Build an Arduino Self-Driving Car

Lesson 1: Controlling a Motor

Estimated time: 90 minutes

Introduction

In this lesson, students will learn to control a motor using an H-bridge, also called a motor driver. Students will learn how to build the circuit to connect the H-bridge, motor, and external battery pack to power the motor. They will write code to change the motor's direction of rotation and will learn how to use functions to make it easier to reuse sections of code.

Learning Objectives

- Build a circuit with an H-bridge
- Write code to change a motor's direction of rotation
- Write custom functions to re-use sections of code

Preparation

- Try this activity yourself before you try it with students. You can keep your working circuit as a demonstration to show students.
- You may wish to print out some of the appendices as a reference for the students.

Materials

Each group will need:

- Computer with USB-A port and Arduino IDE installed
- Arduino UNO
- USB A-B cable
- Breadboard
- L293D H-bridge
- DC motor
- 4xAA battery holder
- AA batteries (4)
- Jumper wires (assorted)

Procedure

Walk your students through the following steps or have them watch the video and follow along.

Step	Timestamp	Description
1	0:00	Demonstrate the working circuit for students.
2	0:52	Explain why the H-bridge is required to control the motor with the Arduino and why you cannot just plug the motor directly into the Arduino's digital pins like you can with an LED.
3	2:00	Assemble the circuit on the breadboard (see Appendix 1A). **Note:** this is a complex circuit. It is important for students to take their time and double-check all wiring before connecting the battery pack to the circuit as the final step. Watch out for accidental short circuits. Immediately disconnect the batteries if any circuits feel hot or you see/smell smoke.
4	11:10	Enter the code to make the motor alternate between spinning and stopping (Appendix 1A). Upload the code and confirm that it works. **Note:** students might claim that something is "broken" if their circuit doesn't work on the first try. Refer to the troubleshooting video on the Science Buddies "How to Use an Arduino" page for help. You can also confirm that the motor works by connecting it directly to the battery pack.
5	12:06	Reverse the two motor wires and observe how this makes the motor spin in the opposite direction.
6	13:20	Modify the circuit to connect the second input pin on the H-bridge to an Arduino pin (see Appendix 1B).
7	14:58	Change the code to use both control pins (see Appendix 1B).

8	15:29	Add code to the program so the motor spins backward for one second after stopping for one second (see Appendix 1C for example solution, but let students try this on their own first).
9	16:39	Change the code so the commands to stop the motor are inside a function (see Appendix 1D). Use the function to make the motor stop again after spinning backward for one second.
10	19:53	Change the code so the commands to spin the motor forward and backward are also in their own functions (see Appendix 1E for example solution, but let students try this on their own first).
11	20:56	Change the code so the delay time is an argument passed to the **`stopMotor`** function (see Appendix 1F).
12	22:40	Change the code so the **`spinForward`** and **`spinBackward`** functions also accept a delay time as an argument (see Appendix 1G for example solution, but let students try this on their own first).

Assessment

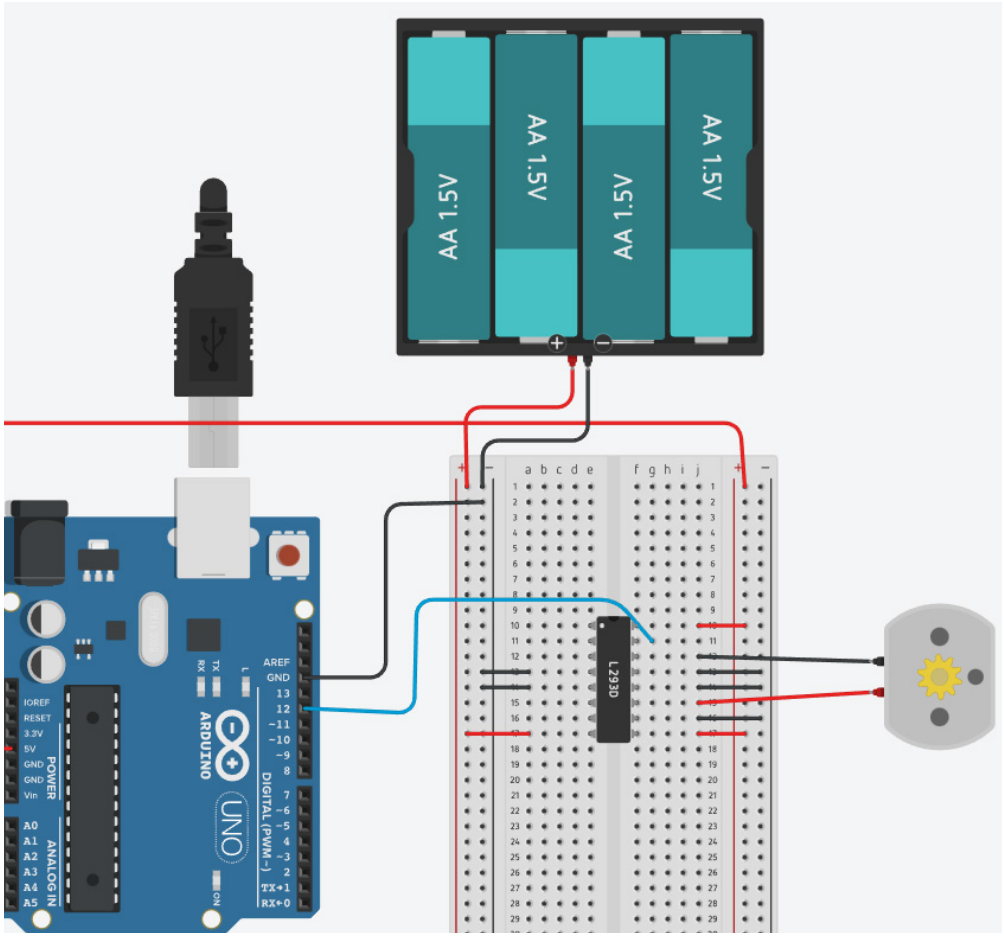
As you walk around the class:

- Confirm that students can properly follow the breadboard diagram to build the circuit and make the motor spin using the provided example code.
- Observe whether students can modify the code to make the motor spin backwards.
- Observe whether students can properly use functions in their code, including functions with arguments.

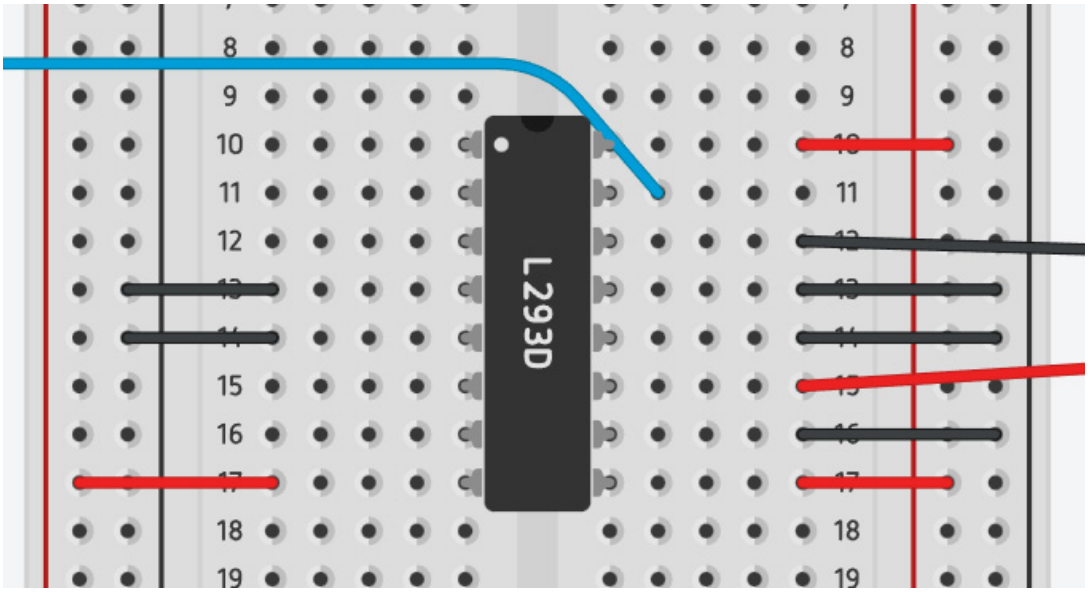
Appendix 1A: Breadboard Diagram and Code for Single-Direction Spinning

The diagram below shows the entire circuit including the Arduino, battery pack, and motor.

Important! Note that some breadboards may have the power (usually marked with a "+" and/or a red line) and ground (usually marked with a "-" and/or black or blue line) buses switched relative to what is shown in this diagram. Your ground wires (the black wires in this diagram) should always go in the ground buses, regardless of whether they are on the left or the right. Base your wiring on *your breadboard*.



This diagram shows a close-up of the H-bridge on the breadboard:



The H-bridge pins are connected as follows. Pins are numbered *counterclockwise starting from the top left*.

H-bridge pin	Connection
1	Not connected
2	Not connected
3	Not connected
4	Ground bus
5	Ground bus
6	Not connected
7	Not connected
8	+6V bus (from 4xAA battery pack)
9	+5V bus (from Arduino)
10	Ground bus
11	Motor wire
12	Ground bus
13	Ground bus
14	Other motor wire
15	Arduino pin 12
16	+5V bus (from Arduino)

The following code will make the motor alternate between spinning for one second and stopping for one second:

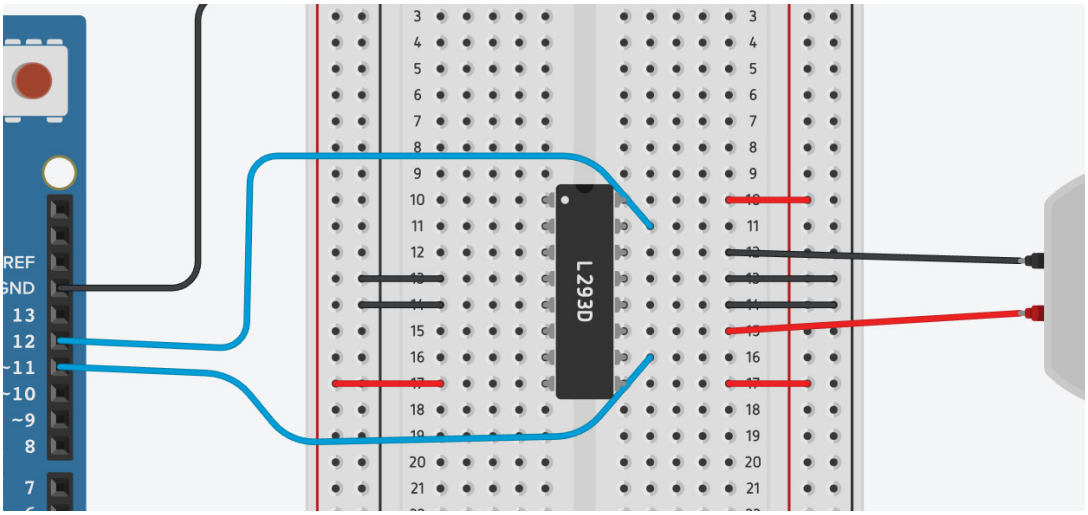
```
int motorPin = 12; // declare variable for motor pin

void setup() {  // code that only runs once
 pinMode(motorPin,OUTPUT);  // set motor pin as output
}

void loop() { // code that loops forever
 digitalWrite(motorPin,HIGH);  // turn motor on
 delay(1000);  // Wait for 1000 milliseconds
 digitalWrite(motorPin,LOW);  // turn motor off
 delay(1000);  // Wait for 1000 milliseconds
}
```

Appendix 1B: Breadboard Diagram and Code for Bi-Directional Steering

Remove the jumper wire connecting H-bridge pin 9 to the ground bus. Instead, connect it to Arduino pin 11 as shown in this diagram.



The following code will make the motor spin for one second then stop for one second. This time, the code uses two pins to control the motor.

```
int backwardPin = 12; // declare variable for motor backward pin
int forwardPin = 11; // declare variable for motor forward pin

void setup() {  // code that only runs once
  pinMode(forwardPin,OUTPUT);   // set forward pin as output
  pinMode(backwardPin,OUTPUT);  // set backward pin as output
}

void loop() { // code that loops forever
  // spin forward
  digitalWrite(forwardPin,HIGH);  // set forward pin high
  digitalWrite(backwardPin,LOW);  // set backward pin low
  delay(1000);  // Wait for 1000 milliseconds

  // stop
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,LOW);  // set backward pin low
  delay(1000);  // Wait for 1000 milliseconds
}
```

Appendix 1C: Code to Spin Forward, Stop, Then Spin Backward

The following code will make the motor spin forward for one second, stop for one second, then spin backward for one second before repeating.

```
int backwardPin = 12; // declare variable for motor backward pin
int forwardPin = 11; // declare variable for motor forward pin

void setup() {  // code that only runs once
  pinMode(forwardPin,OUTPUT);   // set forward pin as output
  pinMode(backwardPin,OUTPUT);  // set backward pin as output
}

void loop() { // code that loops forever
  // spin forward
  digitalWrite(forwardPin,HIGH);  // set forward pin high
  digitalWrite(backwardPin,LOW);  // set backward pin low
  delay(1000);  // Wait for 1000 milliseconds

  // stop
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,LOW);  // set backward pin low
  delay(1000);  // Wait for 1000 milliseconds

  // spin backward
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,HIGH);  // set backward pin high
  delay(1000);  // Wait for 1000 milliseconds
}
```

Appendix 1D: Stop Motor Function

The following code has the commands to stop the motor inside a function called **`stopMotor`**. This function can be called repeatedly from inside the **`loop`** function, without needing to copy and paste the **`digitalWrite`** commands.

```
int backwardPin = 12; // declare variable for motor backward pin
int forwardPin = 11; // declare variable for motor forward pin

void setup() {  // code that only runs once
  pinMode(forwardPin,OUTPUT);   // set forward pin as output
  pinMode(backwardPin,OUTPUT);  // set backward pin as output
}

void loop() { // code that loops forever
  // spin forward
  digitalWrite(forwardPin,HIGH);  // set forward pin high
  digitalWrite(backwardPin,LOW);  // set backward pin low
  delay(1000);  // Wait for 1000 milliseconds

  // stop
  stopMotor();
  delay(1000);  // Wait for 1000 milliseconds

  // spin backward
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,HIGH);  // set backward pin high
  delay(1000);  // Wait for 1000 milliseconds
}

void stopMotor(){  // function to stop motor
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,LOW);  // set backward pin low
}
```

Appendix 1E: Stop, Forward, and Backward Functions

The following code has commands to spin the motor forward, backward, and stop all in their own functions. These functions can be called repeatedly from inside the **loop** function. This makes the **loop** function much neater.

```
int backwardPin = 12; // declare variable for motor backward pin
int forwardPin = 11; // declare variable for motor forward pin

void setup() {  // code that only runs once
  pinMode(forwardPin,OUTPUT);   // set forward pin as output
  pinMode(backwardPin,OUTPUT);  // set backward pin as output
}

void loop() { // code that loops forever

  spinForward();  // call the spinForward function
  delay(1000);  // Wait for 1000 milliseconds

  stopMotor();  // call the stopMotor function
  delay(1000);  // Wait for 1000 milliseconds

  spinBackward();  // call the spinBackward function
  delay(1000);  // Wait for 1000 milliseconds

  stopMotor();  // call the stopMotor function
  delay(1000);  // Wait for 1000 milliseconds
}

void stopMotor(){  // function to stop motor
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,LOW);  // set backward pin low
}
```

```
void spinForward(){  // function to spin forward
  digitalWrite(forwardPin,HIGH);  // set forward pin high
  digitalWrite(backwardPin,LOW);  // set backward pin low
}

void spinBackward(){  // function to spin backward
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,HIGH);  // set backward pin high
}
```

Appendix 1F: Making the Delay Time a Function Input

The following code uses the delay time as an argument (input) to the **stopMotor** function. This eliminates the need for a separate **delay** command in the **loop** function after **stopMotor**.

```
int backwardPin = 12; // declare variable for motor backward pin
int forwardPin = 11; // declare variable for motor forward pin

void setup() {  // code that only runs once
  pinMode(forwardPin,OUTPUT);   // set forward pin as output
  pinMode(backwardPin,OUTPUT);  // set backward pin as output
}

void loop() { // code that loops forever

  spinForward();  // call the spinForward function
  delay(1000);  // Wait for 1000 milliseconds

  stopMotor(1000);  // call the stopMotor function

  spinBackward();  // call the spinBackward function
  delay(1000);  // Wait for 1000 milliseconds

  stopMotor(2000);  // call the stopMotor function
}

void stopMotor(int delay_time){  // function to stop motor
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,LOW); // set backward pin low
  delay(delay_time);             // wait for delay_time ms
}

void spinForward(){  // function to spin forward
  digitalWrite(forwardPin,HIGH);  // set forward pin high
  digitalWrite(backwardPin,LOW);  // set backward pin low
}
```

```
void spinBackward(){  // function to spin backward
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,HIGH);  // set backward pin high
}
```

Appendix 1G: Delay Times in All Functions

The following code uses delay times as arguments for the **stopMotor**, **spinForward**, and **spinBackward** functions. This completely eliminates the need for separate **delay** commands in the **loop** function. Now you can easily change the motor's behavior with a single line of code.

```
int backwardPin = 12; // declare variable for motor backward pin
int forwardPin = 11; // declare variable for motor forward pin

void setup() {  // code that only runs once
  pinMode(forwardPin,OUTPUT);   // set forward pin as output
  pinMode(backwardPin,OUTPUT);  // set backward pin as output
}

void loop() { // code that loops forever

  spinForward(1000);   // call the spinForward function
  stopMotor(1000);     // call the stopMotor function
  spinBackward(2000);  // call the spinBackward function
  stopMotor(2000);  // call the stopMotor function

}

void stopMotor(int delay_time){  // function to stop motor
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,LOW); // set backward pin low
  delay(delay_time);             // wait for delay_time ms
}

void spinForward(int delay_time){  // function to spin forward
  digitalWrite(forwardPin,HIGH);  // set forward pin high
  digitalWrite(backwardPin,LOW);  // set backward pin low
  delay(delay_time);             // wait for delay_time ms
}
```

```
void spinBackward(int delay_time){  // function to spin backward
  digitalWrite(forwardPin,LOW);  // set forward pin low
  digitalWrite(backwardPin,HIGH);  // set backward pin high
  delay(delay_time);              // wait for delay_time ms
}
```