

Robotic Sunflower Lesson 2: Controlling a Servo

AUTHOR: Pat Blount

DESCRIPTION: In this lesson students will learn how to control a servo using the Basic Stamp. Then students will combine the photoresistor from the previous lesson with the servo to create a light controlled servo.

GRADE LEVEL(S): 9, 10, 11, 12

SUBJECT AREA(S): Electricity, electronics, computer science, applied physics

ACTIVITY LENGTH: 1 hour, 40 minutes

LEARNING GOAL(S):

1. Students will be able to apply the pulse width modulation to a servo from a Basic Stamp. 2. Students will synthesize two circuit designs using one to control the other through the Basic Stamp.

STANDARDS MET:

Common Core:

- CCSS.ELA-Literacy.RST.11-12.9. Synthesize information from a range of sources (e.g., texts, experiments, simulations) into a coherent understanding of a process, phenomenon, or concept, resolving conflicting information when possible.
- CCSS.ELA-Literacy.RST.9-10.3. Follow precisely a complex multistep procedure when carrying out experiments, taking measurements, or performing technical tasks, attending to special cases or exceptions defined in the text.
 - CCSS.ELA-Literacy.RST.9-10.7. Translate quantitative or technical information expressed in words in a text into visual form (e.g., a table or chart) and translate information expressed visually or mathematically (e.g., in an equation) into words.

Next Generation Science Standards:

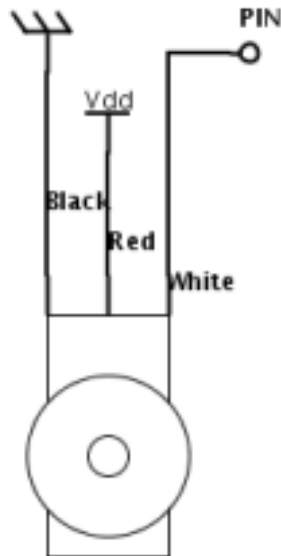
- HS-PS3-1. Create a computational model to calculate the change in the energy of one component in a system when the change in energy of the other component(s) and energy flows in and out of the system are known.
 - HS-PS3-3. Design, build, and refine a device that works within given constraints to convert one form of energy into another form of energy.
 - HS-ETS1-1. Analyze a major global challenge to specify qualitative and quantitative criteria and constraints for solutions that account for societal needs and wants.
 - HS-ETS1-2. Design a solution to a complex real-world problem by breaking it down into smaller, more manageable problems that can be solved through engineering.
 - HS-ETS1-3. Evaluate a solution to a complex real-world problem-based on prioritized criteria and trade-offs that account for a range of constraints, including cost, safety, reliability, and aesthetics, as well as possible social, cultural, and environmental impacts.
-

Student Background:

This lesson was developed as the second in a unit that culminates in the construction of a robotic sunflower that tracks the sun. While it will provide a lot of the foundational knowledge that the rest of the unit will be built on, it is helpful if students have some understanding of basic electricity concepts, including voltage, current, resistance, how a diode works and how to use a multimeter. My students will already have completed the following lessons beforehand: • Robotic Sunflower Lesson 1: Measuring Voltage with a Microcontroller

Educator Background:

Connecting a servo to a Basic Stamp requires just 3 connections. The white cable goes to a pin, red to a voltage source, and black to ground. The voltage source can be Vdd on the Homework Board but only for testing. When ready to drive the servo it is best to connect a separate power source for they can drain a battery fast. Consult the Parallax.com Web site for the data sheet on the servo. In addition you can consult the “learn” section (<http://learn.parallax.com/>) for information on: Parallax Standard Servo. This area of Parallax’s site covers many Basic Stamp applications and is a good place to go to learn how to accomplish a task.



Circuit 1: Servo circuit. Black goes to ground, red to 5V - 6V, and white to the Basic Stamp.

A worthwhile endeavor if an oscilloscope is available is to show the class the pulse width modulation that the STAMP is sending out to the servo. This helps the students understand more of what is happening.

Below is sample code for the servo. The program moves the servo back and forth between its left and right stops. It displays in the debug window what position the servo is being set. Several items are worth noting in program 3. Using pin assignments allows one to change the pin number at one location and affect the whole program. Use variable assignments that are meaningful.

```

'{$STAMP BS2}
'{$PBASIC 2.5}

'Servo control
'this program moves a servo back and forth thru a 180 degree rotation.
'the servo is controlled by a PULSOUT command. the arguments for the command are
'the pin, and the duration. duration is the number of two millionths of a second time
'increments the pulse should last. or 2 micro-seconds.
'this program will rotate the servo connected to servo1 pin first counterclockwise and the clockwise
'in steps of 100 micro-seconds. in addition it will display the steps on the debug screen.

'constants and pin assignments-----

servo1 PIN 12 'assigns the word servo1 to a pin number. this allows the user to change the pin assignment
'for different circuit configurations without changing multiple occurrences in the program.

'variables-----

turn VAR Word 'makes the word "turn" a 16 bit (0 to 65535) sized variable.
counter VAR Nib 'makes the word "counter" a 4 bit (0 to 15) sized variable.

'initializations-----

turn = 300 'sets the initial value of the variable "turn" to 300

'main program-----

DO
'rotates servo counterclockwise
DO
FOR counter = 1 TO 10 STEP 1 'sends a pulse to servo1 for ten intervals
PULSOUT servo1,turn'sends a pulse out to servo1 for a duration equal to the value of "turn"
PAUSE 20 'pauses for 20 micro-sec, necessary for servo operation
NEXT
DEBUG CLS 'clears the debug screen
DEBUG DEC turn, HOME 'displays "turn" value, as a decimal, on debug screen,
'then updates over the old value in top left corner of screen.
turn = turn + 100 'increases "turn" by 100
LOOP UNTIL turn > 1150 'continues to rotate servo until it reaches the max counterclockwise rotation

'rotates servo clockwise
DO
FOR counter = 1 TO 10 STEP 1 'sends a pulse to servo1 for ten intervals
PULSOUT servo1,turn'sends a pulse out to servo1 for a duration equal to the value of "turn"
PAUSE 20 'pauses for 20 micro-sec, necessary for servo operation
NEXT
DEBUG CLS 'clears the debug screen
DEBUG DEC turn, HOME 'displays "turn" value, as a decimal, on debug screen,
'then updates over the old value in top left corner of screen.
turn = turn - 100 'decreases "turn" by 100
LOOP UNTIL turn < 250 'continues to rotate servo until it reaches the max clockwise rotation

LOOP

```

Program 3: Basic servo movement.

Program 4 integrates the photoresistor circuit with the servo circuit. The program reads the photoresistor using the RCTIME command and then uses that information to control the servo. Note the use of pin assignments and the use of subroutines. Keep in mind is the format of calling a subroutine, GOSUB xxxxxx, and the header for a subroutine, xxxx:, followed by RETURN. Subroutines allow students to use programs from previous lessons. A program can be added at the bottom of the main program and called by the main whenever needed. In this way students can create a library of code modules.

```

'{$STAMP BS2}
'{$PBASIC 2.5}

'servo control with photo resistor
'this program allows the user to control a servo by controlling the light level on a photodiode.
'the user moves a hand up and down above a photodiode to control the amount of light incident on it.
'this in turn rotates the servo to a position that relates to the light level

'constants and pin assignments-----
sensor PIN 14 'assigns the word "sensor" to a pin number. This allows the user to change the pin assignment
'for different circuit configurations without change multiple occurrences in the program.
Servo1 PIN 12 'assigns the word servo1 to a pin number. This allows the user to change the pin assignment 'for
different circuit configurations without change multiple occurrences in the program.

'variables-----
rct VAR Word 'variable for the RCTIME count. Is declared as a word, 0 to 65535
turn VAR Word 'makes the word "turn" a 16 bit (0 to 65535) sized variable.
Counter VAR Nib 'makes the word "counter" a 4 bit (0 to 15) sized variable.

'initializations-----
HIGH sensor 'sets the pin to high, 5V, thus setting both plates of the cap to 5V. this allows
'the RCTIME command to time the fall of the voltage to 1.3V when the pin goes low.
FOR counter = 1 TO 10 'this loop centers the servo, 750 represents the approximate center of the
servos PULSOUT servo1,750'range of motion. This must be determined for each servo
PAUSE 20 'a short pause is necessary for the pulsout command to the servo
NEXT 'the FOR ... TO... NEXT control structure acts like a fixed iteration
'number of loops. The counter variable determines the number of loops.
'main program-----

DO
GOSUB lightlevel'sends program control to measure the light level
GOSUB moveservo'moves the servo based on the light level
LOOP

'-----
'subroutines-----
'-----

'read light sensor-----

lightlevel: 'subroutine for reading the light level

DO 'loop to verify that rctime interval count is not outside range of 65535
RCTIME sensor,1,rct'rctime arguments include pin number, initial state, and variable to store time.
'initial state indicates whether the capacitor starts high, 5V, or low, <1.3 V.
HIGH sensor 'cap has discharged so needs to be recharged
LOOP UNTIL (rct<> 0)
RETURN 'returns control of the program to the main loop

'move servo-----

moveservo: 'subroutine for moving the servo

turn = rct * 9 / 10 + 50 'converts the rct variable into a number that can be used for pulsout
'9/10 +50 was determined by the max and min rct values for the lighting conditions
DEBUG CLS, "rct", TAB, "turn",CR'displays the rct and the turn values for adjustments
DEBUG DEC rct,TAB, DEC turn
FOR counter = 1 TO 10 STEP 1 'adjusts the servo at servo1 to an angle determined by the turn variable
PULSOUT servo1,turn
PAUSE 20
NEXT
RETURN

```

Program 4: Controlling a servo w/ the photo-resistor

Lastly, program 5 is a sample code that uses the voltage divider circuit to control the servo. It is much the same as the RC program and differs only in how it reads the light sensor. Of note is that the light reading is either on or off, unlike the RC sensor. So, in effect, this program will set the servo one way or the other based on the photoresistor voltage level.

```

'{$STAMP BS2}
'{$PBASIC 2.5}

'servo control with photoresistor
'this program allows the user to control a servo by controlling the light level on a photoresistor.
'the user moves a hand up and down above a photoresistor to control the amount of light incident on
it. 'this in turn rotates the servo to a position that relates to the light level

'constants and pin assignments-----

servo1 PIN 12 'assigns the word servo1 to a pin number. This allows the user to change the pin assignment 'for
different circuit configurations without changing multiple occurrences in the program.
Delta CON 100 'the increase in the servo position from the last value

'variables-----

sensor VAR IN14 'assigns the word "sensor" to a pin number as an input. This returns either a 1 or 0
'determined by the voltage level. Above 1.4V is 1 and below is 0.
Counter VAR Nib 'makes the word "counter" a 4 bit (0 to 15) sized variable.
Position VAR Word

'initializations-----
position = 750

FOR counter = 1 TO 10 'this loop centers the servo, 750 represents the approximate center of the
servos PULSOUT servo1,position'range of motion. This must be determined for each servo
PAUSE 20 'a short pause is necessary for the pulsout command to the servo
NEXT 'the FOR ... TO.. .. NEXT control structure acts like a fixed iteration
'number of loops. The counter variable determines the number of loops.
'main program-----

DO

IF sensor = 1 THEN
  DEBUG CLS, "sensor", TAB, "position",CR'displays the sensor and the position values for
  adjustments  DEBUG DEC sensor,TAB, DEC position

  position = position + delta 'adds a change in position given by the delta value to position.
  FOR counter = 1 TO 10 STEP 1 'adjusts the servo at servo1 to an angle determined by the position variable
  PULSOUT servo1,position
  PAUSE 20
  NEXT
ELSEIF sensor = 0 THEN
  DEBUG CLS, "sensor", TAB, "position",CR'displays the sensor and the position values for
  adjustments  DEBUG DEC sensor,TAB, DEC position

  position = position - delta
  FOR counter = 1 TO 10 STEP 1 'adjusts the servo at servo1 to an angle determined by the position variable
  PULSOUT servo1,position
  PAUSE 20
  NEXT
ENDIF

LOOP

```

Program 5: Voltage Divider and Servo

The RC circuit is better at determining light levels whereas the voltage divider circuit is useful for determining if the light level is above or below a set level. Both are explored because student teams can choose different solution paths. Each path will have its own problems to overcome.

Other Materials List:

- ["Robotic Sunflower Lesson 2.1- Extension to Controlling a Servo" handout](#)
- ["Student Guide" handout](#)
- ["Parallax Continuous Rotation Servo" handout](#)
- ["StampWorks Hobby Servo Control" handout](#)

- [“Texas Instruments L293” handout](#)
- [“The Nuts and Volts of Basic Stamps” handout](#)
- [“Parallax Standard Servo” handout](#)
- Basic Stamp Homework Board (1 per group)
- Parallax servo (1 per group)
- Photo-resistor (1 per group)
- 5k Potentiometer (1 per group)
- 220 Ohm resistor (1 per group)
- 440 Ohm resistor (1 per group)
- 2k Ohm resistor (1 per group)
- 10K Ohm resistor (1 per group)
- 0.01 uF Capacitor (1 per group)
- Jumper wires (1 set per group)

Vocabulary:

- PIN
- VAR
- Byte
- Bit
- Nib
- Word
- FOR ... TO...NEXT
- IF...THEN...ELSEIF...ENDIF
- PULSOUT
- INS
- DO...LOOP
- HIGH
- LOW
- INq
- BIN1
- Comment
- Directives
- RCTIME
- CLS
- CRSR

Lesson Details:

Planning

Ensure that there are enough materials for each group. In addition, students should be able to access a computer. If not already there, download pBasic 2.5 onto the computers for programming the Basic Stamps. It's a free program available at www.parallax.com. Have students watch the following YouTube video the night before starting day one of this lesson: • Circuit Skills: PWM (Pulse Width Modulation) at

<http://www.youtube.com/watch?v=Lf7JJAAZxEU>

- And this video on the night before day two of this lesson: Servo controller with a

potentiometer at <http://www.youtube.com/watch?v=P3iQENGL6il> . This lesson will replace the potentiometer with the photoresistor.

Class Sequence

- Students should have watched the video, Circuit Skills: PWM (Pulse Width Modulation), for homework. This sets up the day for exploring the servo.
- Day one will have students build Circuit 5, attaching the servo to the Homework Board. It might be worthwhile having students use an external power source for the servos as they can drain a 9V pretty fast. On the final build all servos will be powered externally for this reason. • After building the circuits, give them Program 4 to get the servo moving.
- Challenge the teams to move the servo in various patterns, such as left 45 degrees, then right 60 degrees, then left ..., you get the idea. Come up with a pattern that they must match. Use the grading rubric below to grade their attempts.

Program	Max Pts	Pts Earned	Grading Criteria	Instructor Initial
Servo Control	3		Circuit is wired correctly. ____/1 Program entered correctly, ____/2 downloads and runs as specified.	
Servo Challenge	3		Circuit is wired correctly. ____/1 Program entered correctly, ____/2 downloads and runs as specified.	
Document	2		Documentation and comments. ____/2	
Teamwork	2		Worked well and contributed. ____/2	
Total:				

Table 1: Servo rubric.

- Prior to day two have students watch at home the video as noted above: Servo controller with a potentiometer. Ask them to record in their engineering notebook what they think the programming is to use a potentiometer to control the servo.
- Day two will take a voltage divider circuit **or** an RC circuit and combine it with the servo from day one of this lesson. Program 4 and 5 are samples of possible programming solutions. Start the period off by discussing the video and possible solutions. Then set the students loose to build and program the circuit.
- Assess their efforts with the rubric in Table 1.

Program	Max Pts	Pts Earned	Grading Criteria	Instructor Initial
---------	---------	------------	------------------	--------------------

Servo Control w/ photo	3		Circuit is wired correctly. ____/1 Program entered correctly, downloads and runs as ____/2 specified.	
Document	2		Documentation and comments. ____/2	
Teamwork	2		Worked well and contributed. ____/2	
Total:				

Table 2: Servo control with photoresistor

- Have students document the circuits in their engineering notebook. Be sure to have them include pulsout values that give a max counterclockwise and clockwise rotation and the min value as well. At this point they should have the following in their notebooks:
 - ✓ Photoresistor characteristics in full/partial/no sun.
 - ✓ RC values with average decay times.
 - ✓ Servo minimum and maximum values for the range of motion.
 - ✓ List of programming commands and their syntax.

Assessment

Formative in class assessment and homework occur during this lesson