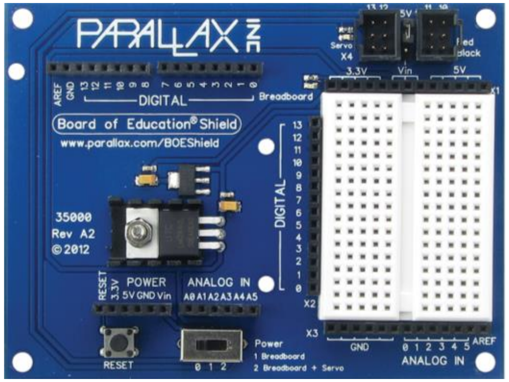
Arduino BOE kit and Raspibot Board

Class 2: Basics of Arduino Coding



August 8, 2017
Don Wilcher

DesignNews CEC CONTINUING EDUCATION CENTER

Presented by:

Digi-Key ELECTRONICS ROHDE&SCHWARZ

Class 2: Basics of Arduino Coding

Agenda

- Servo Motor Basics
- Attaching Servo Motors to the Parallax Arduino BOE Shield
- Coding with Loops
- Hands-On Labs: Coding Examples

Servo Motor Basics

What is a Servo Motor?

a) A closed loop servo mechanism

b) An electromechanical component that uses position feedback to control:

i. its motion

ii. Final position

c) input Digital or Analog control signals are use to operate the output shaft appropriately.

Servo Motor Basics…

What is a Servo Motor?

d) Can be control more precisely than DC motors

e) They have three wires

  i. Vsupply

  ii. Gnd

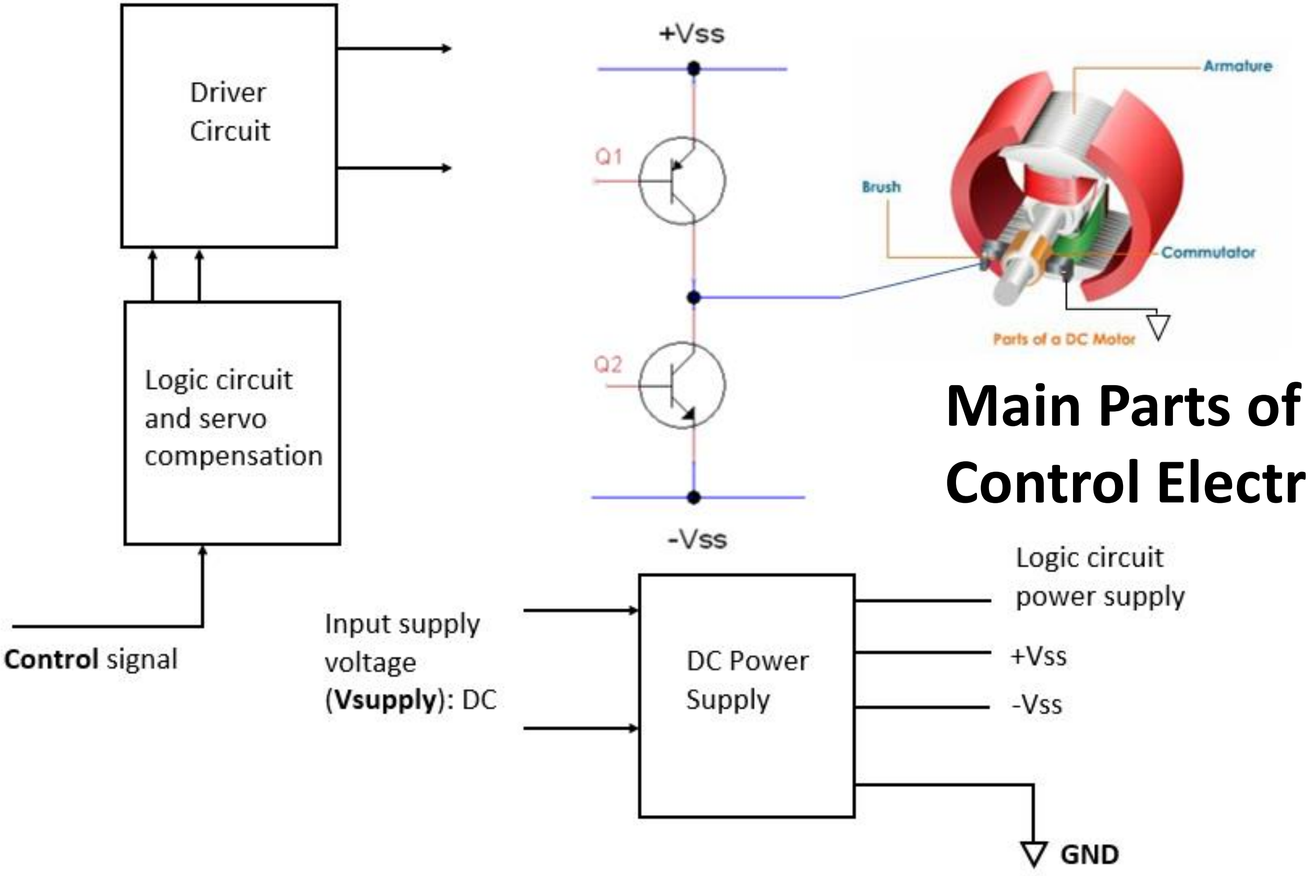
  iii. Control

Question 1

What three attributes that describes a typical servo motor?

Servo Motor Basics…

What is a Servo Motor?



Driver Circuit

Logic circuit and servo compensation

Control signal

+Vss

Q1

Q2

-Vss

Input supply voltage (**Vsupply**): DC

DC Power Supply

Logic circuit power supply

+Vss

-Vss

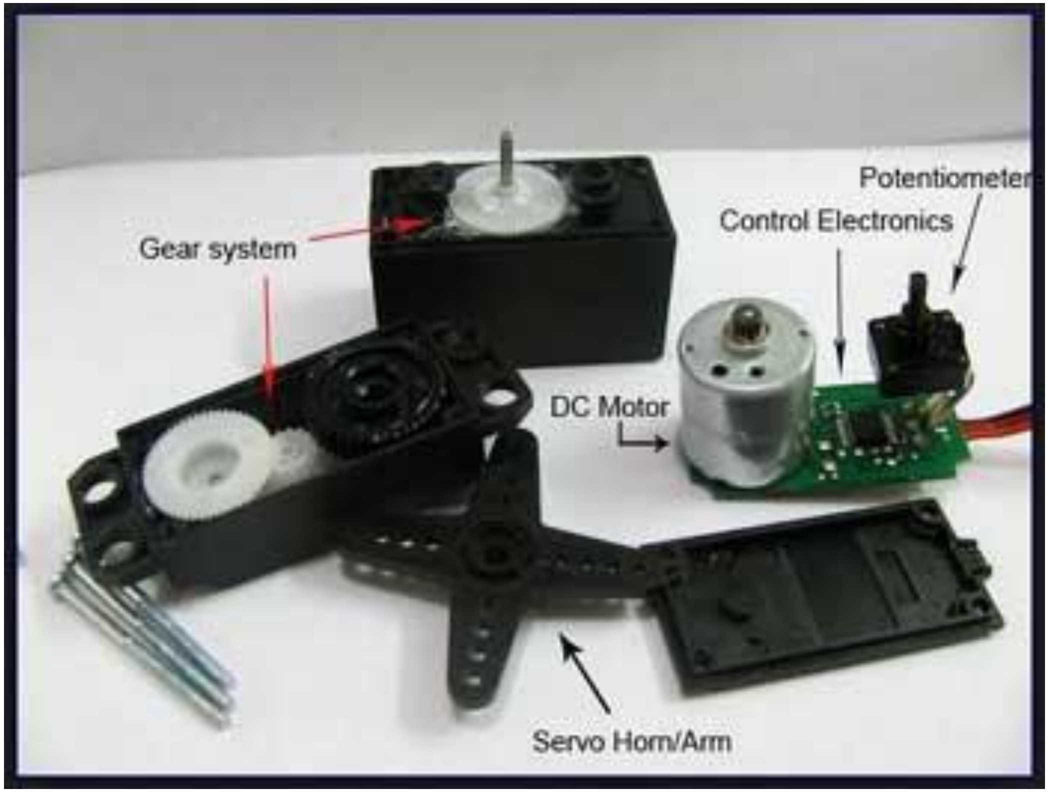
GND

Main Parts of Servo Motor: Control Electronics

Servo Motor Basics…

What is a Servo Motor?

Main Parts of Servo Motor: Control Electronics and Mechanicals



Source:

https://www.electrical4u.com/what-is-servo-motor/

Question 2

What component is not part of a servo motor?

a) Gear System

b) Rheostat

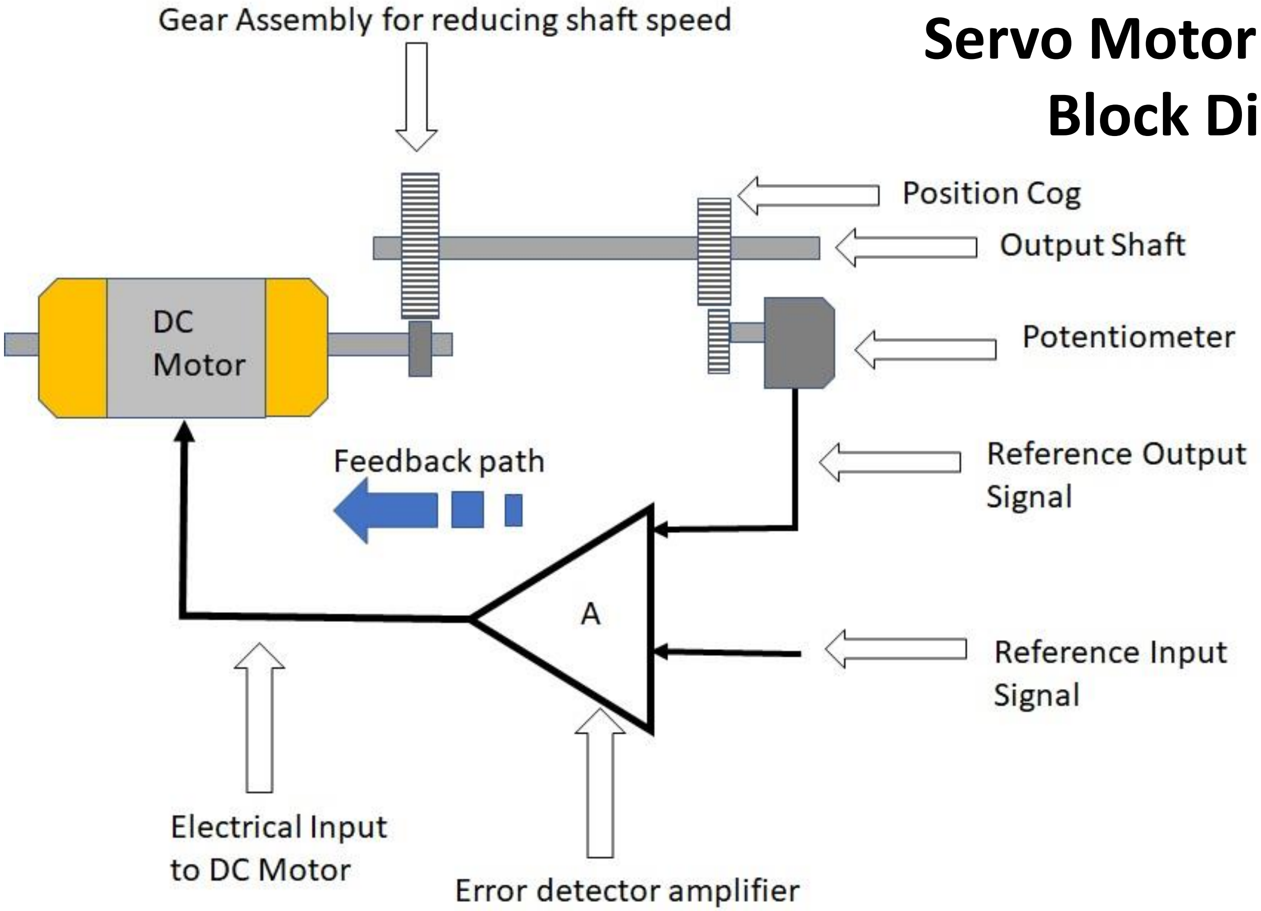
c) DC motor

d) all parts listed are correct

Servo Motor Basics…

What is a Servo Motor?

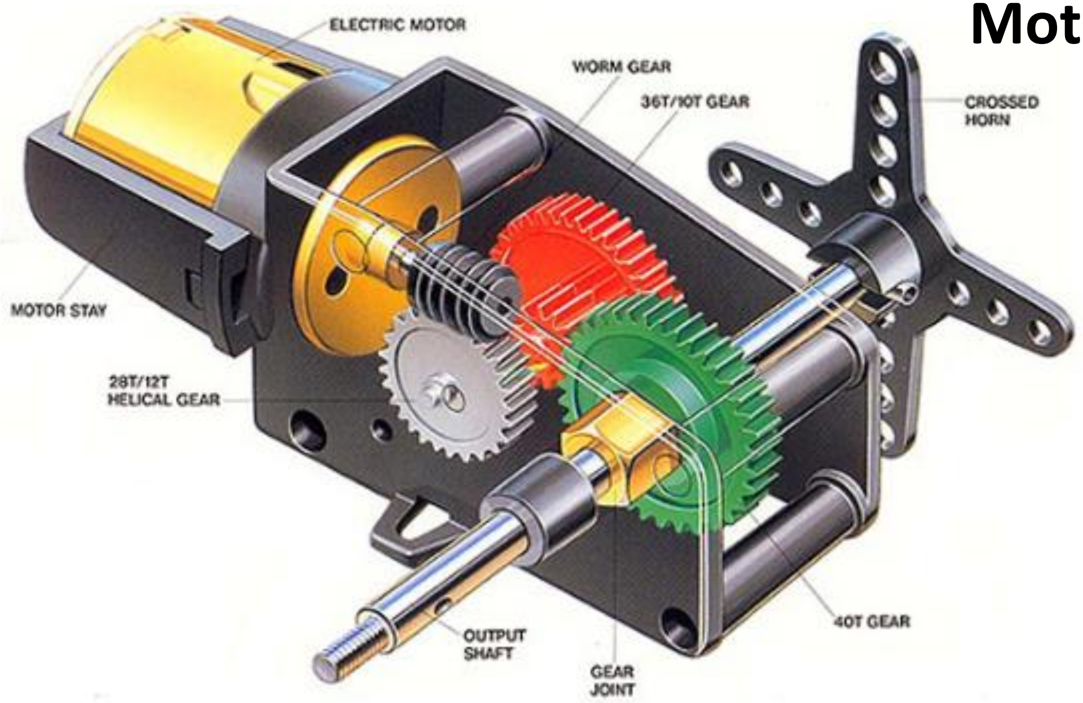
Servo Motor Functional Block Diagram



Presented by:

Servo Motor Basics…

What is a Servo Motor?



Main Parts of a Servo Motor: Mechanical Parts

Source:
https://www.elprocus.com/difference-dc-motor-servo-motor-stepper-motor/

Presented by:

Question 3

In slide 9, what signal is produced by the potentiometer?

a) Feedback signal

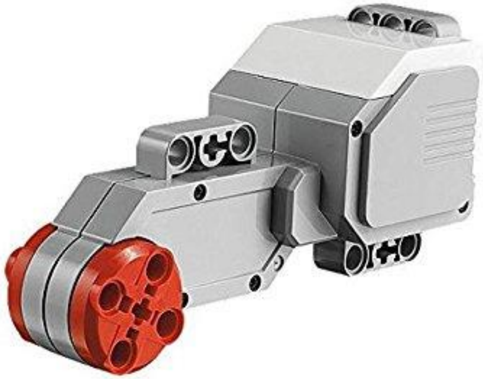
b) Reference Input signal

c) Reference Output signal

d) None of the above

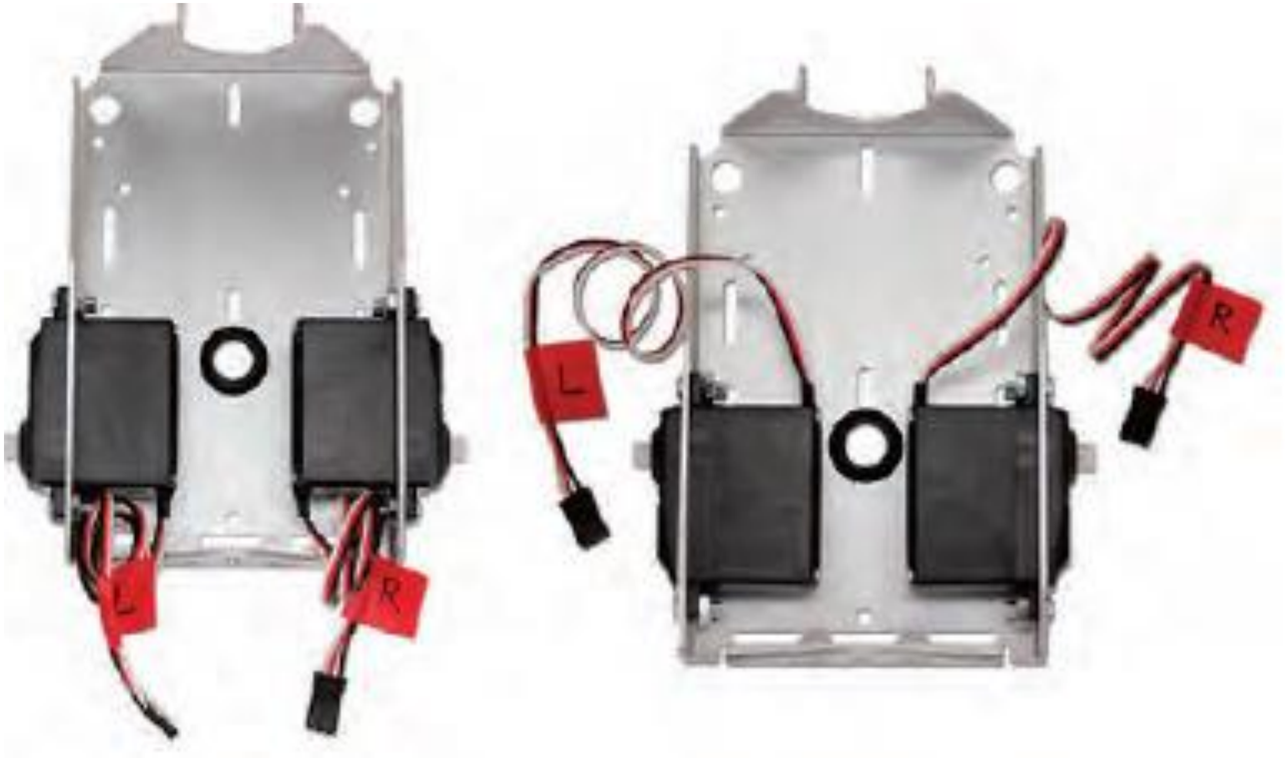
Servo Motor Basics…

Variety of Servo Motors



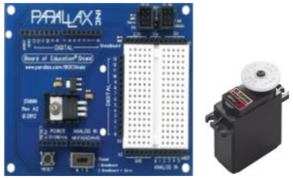
Attaching Servo Motors to the Parallax BOE Shield

Attaching Servo Motors to Chassis

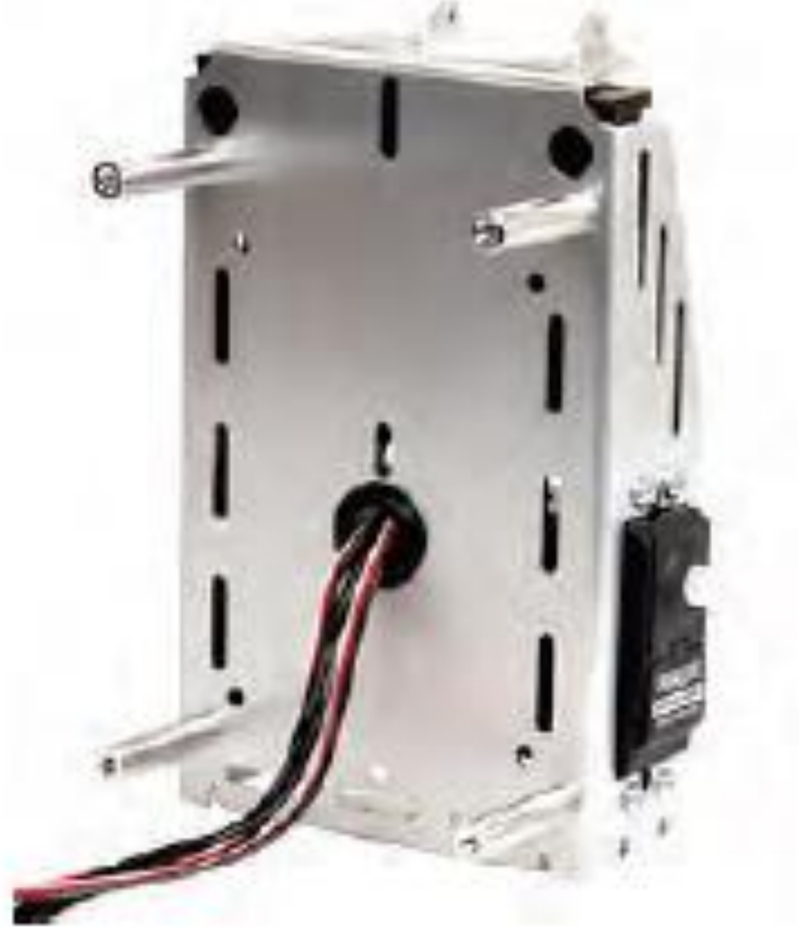


Presented by:

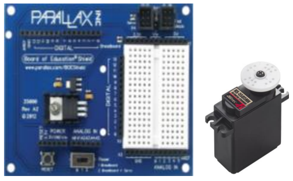
Attaching Servo Motors to the Parallax BOE Shield



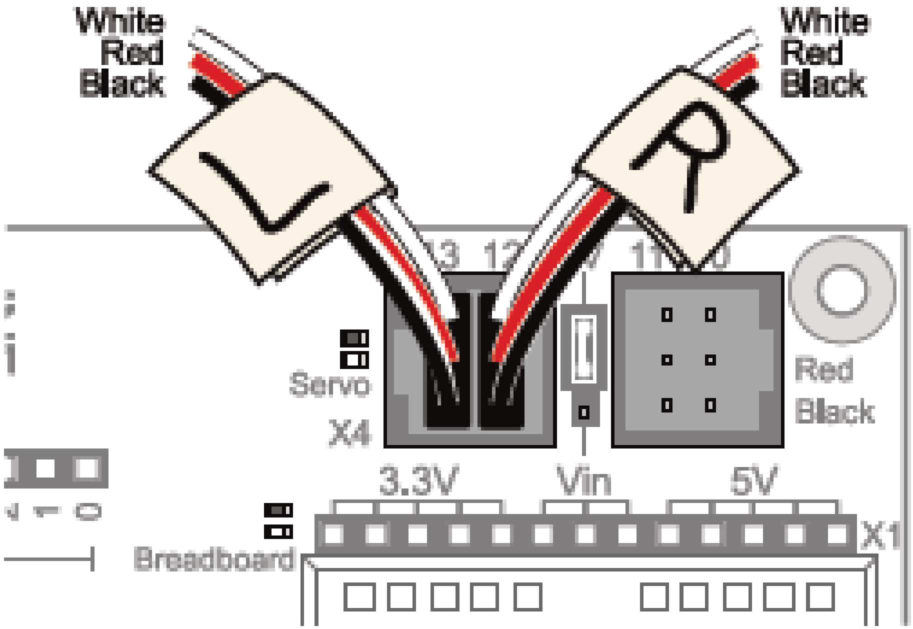
Passing Servo Motor wires through Chassis Grommet



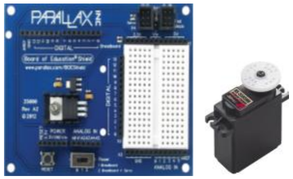
Attaching Servo Motors to the Parallax BOE Shield



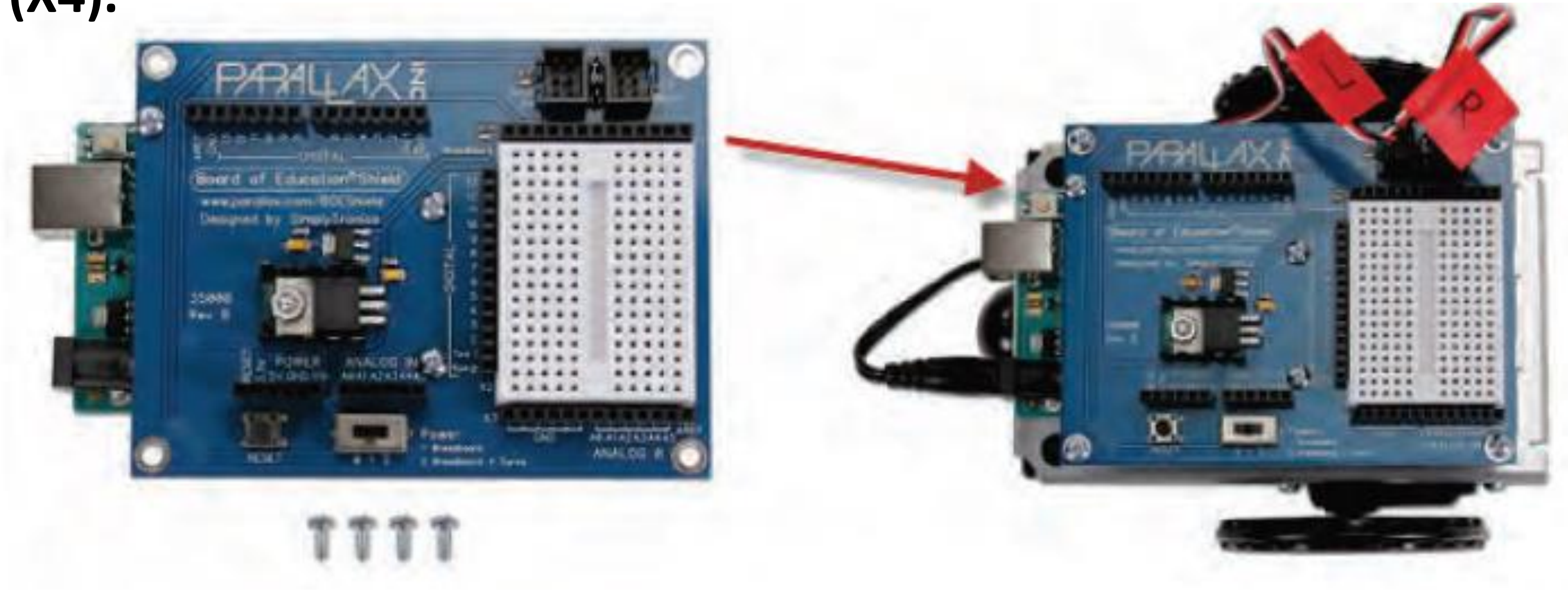
Identifying Servo Motor wires.



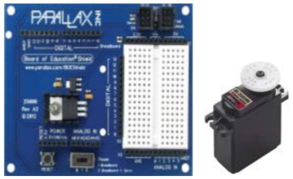
Attaching Servo Motors to the Parallax BOE Shield



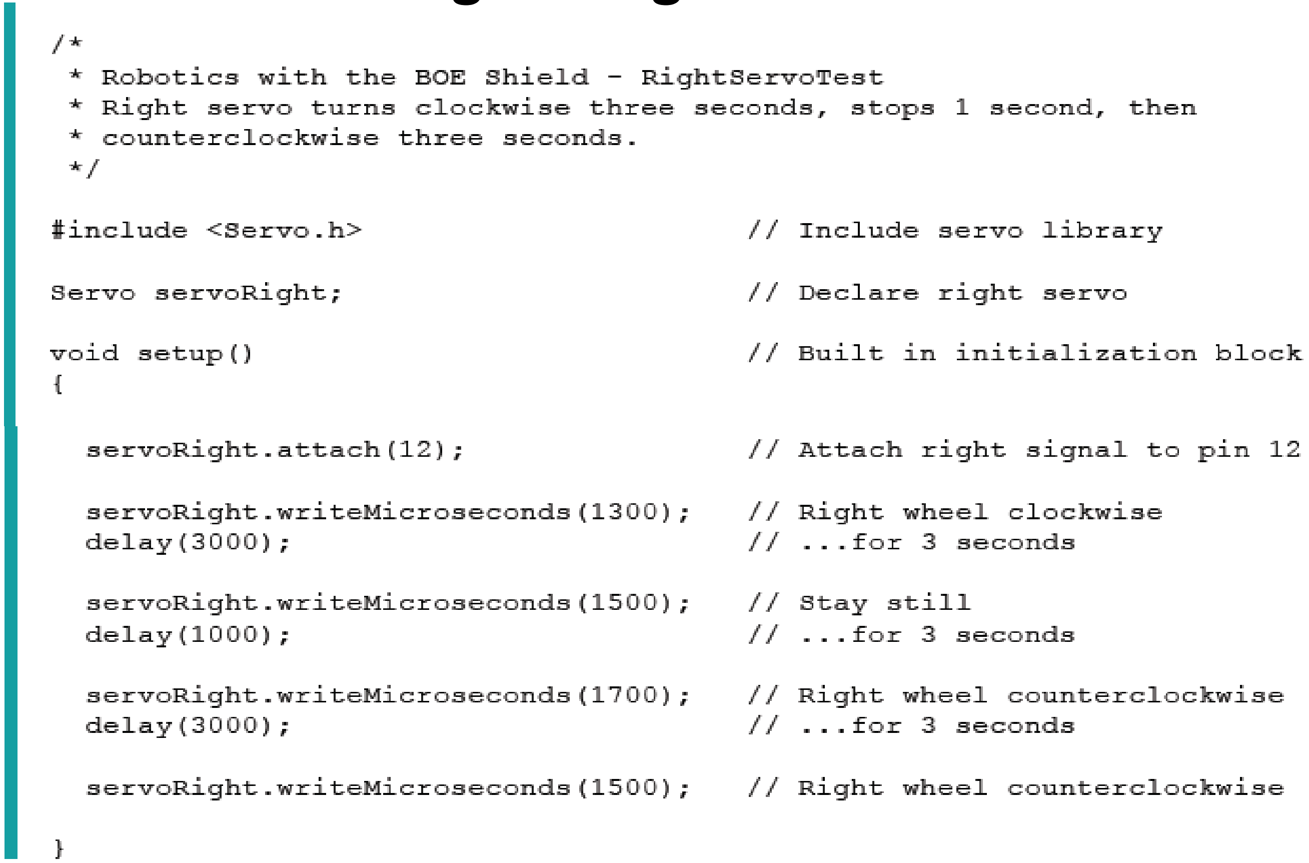
Connecting Servo Motor wires (L&R) to onboard 2x3 male connector (X4).



Attaching Servo Motors to the Parallax BOE Shield



Testing the Right Servo Motor

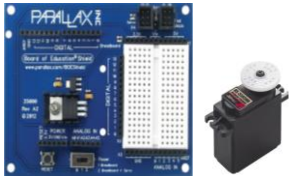


```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{

  servoRight.attach(12);                    // Attach right signal to pin 12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}
```

Question 4

What Arduino BOE Shield onboard connector is used to attach the servo motor wires?

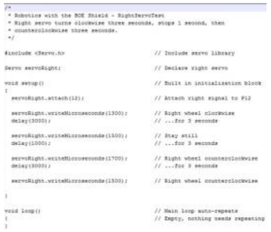
Attaching Servo Motors to the Parallax BOE Shield



Design Challenge:

Modify the code in the previous slide to test the left servo motor.

Coding with Loops



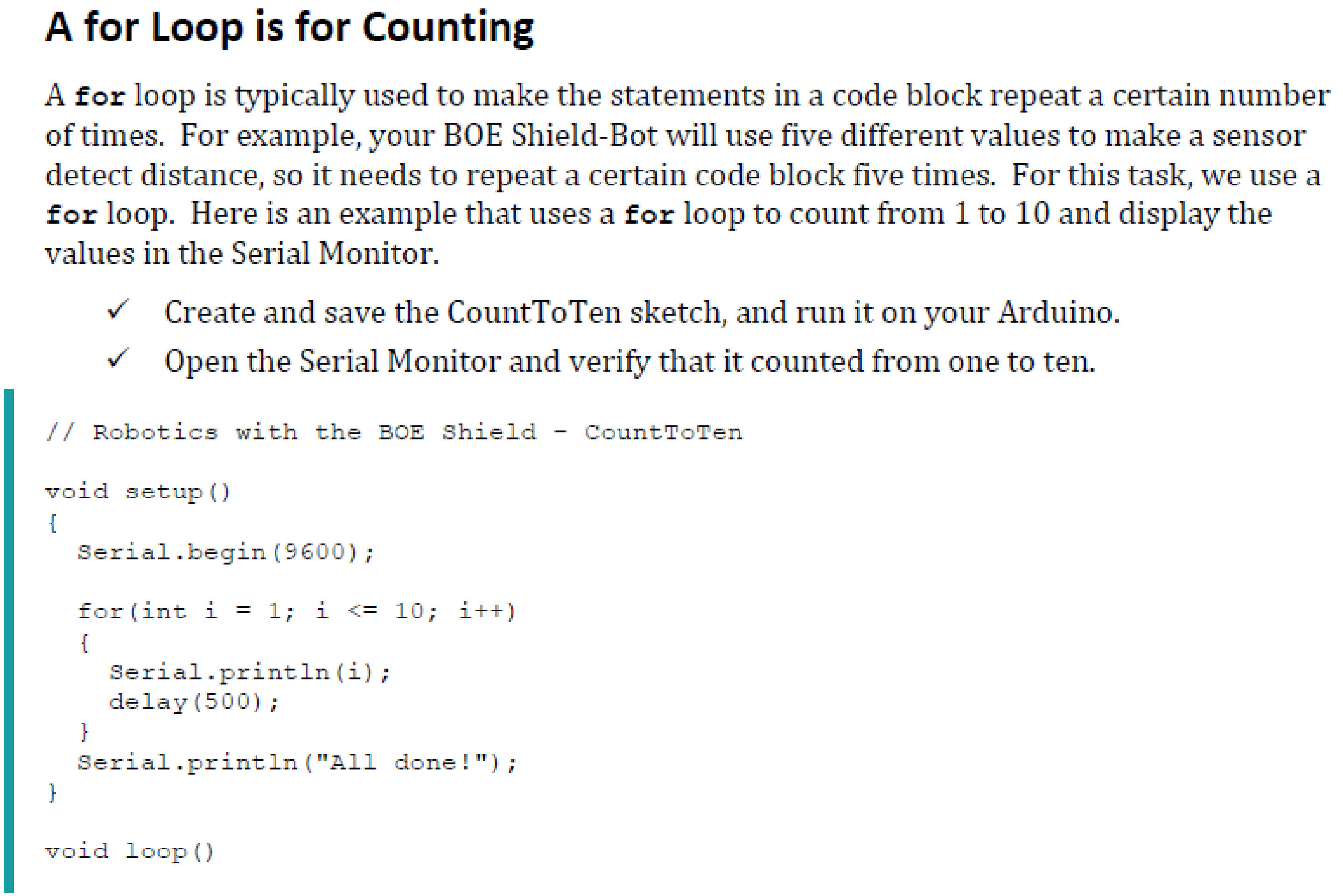
```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                           // Include servo library

Servo servoRight;                            // Declare right servo

void setup()                                 // Built in initialization block
{
  servoRight.attach(12);                     // Attach right signal to P12

  servoRight.writeMicroseconds(1300);        // Right wheel clockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Stay still
  delay(1000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1700);        // Right wheel counterclockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Right wheel counterclockwise

}

void loop()                                  // Main loop auto-repeats
{                                            // Empty, nothing needs repeating
}
```

Note: We'll be using examples from Robotics with Board of Education Shield for Arduino Book by Andy Lindsay, version 1.0

Source: http://www.robotshop.com/blog/en/how-to-make-a-robot-lesson-4-understanding-microcontrollers-2-3700

Coding with Loops

```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                              // Include servo library

Servo servoRight;                               // Declare right servo

void setup()                                    // Built in initialization block
{
  servoRight.attach(12);                        // Attach right signal to P12

  servoRight.writeMicroseconds(1300);           // Right wheel clockwise
  delay(3000);                                  // ...for 3 seconds

  servoRight.writeMicroseconds(1500);           // Stay still
  delay(1000);                                  // ...for 3 seconds

  servoRight.writeMicroseconds(1700);           // Right wheel counterclockwise
  delay(3000);                                  // ...for 3 seconds

  servoRight.writeMicroseconds(1500);           // Right wheel counterclockwise

}

void loop()                                     // Main loop auto-repeats
{                                               // Empty, nothing needs repeating
}
```



A for Loop is for Counting

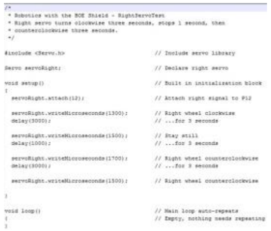
A **for** loop is typically used to make the statements in a code block repeat a certain number of times. For example, your BOE Shield-Bot will use five different values to make a sensor detect distance, so it needs to repeat a certain code block five times. For this task, we use a **for** loop. Here is an example that uses a **for** loop to count from 1 to 10 and display the values in the Serial Monitor.

- ✓ Create and save the CountToTen sketch, and run it on your Arduino.
- ✓ Open the Serial Monitor and verify that it counted from one to ten.

```
// Robotics with the BOE Shield - CountToTen

void setup()
{
  Serial.begin(9600);

  for(int i = 1; i <= 10; i++)
  {
    Serial.println(i);
    delay(500);
  }
  Serial.println("All done!");
}

void loop()
```

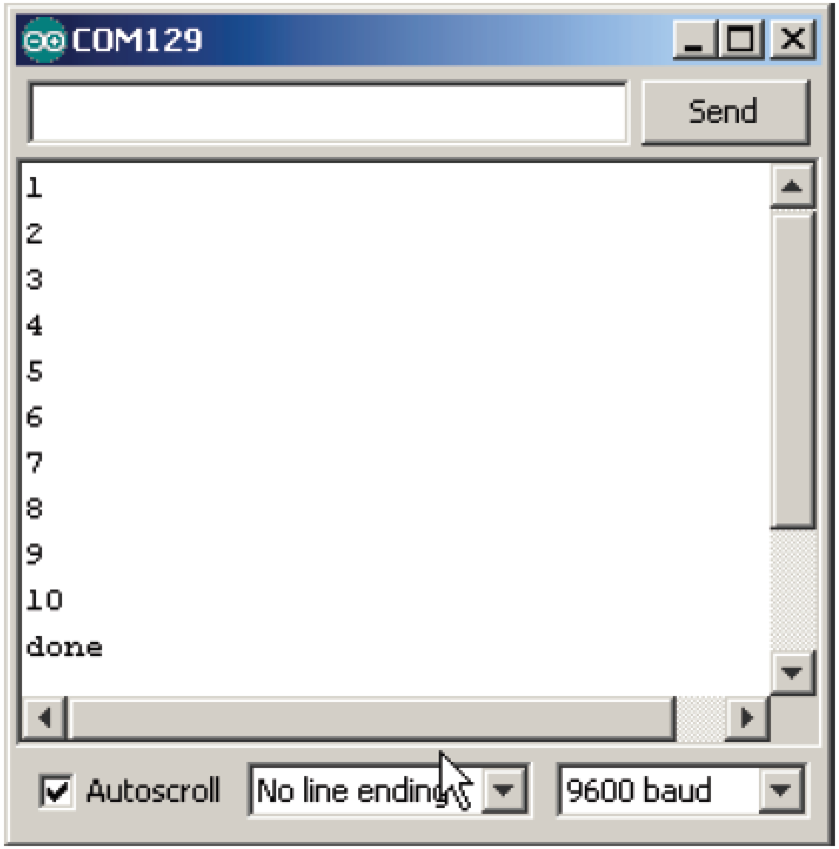
Presented by:

Coding with Loops…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                         // Include servo library

Servo servoRight;                          // Declare right servo

void setup()                               // Built in initialization block
{
  servoRight.attach(12);                   // Attach right signal to P12

  servoRight.writeMicroseconds(1300);      // Right wheel clockwise
  delay(3000);                             // ...for 3 seconds

  servoRight.writeMicroseconds(1500);      // Stay still
  delay(1000);                             // ...for 3 seconds

  servoRight.writeMicroseconds(1700);      // Right wheel counterclockwise
  delay(3000);                             // ...for 3 seconds

  servoRight.writeMicroseconds(1500);      // Right wheel counterclockwise

}

void loop()                                // Main loop auto-repeats
{                                          // Empty, nothing needs repeating
}
```

Output of “for” loop



```
1
2
3
4
5
6
7
8
9
10
done
```

☑ Autoscroll

No line ending

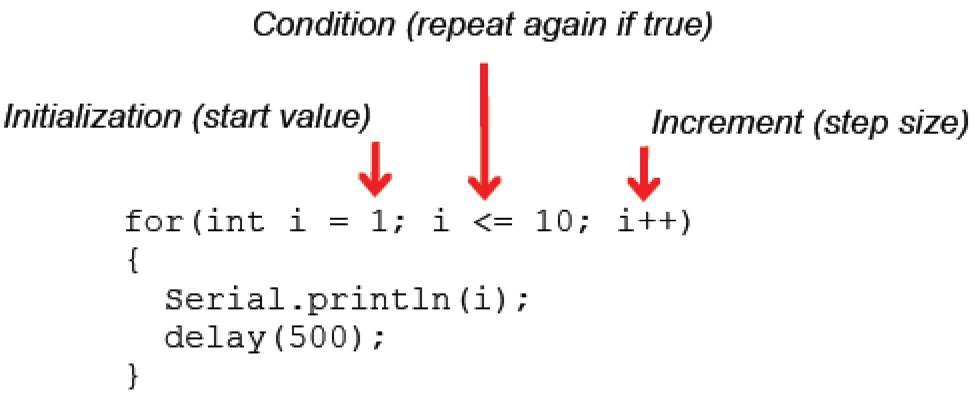
9600 baud

Coding with Loops…

```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

How the for Loop Works

The figure below shows the **for** loop from the last example sketch, CountTenTimes. It labels the three elements in the **for** loop's parentheses that control how it counts.



Condition (repeat again if true)

Initialization (start value)

Increment (step size)

```
for(int i = 1; i <= 10; i++)
{
  Serial.println(i);
  delay(500);
}
```

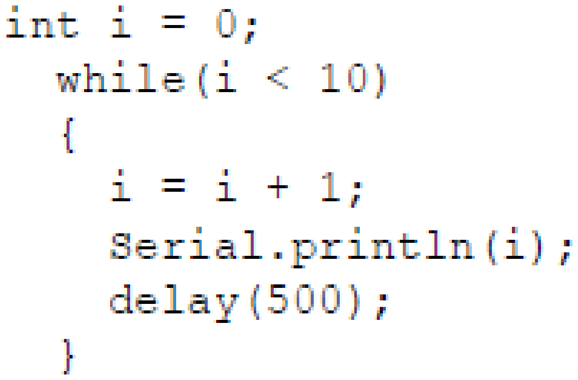
Initialization: the starting value for counting. It's common to declare a local variable for the job as we did here with **int i = 1**; naming it **i** for 'index.'

Condition: what the **for** loop checks between each repetition to make sure the condition is still true. If it's true, the loop repeats again. If not, it allows the code to move on to the next

Coding with Loops…

```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                             // Include servo library

Servo servoRight;                              // Declare right servo

void setup()                                   // Built in initialization block
{
  servoRight.attach(12);                       // Attach right signal to P12

  servoRight.writeMicroseconds(1300);          // Right wheel clockwise
  delay(3000);                                 // ...for 3 seconds

  servoRight.writeMicroseconds(1500);          // Stay still
  delay(1000);                                 // ...for 3 seconds

  servoRight.writeMicroseconds(1700);          // Right wheel counterclockwise
  delay(3000);                                 // ...for 3 seconds

  servoRight.writeMicroseconds(1500);          // Right wheel counterclockwise

}

void loop()                                    // Main loop auto-repeats
{                                              // Empty, nothing needs repeating
}
```

A Loop that Repeats While a Condition is True

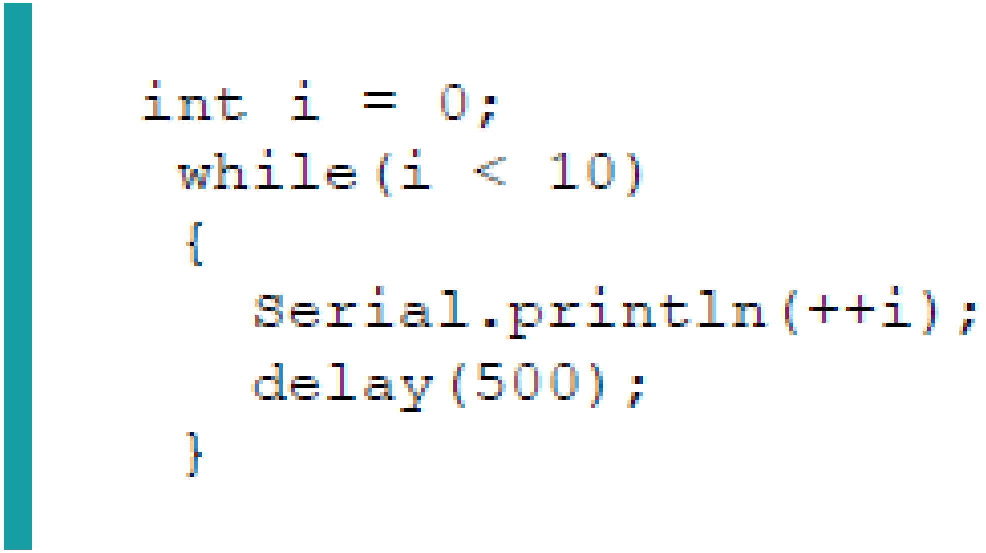


```
int i = 0;
  while(i < 10)
  {
    i = i + 1;
    Serial.println(i);
    delay(500);
  }
```

Coding with Loops…

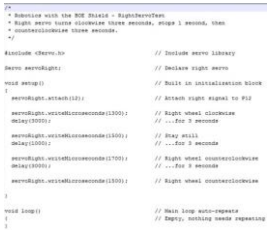
```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

Condensing code using "++" within the Serial.println ( ). The variable "i" will increment by 1.



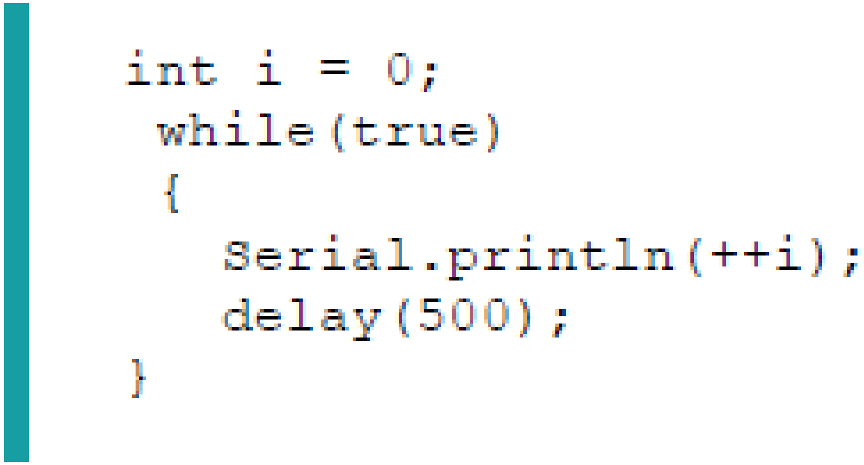
```
int i = 0;
 while(i < 10)
 {
    Serial.println(++i);
    delay(500);
 }
```

Coding with Loops…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                           // Include servo library

Servo servoRight;                            // Declare right servo

void setup()                                 // Built in initialization block
{
  servoRight.attach(12);                     // Attach right signal to P12

  servoRight.writeMicroseconds(1300);        // Right wheel clockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Stay still
  delay(1000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1700);        // Right wheel counterclockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Right wheel counterclockwise

}

void loop()                                  // Main loop auto-repeats
{                                            // Empty, nothing needs repeating
}
```

A “while” loop keeps repeating as long as the Boolean statement is true. The word “true” is a pre-defined constant.



```
int i = 0;
 while(true)
 {
    Serial.println(++i);
    delay(500);
}
```

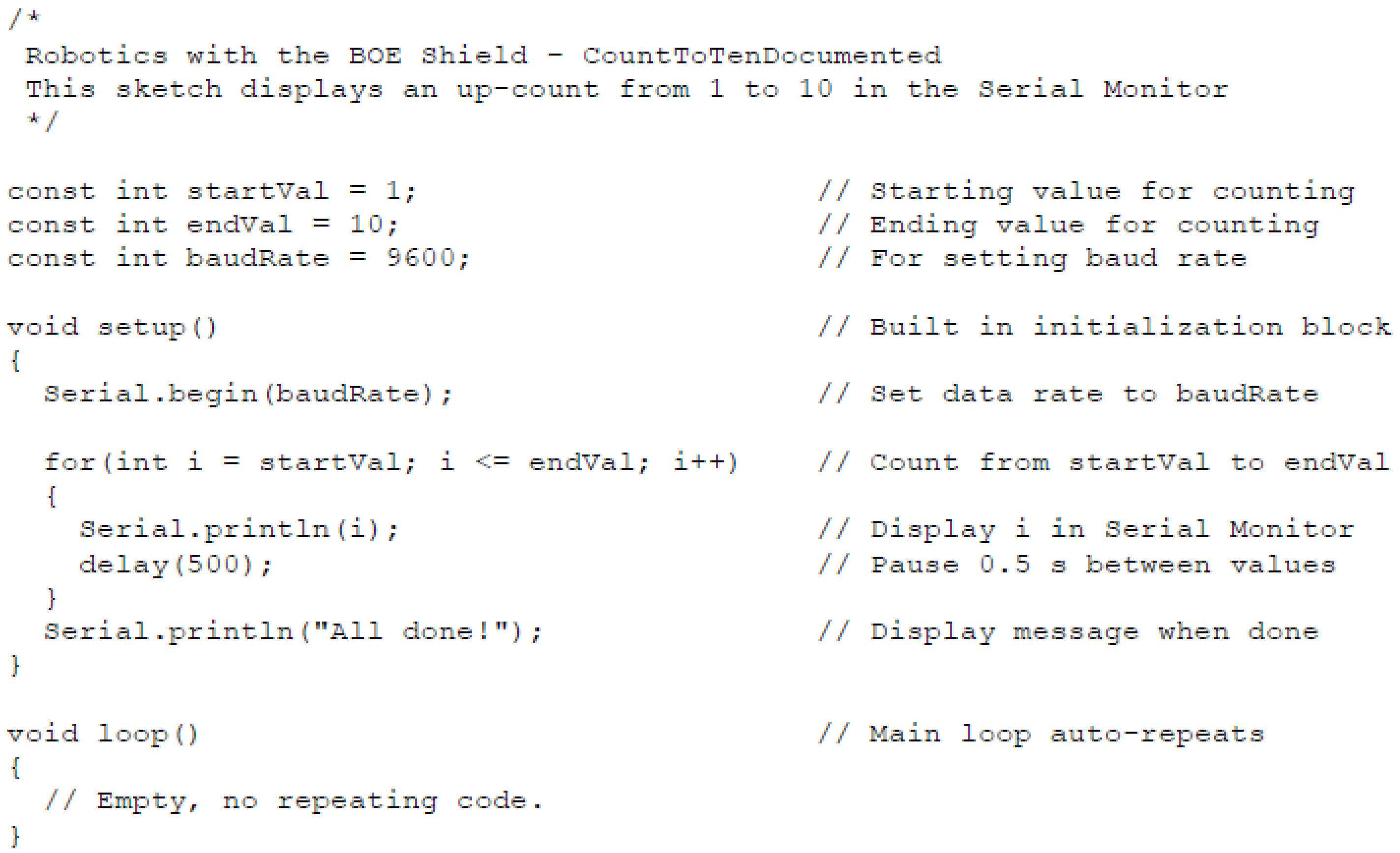
Question 5

Write the coding statement that will achieve condensing code when creating program loops.

Coding with Loops…

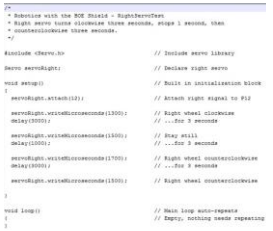
```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

Constants and Comments

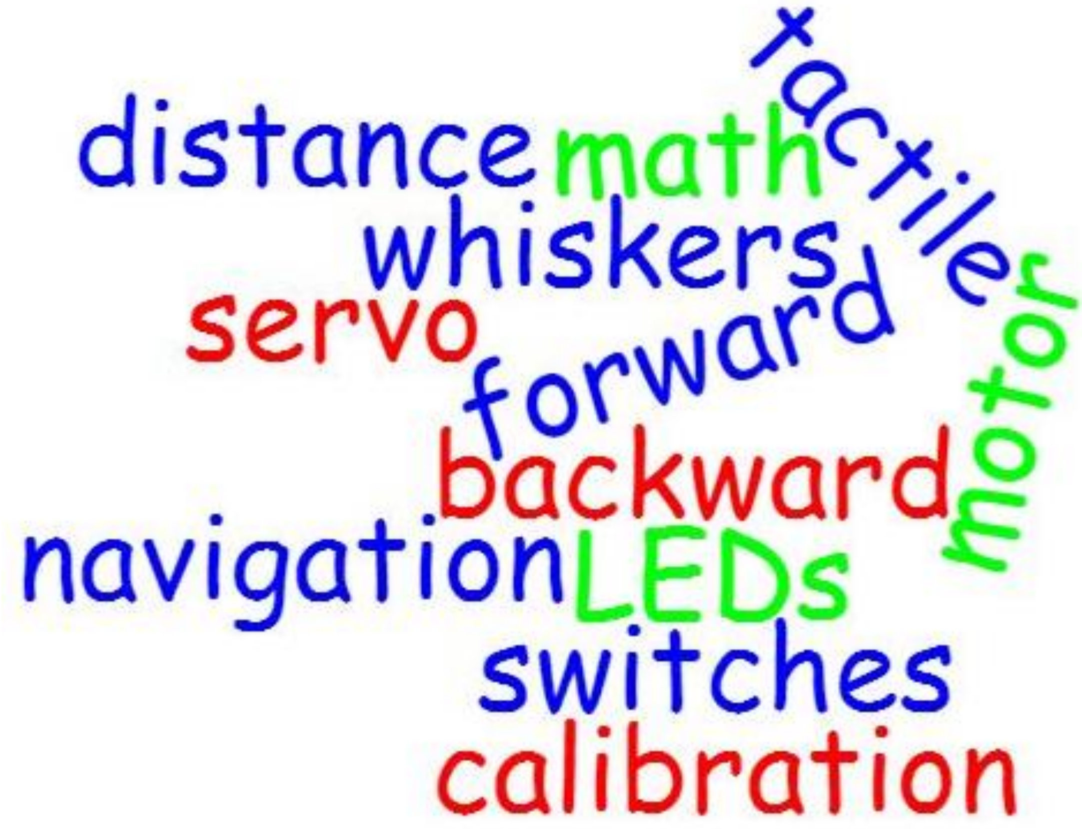


```
/*
 Robotics with the BOE Shield - CountToTenDocumented
 This sketch displays an up-count from 1 to 10 in the Serial Monitor
 */

const int startVal = 1;                    // Starting value for counting
const int endVal = 10;                     // Ending value for counting
const int baudRate = 9600;                 // For setting baud rate

void setup()                               // Built in initialization block
{
  Serial.begin(baudRate);                  // Set data rate to baudRate

  for(int i = startVal; i <= endVal; i++)  // Count from startVal to endVal
  {
    Serial.println(i);                     // Display i in Serial Monitor
    delay(500);                            // Pause 0.5 s between values
  }
  Serial.println("All done!");             // Display message when done
}

void loop()                                // Main loop auto-repeats
{
  // Empty, no repeating code.
}
```

Hands-On Labs: Coding Examples

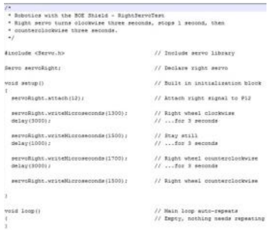


```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                           // Include servo library

Servo servoRight;                            // Declare right servo

void setup()                                 // Built in initialization block
{
  servoRight.attach(12);                     // Attach right signal to P12

  servoRight.writeMicroseconds(1300);        // Right wheel clockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Stay still
  delay(1000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1700);        // Right wheel counterclockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Right wheel counterclockwise

}

void loop()                                  // Main loop auto-repeats
{                                            // Empty, nothing needs repeating
}
```



Presented by:

Hands-On Labs: Coding Examples…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

Objectives of Coding Labs

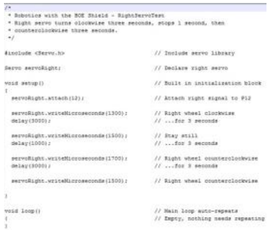
- To insure the Arduino IDE is installed correctly.
- To explore the Arduino IDE's programming environment.
- To explore turning the Arduino into a servo motor controller.

Note: We'll be using examples from Robotics with Board of Education Shield for Arduino Book by Andy Lindsay, version 1.0

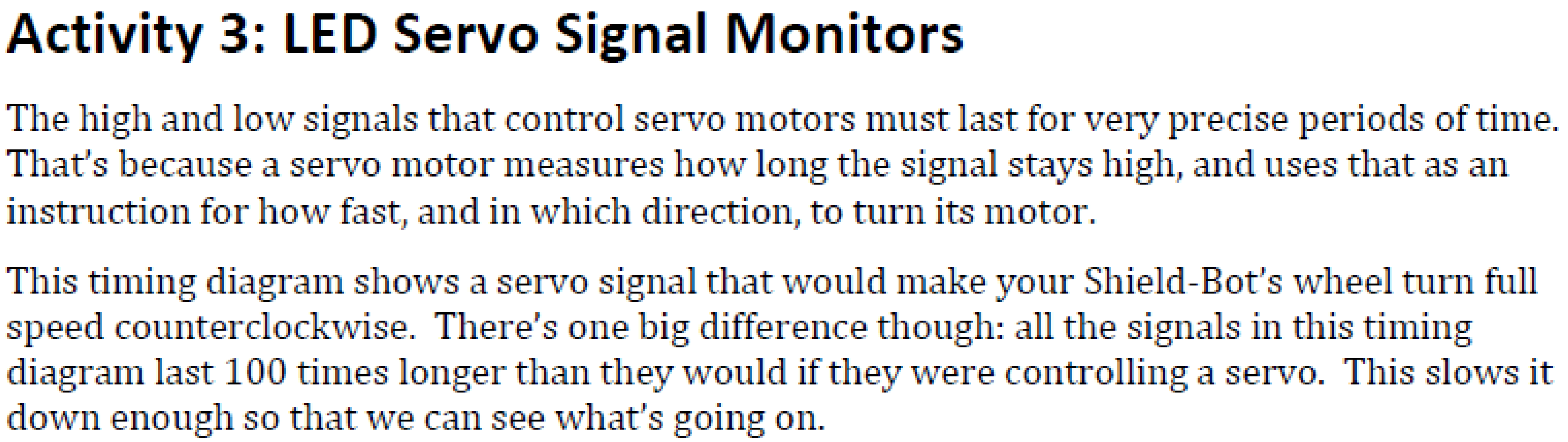
Book Source:

https://www.parallax.com/sites/default/files/downloads/122-32335-Robotics-BOE-Shield-Bot-Arduino-v1.0.pdf

Hands-On Labs: Coding Examples…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

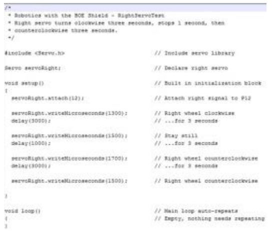


Activity 3: LED Servo Signal Monitors

The high and low signals that control servo motors must last for very precise periods of time. That's because a servo motor measures how long the signal stays high, and uses that as an instruction for how fast, and in which direction, to turn its motor.

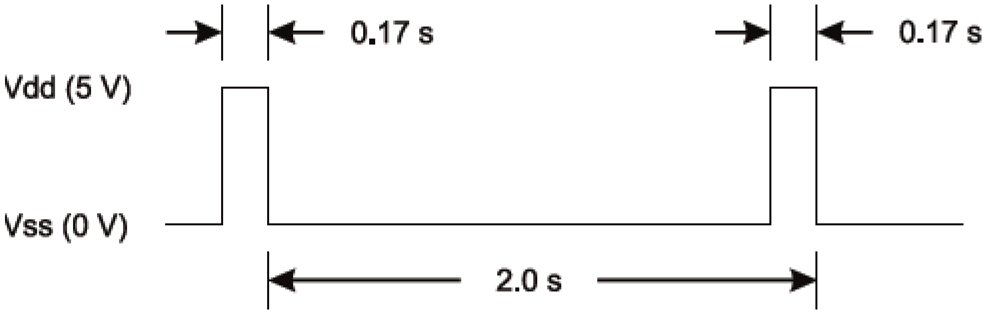
This timing diagram shows a servo signal that would make your Shield-Bot's wheel turn full speed counterclockwise. There's one big difference though: all the signals in this timing diagram last 100 times longer than they would if they were controlling a servo. This slows it down enough so that we can see what's going on.

Hands-On Labs: Coding Examples…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

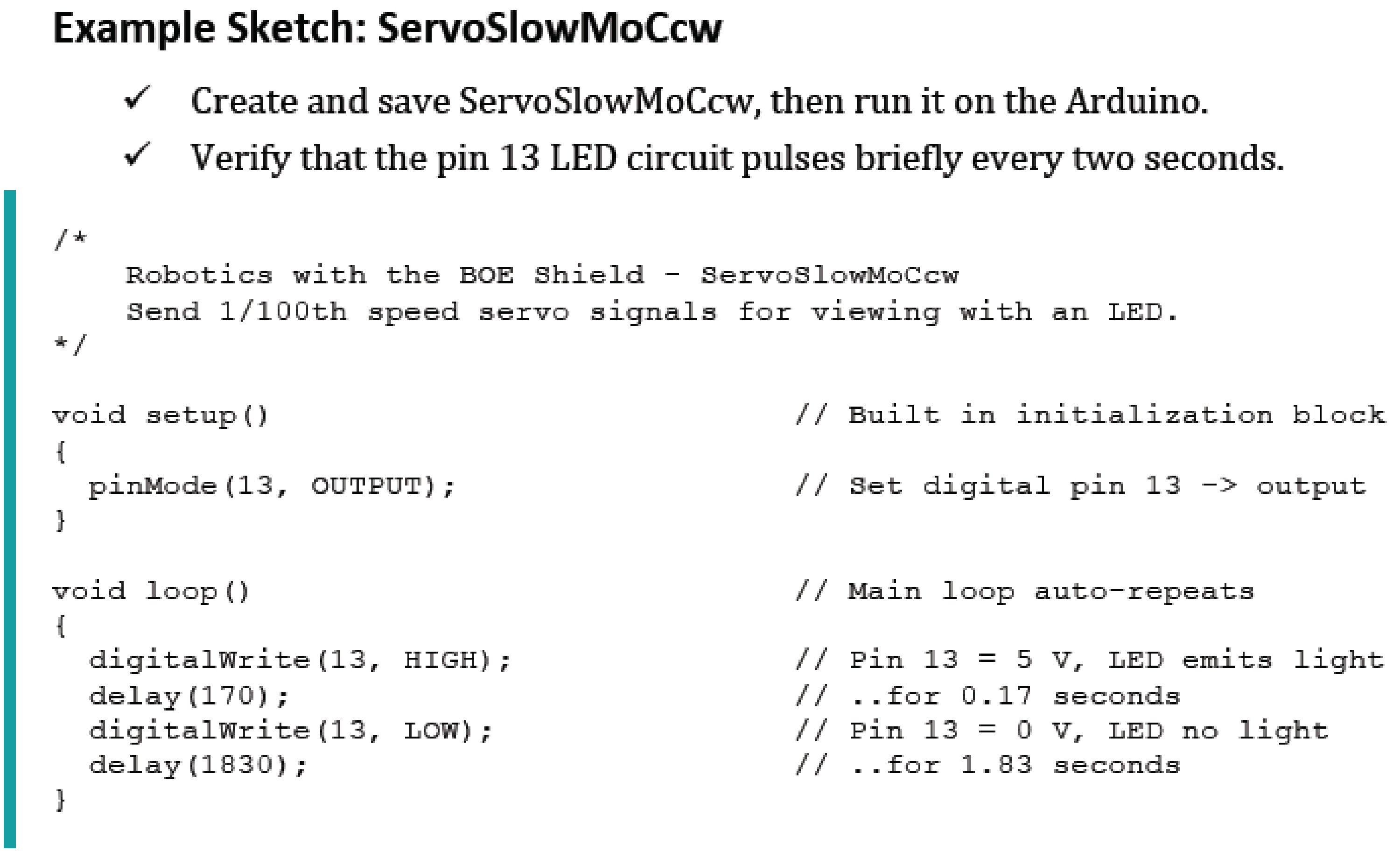
Timing Diagram for Servo Motor full speed counterclockwise rotation



Presented by:

Hands-On Labs: Coding Examples…

```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

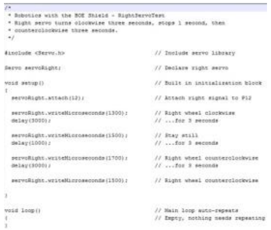


Example Sketch: ServoSlowMoCcw

- ✓ Create and save ServoSlowMoCcw, then run it on the Arduino.
- ✓ Verify that the pin 13 LED circuit pulses briefly every two seconds.

```
/*
   Robotics with the BOE Shield - ServoSlowMoCcw
   Send 1/100th speed servo signals for viewing with an LED.
*/

void setup()                                 // Built in initialization block
{
  pinMode(13, OUTPUT);                       // Set digital pin 13 -> output
}

void loop()                                  // Main loop auto-repeats
{
  digitalWrite(13, HIGH);                    // Pin 13 = 5 V, LED emits light
  delay(170);                                // ..for 0.17 seconds
  digitalWrite(13, LOW);                     // Pin 13 = 0 V, LED no light
  delay(1830);                               // ..for 1.83 seconds
}
```

Hands-On Labs: Coding Examples…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

Using the Servo library to send servo control signals takes four steps:

1. Tell the Arduino editor that you want access to the Servo library functions with the #include declaration at the start of your sketch, before the **setup** function.

```
#include <Servo.h>          // Include servo library
```

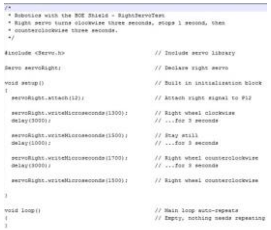
2. Declare and name an instance of the Servo library for each signal you want to send, between **#include** and the **setup** function.

```
Servo servoLeft;            // Declare left servo
```

3. In the **setup** function, use the name you gave the servo signal followed by a dot, and then the attach function call to attach the signal pin. This example is telling the system that the servo signal named **servoLeft** should be transmitted by digital pin 13.

```
servoLeft.attach(13);      // Attach left signal to pin 13
```

Hands-On Labs: Coding Examples…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                           // Include servo library

Servo servoRight;                            // Declare right servo

void setup()                                 // Built in initialization block
{
  servoRight.attach(12);                     // Attach right signal to P12

  servoRight.writeMicroseconds(1300);        // Right wheel clockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Stay still
  delay(1000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1700);        // Right wheel counterclockwise
  delay(3000);                               // ...for 3 seconds

  servoRight.writeMicroseconds(1500);        // Right wheel counterclockwise

}

void loop()                                  // Main loop auto-repeats
{                                            // Empty, nothing needs repeating
}
```

4. Use the **`writeMicroseconds`** function to set the pulse time. You can do this inside either the **`setup`** or **`loop`** function:

```
servoLeft.writeMicroseconds(1500); // 1.5 ms stay-still signal
```

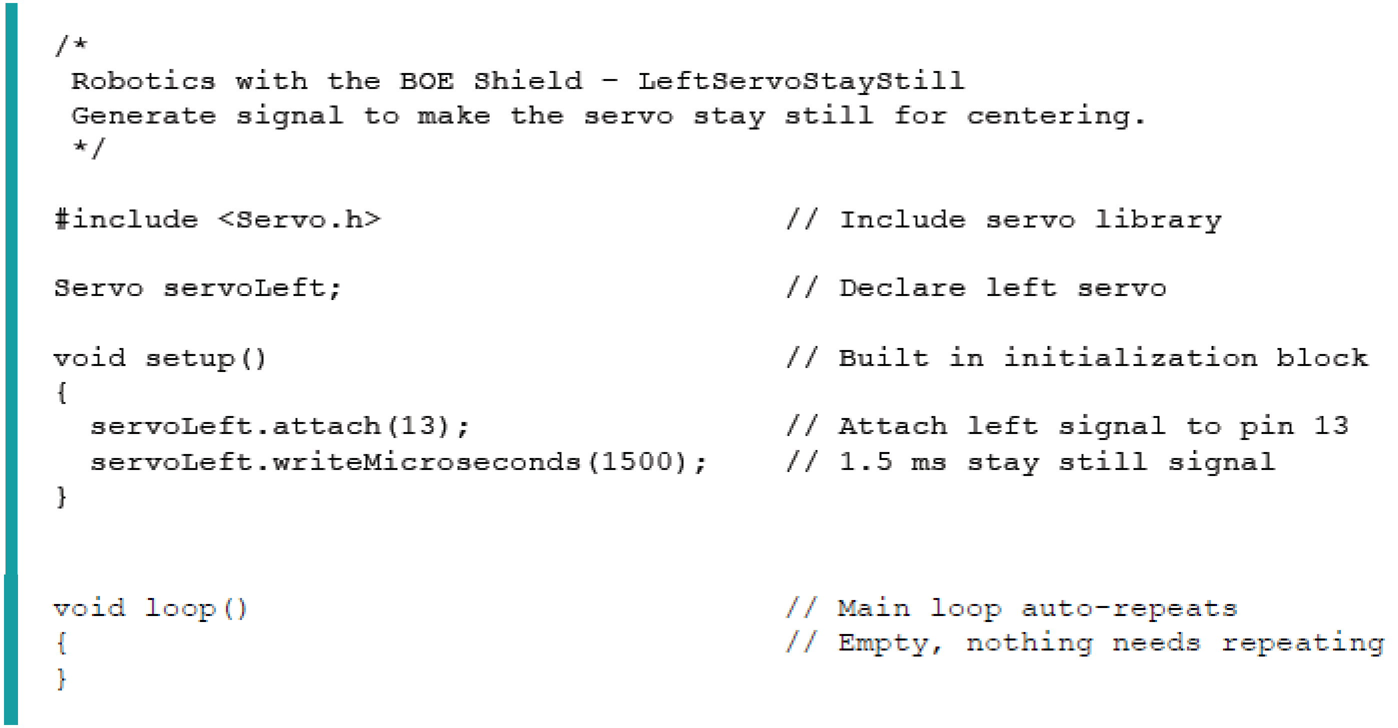
Seconds, Milliseconds, Microseconds

- A **millisecond** is a one-thousandth of a second, abbreviated ms.
- A **microsecond** is a one-millionth of a second, abbreviated µs.
- There are 1000 microseconds (µs) in 1 millisecond (ms).
- There are 1,000,000 microseconds in 1 second (s).

Hands-On Labs: Coding Examples…

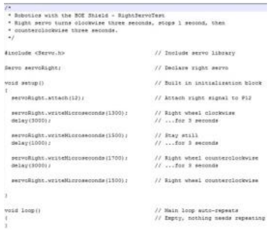
```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

LeftServoStayStill Code



```
/*
 Robotics with the BOE Shield - LeftServoStayStill
 Generate signal to make the servo stay still for centering.
 */

#include <Servo.h>                          // Include servo library

Servo servoLeft;                            // Declare left servo

void setup()                                // Built in initialization block
{
  servoLeft.attach(13);                     // Attach left signal to pin 13
  servoLeft.writeMicroseconds(1500);        // 1.5 ms stay still signal
}


void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

Hands-On Labs: Coding Examples…



```
/*
 * Robotics with the BOE Shield - RightServoTest
 * Right servo turns clockwise three seconds, stops 1 second, then
 * counterclockwise three seconds.
 */

#include <Servo.h>                          // Include servo library

Servo servoRight;                           // Declare right servo

void setup()                                // Built in initialization block
{
  servoRight.attach(12);                    // Attach right signal to P12

  servoRight.writeMicroseconds(1300);       // Right wheel clockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Stay still
  delay(1000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1700);       // Right wheel counterclockwise
  delay(3000);                              // ...for 3 seconds

  servoRight.writeMicroseconds(1500);       // Right wheel counterclockwise

}

void loop()                                 // Main loop auto-repeats
{                                           // Empty, nothing needs repeating
}
```

Design Challenge:

Modify the code in the previous slide to make the right servo motor stay still for centering.