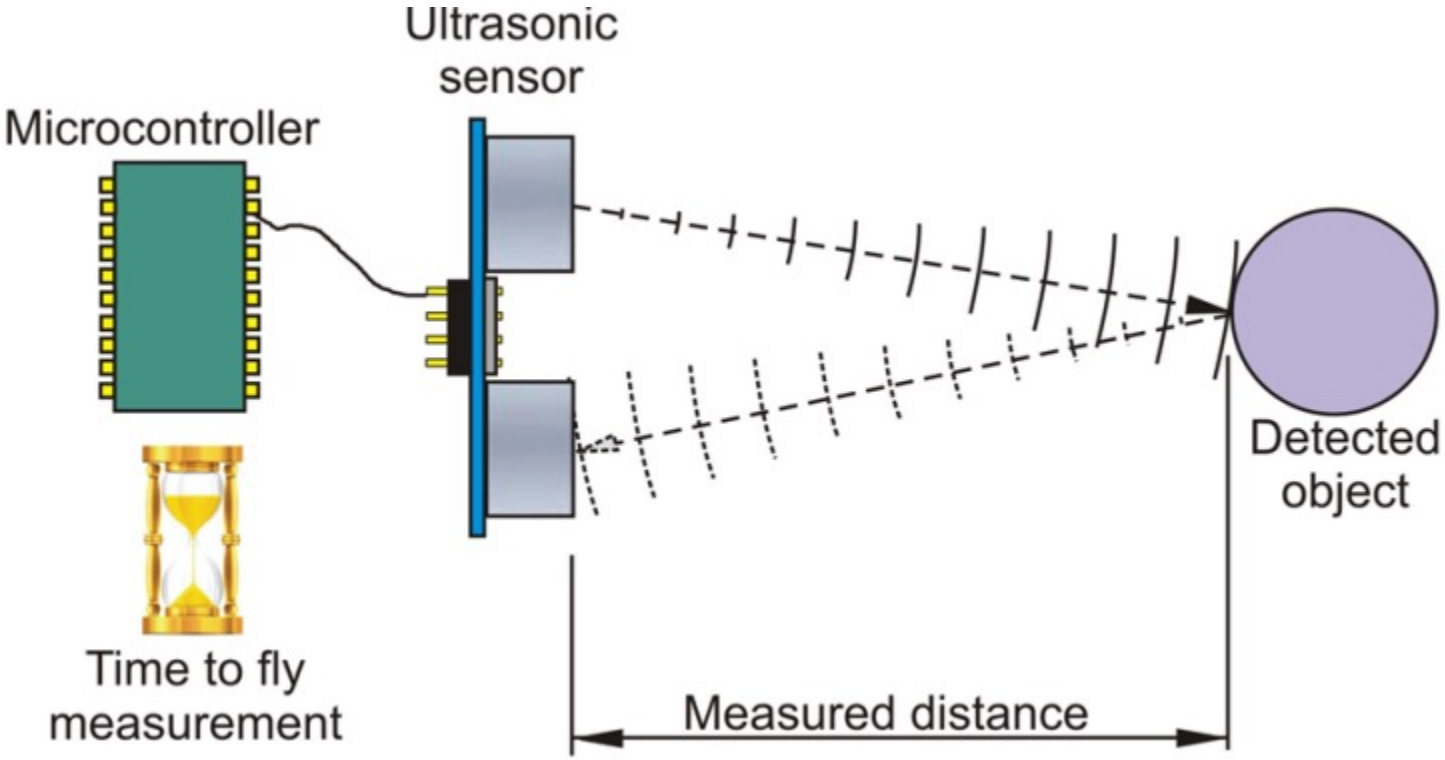
EXTERNAL COMPONENTS

Interfacing with Components

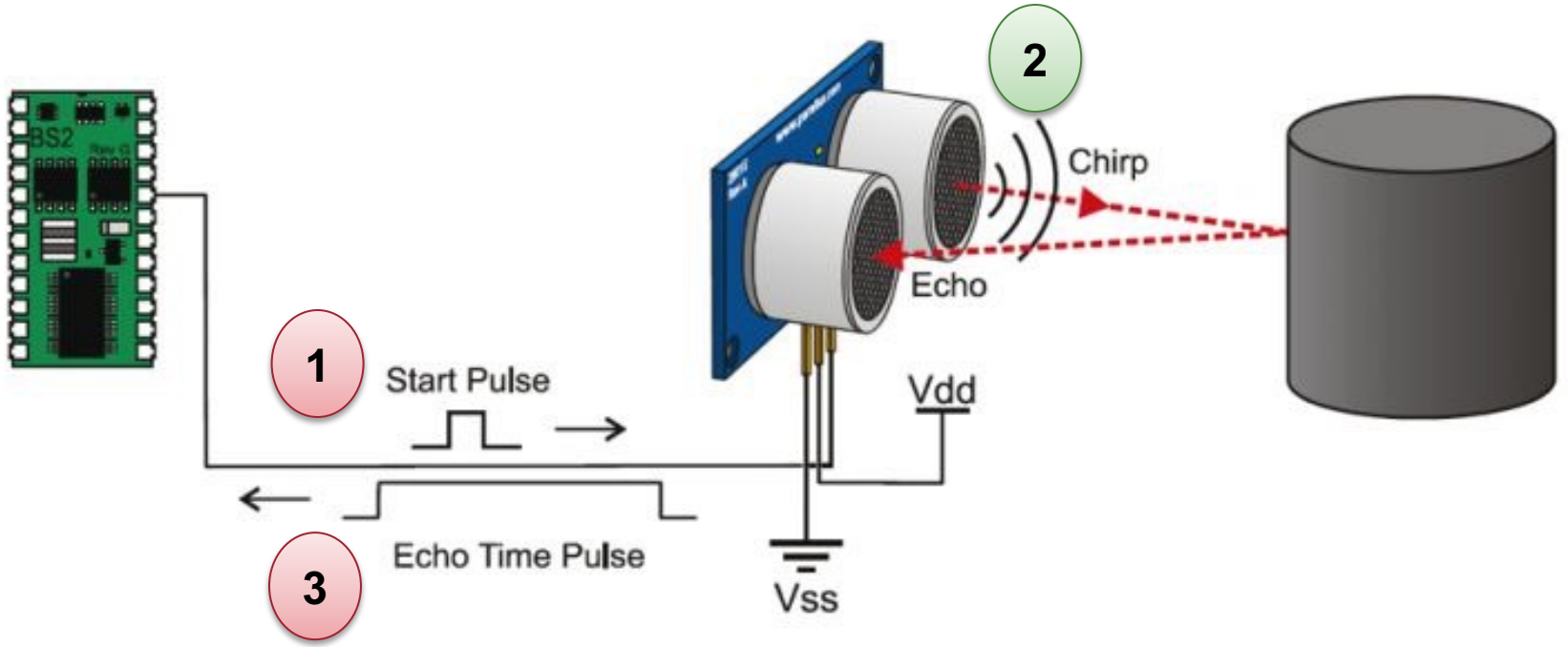
- Arduino can interface with many electronic components

- **Common steps:**
  1. Connect the electronic components to the correct pin(s)
     - Usually wiring diagram is provided
  2. Write code to interact with the component:
     - Sometimes, libraries are provided together with the components

Ultrasonic Sensor - Principle

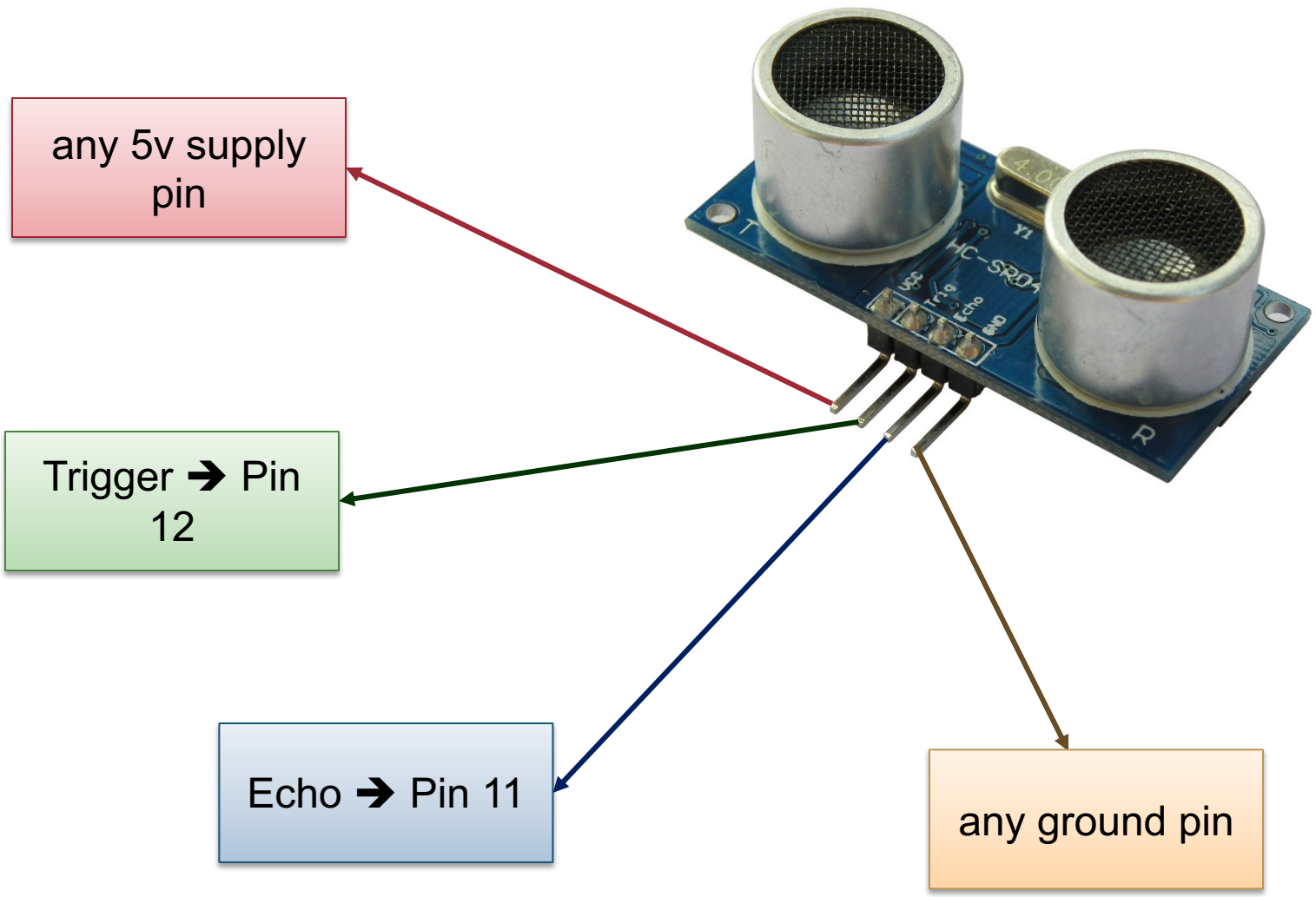


Ultrasonic Sonic - Implementation

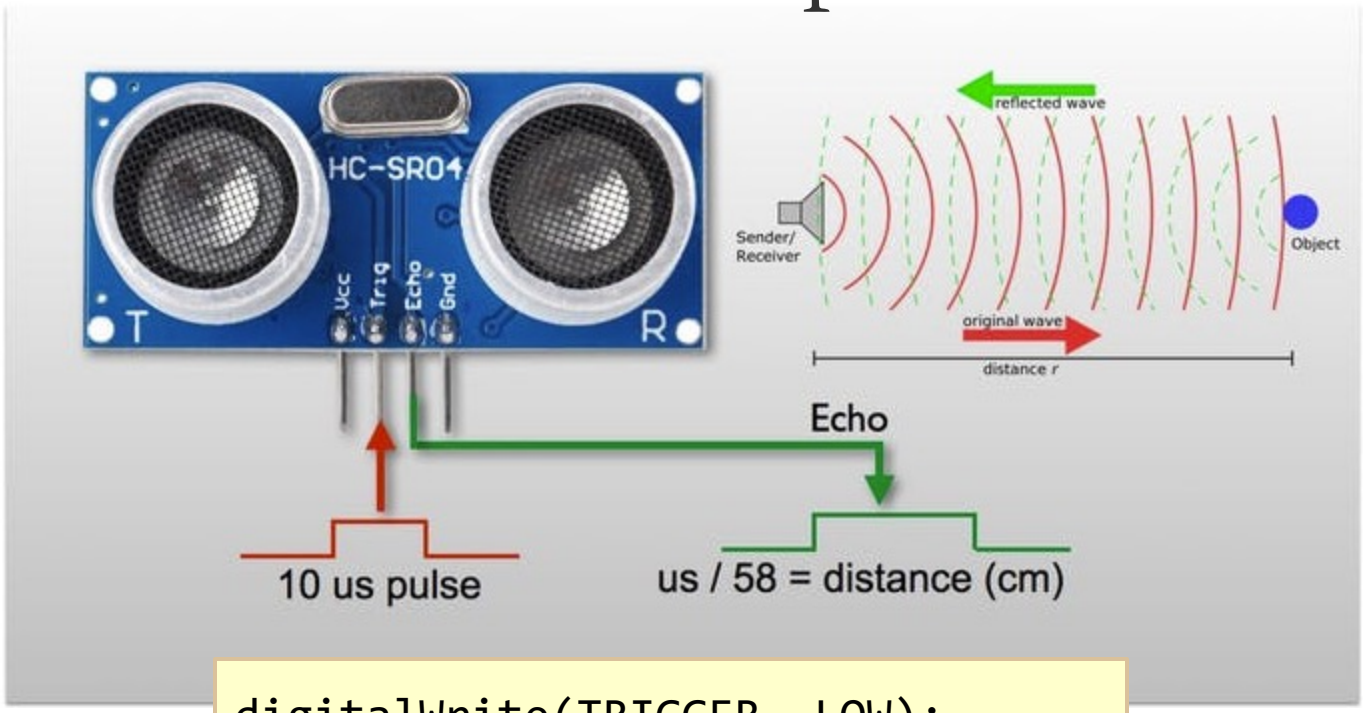


- Given **Echo Time Pulse** (i.e. time difference between Chirp and Echo), how do we get the distance, **D**?

Ultrasonic Sensor: Connection



Ultrasonic Sensor: Operation

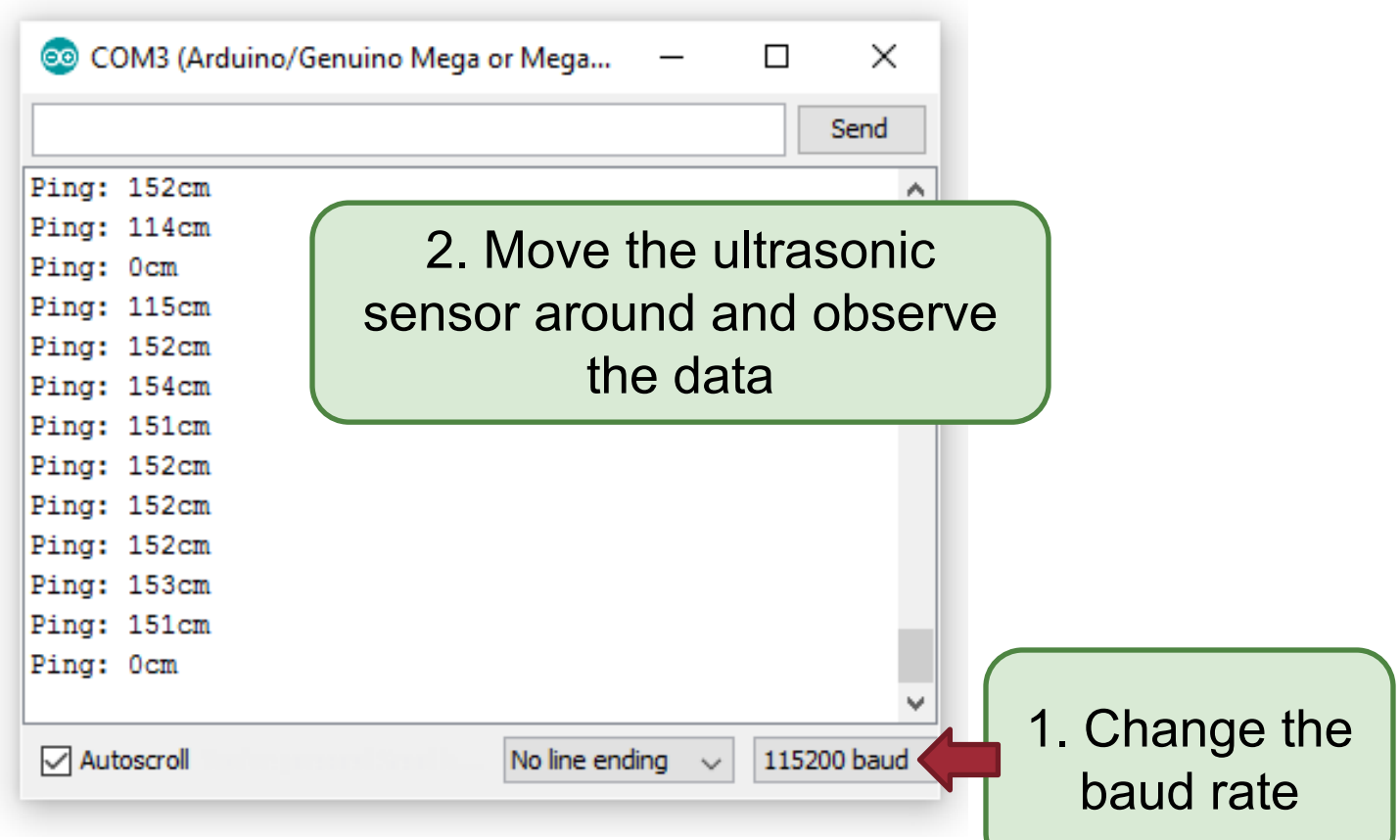



```
digitalWrite(TRIGGER, LOW);
delayMicroseconds(2);
digitalWrite(TRIGGER, HIGH);
delayMicroseconds(10);
digitalWrite(TRIGGER, LOW);
duration = pulseIn(ECHO, HIGH);
cm = duration / 58 ;
```

Ultrasonic Sensor: **Adding Libraries**

- In this case, the ultrasonic sensor manufacturer provided sample library to use the sensor in a painless way
- We will demonstrate how to add a library and use sample code
- [For exploration] You can take a look in the library code to see the details, e.g. how to trigger a pulse, how to measure echo time pulse, etc

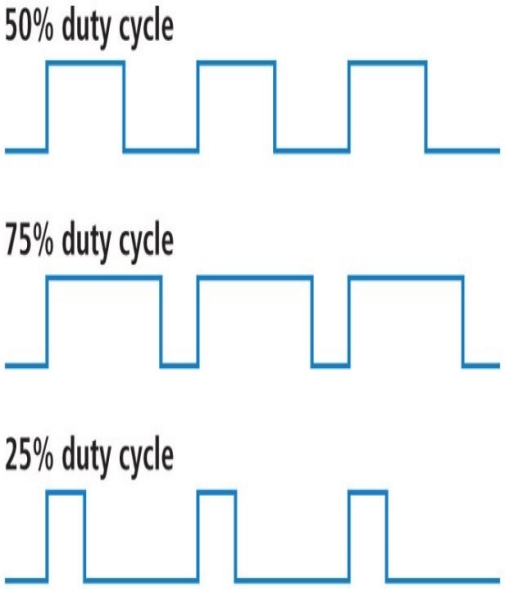
Ultrasonic Sensor: **Code Example**

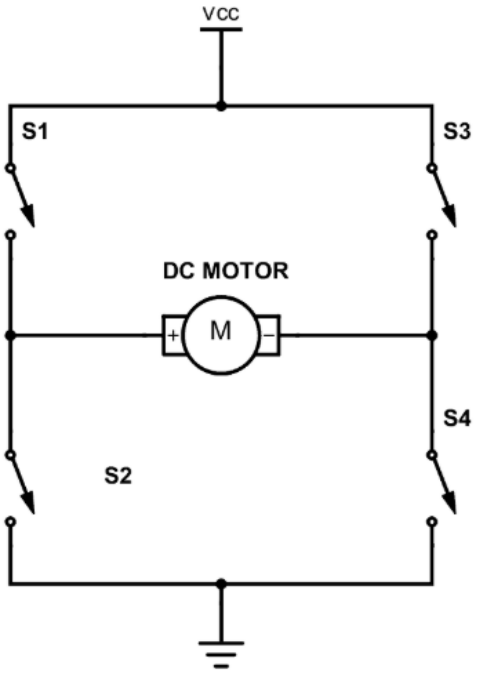


- Understand the code:
  - Especially on how to define the pins used correctly

DC MOTOR

DC Motor: **Speed** and **Direction**





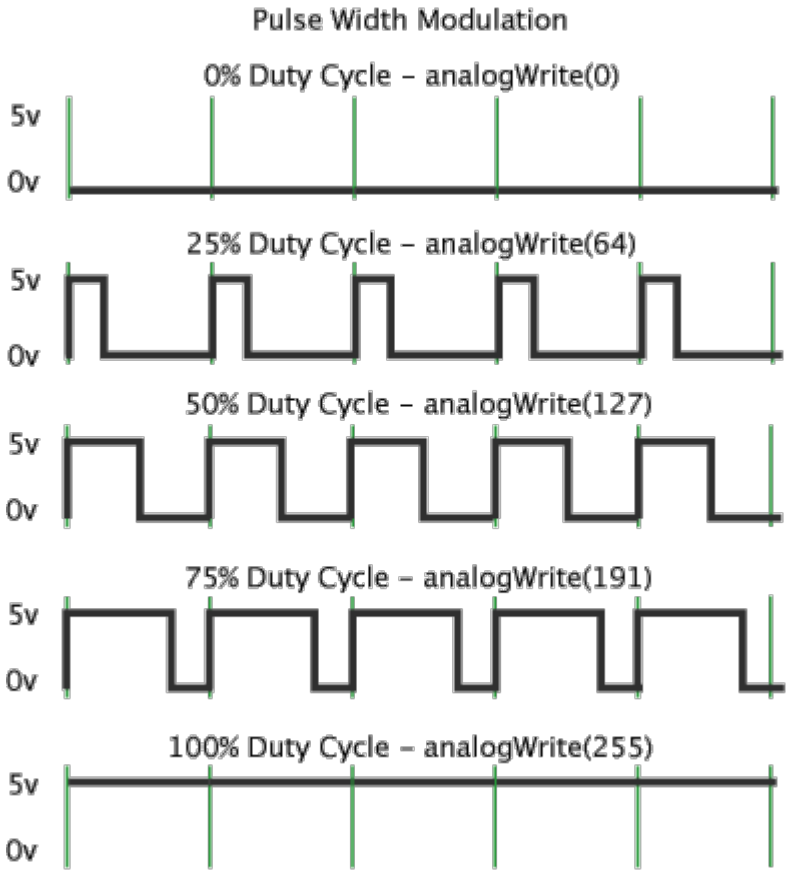
1. Turning speed is control by **PWM**

2. Turning direction is handled by **H-Bridge**

Controlling the Speed of Motor

- Intuitively, we can control the speed of motor by varying the voltage supply
- So, if 5v == maximum speed, supplying 2v should give us a slower speed…
- The idea is correct, but there is a problem…

Pulse Width Modulation (PWM)



- Sometimes we need to have **analog output**:
  - e.g. varying amount of power to the motor will modify the speed
- PWM enables us to simulate analog output by switching on/off rapidly on an digital output channel
- Pin 2 to 13, 44 to 46 supports PWM

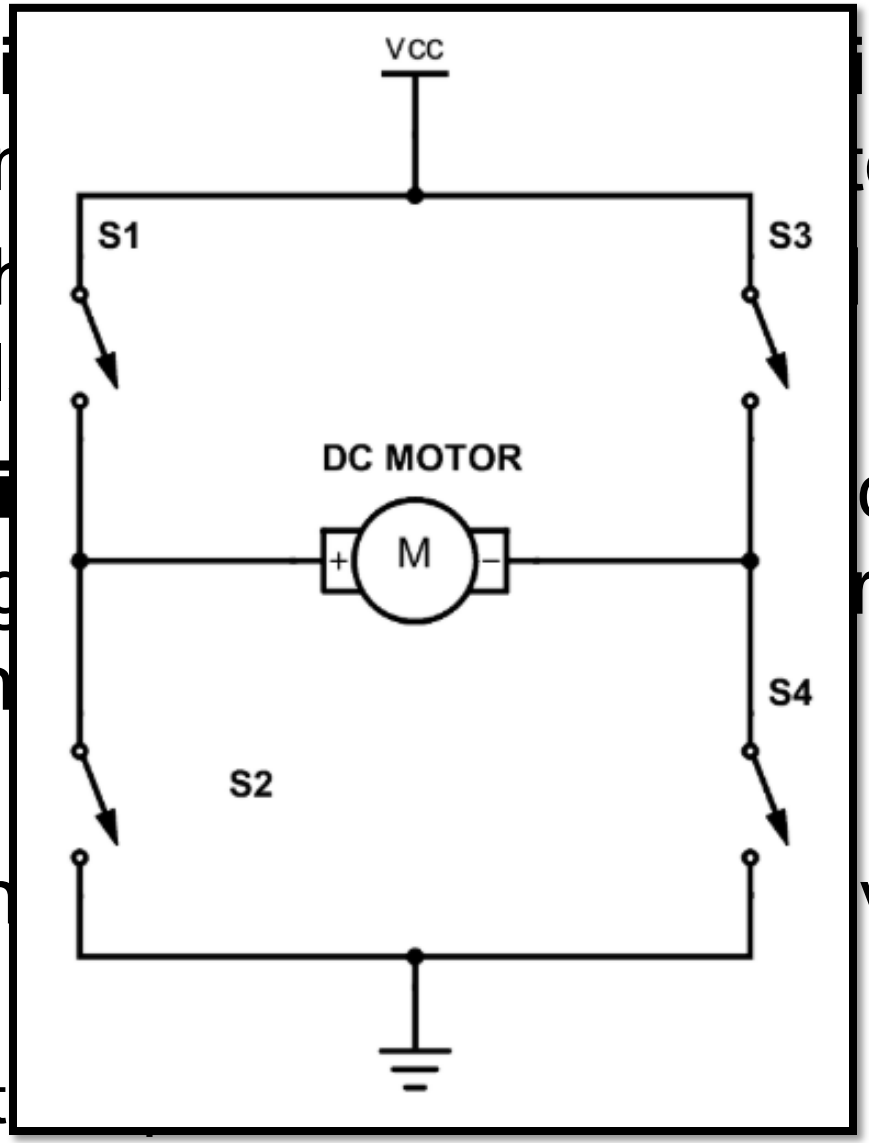
Exercise: **PWM for LED**

- Let's try to use PWM to control the brightness of the LED
  - Conveniently, Pin 13 supports PWM!
  - See code **`Blink_PWM.ino`**

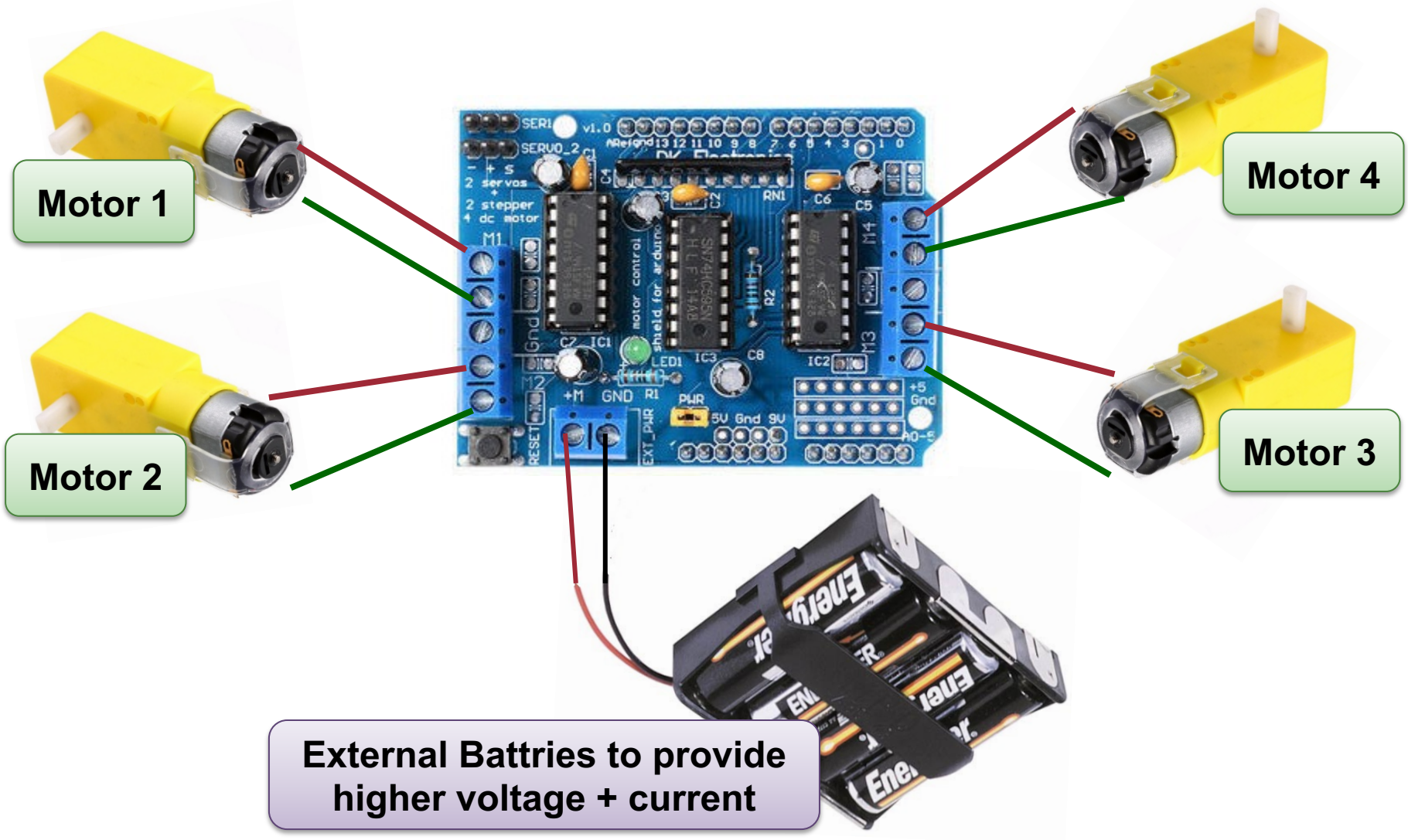
- [**Additional Challenge**] Write a program to control the brightness of LED using Serial Communication

Controlling the Direction of Motor

- H-Bridge i… ic componer… tor etc)
- Imagine th… to control 2-4 wheel…
- **Motor Dri…** commonly seen integ… rovide H-bridge fun…
- We use th… ver Arduino Shield:
  - Can cont…



Dual L293D Motor Shield



Motor Shield: Library

- Basic functionality is provided by the "Adafruit Basic Motor Shield"

```
#include <AFMotor.h>

void setup() {
  // put your setup code here, to run once:
  AF_DCMotor motor(4);  // The motor number, i.e. 1, 2, 3 or 4

  motor.setSpeed( 200 );

  motor.run( FORWARD );
  delay(1000);

  motor.run( BACKWARD );
  delay(1000);

  motor.run( RELEASE );
  delay(1000);

}
```

END