Robotic Sunflower Lesson 4: Dual Axis Light Tracking

AUTHOR: Pat Blount

DESCRIPTION: Students will take the previous lesson and apply them in creating a light tracker with two degrees of freedom. The axis of rotation will be about the horizontal and vertical. Teams will have everything they need to make this build work. They have already created a tracker with one degree of freedom, so adding a second is a matter of replication. Students will use an additional servo circuit to add the next degree of freedom.

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):
Students will create a solar tracker with two degrees of freedom.

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 fourth in a unit that culminates in the construction of a robotic sunflower that tracks the sun. My students will already have completed the following lessons beforehand:

- Robotic Sunflower Lesson 1: Measuring Voltage with a Microcontroller
- Robotic Sunflower Lesson 2: Controlling a Servo
- Robotic Sunflower Lesson 2.1: Extension to Controlling a Servo
- Robotic Sunflower Lesson 3: Creating a Light-Tracking Servo

Educator Background:

As student programs grow in complexity, shortcomings in their programming techniques start to show. Variable names, for example, can help interpret what's going on in their programs. At this point the teacher is floating and helping to put out spot fires with the teams. Students have all the prerequisites to successfully complete this build.

Tell students to think about how the program will handle cloudy weather, sunrise and sunset. Below is a sample program (Program 8). The program includes an LCD display for measurement purposes. It is not needed in the tracking portion of the program.

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

‘dualservo control with quad photoresistors
'this program controls a servos by the using light levels on a 4 of photoresistors.
'this version has anlcddisplay added to it

'constants and pin assignments----------------------------------------------------------------------------

left_photo      PIN     14      'assigns the word "left_photo" to a pin number. this is for the photoresistor. this
allows the                                                      ‘user to change the pin assignment
'for different circuit configurations without change multiple occurrences in the program.
right_photo     PIN     0       'left_light is the left photoresistor and right_light is the right.
up_photo PIN     15     'up photoresistor
down_photoPIN     1     'down photoresistor
LR_servo  PIN     12     'assigns the word LR_servo to a pin number. this allows the user to change the pin
assignment
                                        ‘for the servo controlling Left/Right Motion
 'for different circuit configurations without change multiple occurrences in the program.
UD_servo        PIN     12
LCDpin PIN     2       'LCD pin assignment


baudrateCON 84     'set baudrate for the lcd to 9800 bps
differentialCON 100     'the stepping angle for the servos
```

```
LR_sensorcalibration CON 100    'a left-right light sensor calibration constant determined by dividing the max
photo rct
                                        'value by 10 and rounding to nearest 100
UD_sersorcalibration CON 100    'a up-down light sensor calibration constant determined by dividing the max
photo rct
                                        'value by 10 and rounding to nearest 100
servomin  CON 250    'min servo pulsout value, find these values with servo
servomaxCON 1150   'max servo pulsout value
night CON 1000   'a constant that determines how many loops to sample light measurements that are
                                        'outside the sensor range, i.e. night, before going to sleep

'variables----------------------------------------------------------------------------------------------------

L_rct              VAR Word    'variable for the RCTIME count for photoresistor 1.is declared as a word, 0 to
65535, to handle
                                    'higher number counts
R_rct              VAR Word    'variable for the RCTIME count for photoresistor 2.is declared as a word, 0 to
65535, to handle
                                    'higher number counts
U_rct              VAR Word    'variable for the RCTIME count for photoresistor 3.
D_rct              VAR Word    'variable for the RCTIME count for photoresistor 4.
counterVAR Nib     'makes the word "counter" a 4 bit (0 to 15) sized variable used to count a loop control.
position1 VAR Word    'makes the word "position1" a 16 bit sized variable to convert rct1 to values that
represent
                                    'ranges.
position2          VAR Word    'makes the word "position2" a 16 bit sized variable to convert rct2 to values that
represent
                                    'ranges.
position3          VAR Word
position4          VAR Word
LRposition VAR Word    'makes the word "LRposition" a 16 bit sized variable for the left right servo position to
                                    'represent the servo position from 200 to 1200
UDposition VAR Word    'makes the word "UDposition" a variable for up down servo position
dark VAR Byte    'makes the work "dark" a 8 bit sized varialbe to be used to count how many times the sensor
                                    'reading loop cycles

'initializations-----------------------------------------------------------------------------------------------

PAUSE 100                   'short pause to let things warm up
position = 700              'sets the initial value of position

HIGH left_photo             'sets the pin to high, 5V, thus setting both plates of the cap to 5V. this allows
HIGH right_photo            'the RCTIME command to time the fall of the voltage from 5V to 1.3V when the pin
goes low.
HIGH up_photo
HIGH down_photo
FOR counter = 1 TO 10
  PULSOUT LR_servo,position     'sets LR_servo to a middle position at startup
  PULSOUT UD_servo,position     'sets UD_servo to a middle position at startup
  PAUSE 20
NEXT

HIGH LCDpin                 'normal state of serial port when it isn't sending data
PAUSE 100                   'LCD needs this pause to initialize can be 100

'main program-------------------------------------------------------------------------------------------------

DO

 GOSUB lightlevel           'sends program control to measure the light level
                    'determines if the position variable is outside the servo max/min values
 IF (LRposition<servomin) OR (LRposition>servomax)OR (UDposition<servomin) OR (UDposition>servomax)
THEN
    GOSUB servoreset        'if servo outside it's operational limits then sends program control to reset the
servo
    GOSUB moveservo         'then sends control to move the servo after the reset
  ELSE
    GOSUB moveservo         'otherwise, move the servo based on the light levels
```

```
 ENDIF
 GOSUB display              'show the rct numbers on the LCD screen for calibration in sunlight
 NAP 6                      'places BS2 in low power mode (50 micro-amps vs 3 mA)for 1.152 secs, will replace w/
sleep

LOOP

'subroutines
================================================================================================
===============

'read light sensors-------------------------------------------------------------------------------------------

lightlevel:                 'subroutine for reading the light level

dark = 0                    'reinitializes the dark counter
DO                          'loop to verify that rctime interval count is not outside range of 65535
  RCTIME left_photo,1,L_rct         'rctime arguments include pin number, initial state, and variable to store time.
  RCTIME right_photo,1,R_rct        'initial state indicates whether the capacitor starts high, 5V, or low, <1.3 V.
  RCTIME up_photo,1,U_rct
rcitme down_photo,1,D_rct
  HIGH left_photo           'cap has discharged so needs to be recharged
  HIGH right_photo
  HIGH up_photo
  HIGH down_photo
dark = dark + 1
  IF dark > night THEN SLEEP 7800     'if it is too dark for sensors then the stamp will sleep. the argument is in
multiples of
                                      '2.304 secs, about 5 hours in this case
LOOP UNTIL (L_rct<>0)AND (R_rct<> 0)AND (U_rct<> 0) AND D_rct<> 0)
 'makes sure that the sensors are not overloaded i.e. does not exceed 65535, which
                                      'happens in darknesswhen rct time exceeds the max value a 0 is
returned.
RETURN                      'returns control of the program to the main loop

'move servo---------------------------------------------------------------------------------------------------

moveservo:                          'subroutine for moving the servo

SELECT L_rct                        'divides the rct1 readings into 10 intervals and assigns a number
based on
  CASE 0 TO LR_sensorcalibration            'the range of values that rct1 falls into
    position1 = 0
  CASE     LR_sensorcalibration + 1 TO 2 * LR_sensorcalibration
    position1 = 1
  CASE 2 * LR_sensorcalibration + 1 TO 3 * LR_sensorcalibration
    position1 = 2
  CASE 3 * LR_sensorcalibration + 1 TO 4 * LR_sensorcalibration
    position1 = 3
  CASE 4 * LR_sensorcalibration + 1 TO 5 * LR_sensorcalibration
    position1 = 4
  CASE 5 * LR_sensorcalibration + 1 TO 6 * LR_sensorcalibration
    position1 = 5
  CASE 6 * LR_sensorcalibration + 1 TO 7 * LR_sensorcalibration
    position1 = 6
  CASE 7 * LR_sensorcalibration + 1 TO 8 * LR_sensorcalibration
    position1 = 7
  CASE 8 * LR_sensorcalibration + 1 TO 9 * LR_sensorcalibration
    position1 = 8
  CASE 9 * LR_sensorcalibration + 1 TO 10 * LR_sensorcalibration
    position1 = 9
ENDSELECT

SELECT R_rct
  CASE 0 TO LR_sensorcalibration
    position2 = 0
  CASE     LR_sensorcalibration + 1 TO 2 * LR_sensorcalibration
    position2 = 1
```

```
  CASE 2 * LR_sensorcalibration + 1 TO 3 * LR_sensorcalibration
    position2 = 2
  CASE 3 * LR_sensorcalibration + 1 TO 4 * LR_sensorcalibration
    position2 = 3
  CASE 4 * LR_sensorcalibration + 1 TO 5 * LR_sensorcalibration
    position2 = 4
  CASE 5 * LR_sensorcalibration + 1 TO 6 * LR_sensorcalibration
    position2 = 5
  CASE 6 * LR_sensorcalibration + 1 TO 7 * LR_sensorcalibration
    position2 = 6
  CASE 7 * LR_sensorcalibration + 1 TO 8 * LR_sensorcalibration
    position2 = 7
  CASE 8 * LR_sensorcalibration + 1 TO 9 * LR_sensorcalibration
    position2 = 8
  CASE 9 * LR_sensorcalibration + 1 TO 10 * LR_sensorcalibration
    position2 = 9
ENDSELECT

IF position1 < position2 THEN                'decides if rct1 or rct2 is larger then increases or decreases the
position
position = position + differential        'of the servo by the deferential.
    FOR counter = 1 TO 10
      PULSOUT LR_servo,position
      PAUSE 20
    NEXT
ELSEIF position1 > position2 THEN
position = position - differential
    FOR counter = 1 TO 10
      PULSOUT LR_servo,position
      PAUSE 20
    NEXT
ENDIF

DEBUG CLS, "L_rct", TAB,"R_rct",TAB,"position1",TAB,"position2",TAB,"position",CR          'DISPlays the rct1 ,
rct2, and
                                                                                                                    'the turn
values for adjustments
DEBUG DEC L_rct,TAB, DEC R_rct,TAB,TAB,DEC position1,TAB,TAB,DEC position2,TAB, DEC position

RETURN

'servorest ----------------------------------------------------------------------------------------------------------

servoreset:

position = 700
FOR counter = 1 TO 10
  PULSOUT LR_servo, position
  PAUSE 20
NEXT

RETURN

'LCD display ---------------------------------------------------------------------------------------------------------

display:

SEROUT LCDpin,baudrate,[12]
PAUSE 5
SEROUT LCDpin,baudrate,["L_rct",TAB,DEC L_rct,13]
SEROUT LCDpin,baudrate,["R_rct",TAB, DEC R_rct,CR]

RETURN
```

Program 8: Two degrees of Freedom.

Other Materials List:

Students will need to build an additional servo photoresistor circuit in order to complete this activity, which will require the following materials:

- "DC Motors" student handout
- "Laboratory 7: DC Motor Driver, Matrix Keypad, and Liquid Crystal Display" student handout
- "BASIC Stamp Laboratory, Part 1" student handout
- "DC Motors Direction and Speed" student handout
- Photoresistor (2/group)
- 220 Ohm resistor (2/group)
- 440 Ohm resistor (2/group)
- 2k Ohm resistor (2/group)
- 10K Ohm resistor (2/group)
- 0.01 uF Capacitor (2/group)
- 5k Ohm potentiometer (2/group)
- One DC gearhead servo motor

Vocabulary:

- SEROUT
- SELECT…CASE
- Baudrate

……

Lesson Details:

Planning and Prep

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.

Class Sequence

Students have all the information they need to accomplish this task. Teams should split the work with one or two members working on the circuit and the physics set up for the sunflower while the other(s) work on the programming. Most of the programming is cut and paste. Teams should be able to finish in the given time.

Assessment

The rubric below can be used to evaluate the final product.

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

T

Table 1: Sunflower

At this point students should present their solution to the class. Teams should highlight details of their solution and demonstrate tracking with a light source. Also, things are ready to put in place with the Industrial Tech and Art classes.