The Light Follower Circuit

Written by Walker Arce

The general theory behind this project is this: the CEENBoT will be controlled by a photoresistive circuit that uses a flashlight as its 'controller'. If the extension is completed then the theory is this: the photoresistive circuit will be controlled by an LED attached to an RC servo that interfaces with the PS2 controller to control the servo and thus control the movement of the BoT.

Before starting on this project make certain that you have completed the *CEENBoT-API: Getting Started Guide (Atmel Studio 6)* and *CEENBoT-API: Programming Fundamentals*, both by Mr. Jose Santos and available for download at his site[1]. This will give you the tools needed to begin programming the CEENBoT effectively. It would also be of interest to someone working on this project to have the *CEENBoT-API: Programmer's Reference*, which can also be found on Mr. Santos's website.

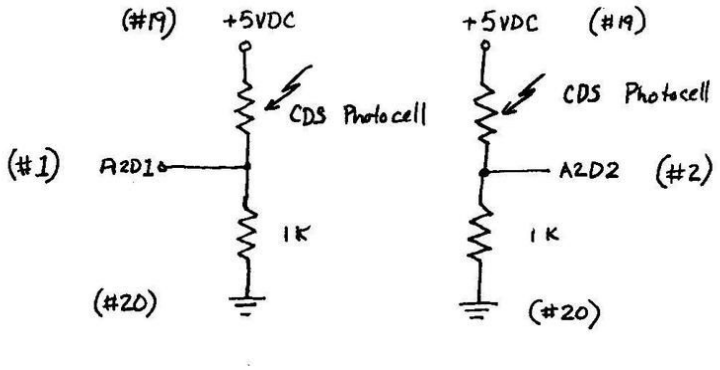
This project for the CEENBoT is done in three parts, or four if you decide to extend the functionality of the program. First, let's go over what is needed for the circuit:

- Two CDS photocells
- Two 1,000 Ω resistors[2]
- Six jumper wires

Once you have those together, the circuit will need to be assembled on the CEENBoT. To the right of the LCD display is a twenty-pin J9 header that will be used for connecting to the ADC channels. Below is a chart that shows the pinout for that header to be used as reference when setting up the circuit:

19	17	15	13	11	9	7	5	3	1
20	18	16	14	12	10	8	6	4	2

(The diagram for the circuit is shown below)[3]



So, to put together the circuit:

1. Connect pin 19 (VCC) to the power bus of the breadboard and connect pin 20 (GND) to the ground power bus of the breadboard.
2. Set up the photocells so that they are spaced evenly away from one another on either side.
3. For the left photocell, wire a jumper from pin 1 (ADC_CHAN3) to the hole below the left side pin of the photocell.
4. For the right photocell, wire a jumper from pin 2 (ADC_CHAN4) to the hole below the right side pin of the photocell.

5. For both photocells, wire one resistor from the hole below the jumper for the ADC channel to ground so that on the far pin of each photocell, there is a jumper wire to the J3 header pin and below that is the resistor connected to ground.
6. Finally, for the left photocell, wire a jumper from the power bus to the pin above the right pin of the photocell.
7. For the right photocell, wire a jumper from the power bus to the pin above the left pin of the photocell.

First Program:

For the first part of this activity, a program will be written that will test if the physical circuit is working. To do this, start with the template that was made in AVR Studio 6 and begin by opening the ADC submodule of the CEENBoT and running two sample variables, one for the left and one for the right, and then printing those to the LCD screen. It is also required that the voltage is set before hand and that the correct channel is set when working with both the left and right ADC channels.

A snippet of code is included below to get started:

```
/*
 * Light_Follower_Test_Circuit
 * DESC: This program tests the light follower circuit as described in
 *       associated documentation, it is meant as a tool to debug the physical
 *      circuit that is built on the CEENBoT.
 * Created: 3/30/2016 3:32:40 PM
 * Author: Walker Arce
 */

void CBOT_main( void )
{
	//Set the data direction of I/O PORTA
	DDRA = 0x00;

	//Set the open status for the ADC
	SUBSYS_STATUS ops_adc;

	//Initialize the ADC module
	ops_adc = ADC_open();

	//Open the LCD.
	LCD_open();

	while(1)
	{
		if ( ops_adc == SUBSYS_OPEN )
		{
			//Create sample variables
			ADC_SAMPLE saml;
			ADC_SAMPLE samr;

			//Set the voltage reference first so VREF=5V
			ADC_set_VREF(ADC_VREF_AVCC);

			//Set the channel that will be sampled
			ADC_set_channel(ADC_CHAN3);

			//Sample it
			saml = ADC_sample();

			//Finish the rest…
		}
```

```
        }
}
```

Second Program:

Once that is completed, the physical limits of the circuit can be found by first covering the photocell with your finger and then shining a light on it. This will give a physical maximum and minimum, which is likely to be different between the two photocells. To do this, modify the program so that when the values are being read, it is setting the values into both a maximum and minimum variable. Initialize the variable at the value 512 so that a middle point can be used as reference.

Write the program so that when the ADC sample is found, it is stored into a variable that is tested against the preset maximums and minimums and so that when you test both extremes, both values are displayed on the LCD display.

Take these two numbers and then perform the following

The code below is included to help get started:

```
/*
 * Value_Range_Test
 * DESC: This program is meant to allow the user to test and find the minimum
 *       and maximum values that are produced on the photocell resistors.
 * Created: 3/30/2016 3:32:40 PM
 *  Author: Walker Arce
 */

#include "capi324v221.h"

void CBOT_main( void )
{
       int MAXL = 512;
       int MINL = 512;
       int MAXR = 512;
       int MINR = 512;

       //Set the data direction of I/O PORTA
       DDRA = 0x00;

       //Set the open status for the ADC
       SUBSYS_STATUS ops_adc;

       //Initialize the ADC module
       ops_adc = ADC_open();

       //Open the LCD.
       LCD_open();

       while(1)
       {
              if ( ops_adc == SUBSYS_OPEN )
              {
                     //Perform ADC sampling here…

                     if ( saml > MAXL )
                     {
                            MAXL = saml;
                     }
                     if ( saml < MINL )
                     {
                            MINL = saml;
                     }
```

```
                        //Finish the code here…
                }
        }
}
```

Program Three:

For this program the robot will be made to move using the values found in the previous experiment. For the motor move function, you will need to set the motor running mode, which will be 'freerunning', the motor direction will need to be set to 'forward', and the acceleration will need to be set. All of the specifics of that can be found in the *CEENBoT-API: Programmer's Reference*.

Once that is completed, make sure that the ADC sampling is set into two separate functions for left and right so that there is no confusion and then invoke them in the `CBOT_main()` function so that they set the left and right values when they return. Then perform the conversion from the second program on the ADC values and from there, port the values to the LCD screen and the `motor_move()` function to begin the movement on the CEENBoT. Finally, put this into a `while(1)` loop so that it continuously does this.

The code below is meant as reference to help with getting started:

```
/*
 * Light_Follower_Final
 * DESC: This program is the culmination of the previous exercises and
 *       it allows the CEENBoT to move in accordance with the photoresistors.
 * Created: 3/30/2016
 * Author: Walker Arce
 */

#include "capi324v221.h"
//---------------------
void CBOT_main( void );
ADC_SAMPLE check_ADCl( void );
ADC_SAMPLE check_ADCr( void );
void motor_move( ADC_SAMPLE left, ADC_SAMPLE right );
//---------------------
void CBOT_main( void )
{
       //Set the data direction of I/O PORTA
       unsigned char r16 = 0x00;
       DDRA = r16;

       //Set the open status for the ADC
       SUBSYS_STATUS ops_adc, ops_stepper;

       //Initialize the ADC module
       ops_adc = ADC_open();

       //Open the LCD.
       LCD_open();

       //Set the open status for the stepper motors
       ops_stepper = STEPPER_open();

       //Finish CBOT_main().
}//End CBOT_main()

//Put your ADC channel sampling in separate functions for both left and right.
```

```
void motor_move( ADC_SAMPLE left, ADC_SAMPLE right )
{
	//Set the motors into 'freerunning' mode, set their direction as
	//'forward', and set the acceleration to ~300.

	//Set the velocity with the found values
	STEPPER_set_speed2( left, right );

	//Return nothing
	return;
} //End motor_move()
```

So the completed program should react to a flashlight being shone on the photoresistors and have the appropriate motor move in accordance with how strong the light is shining.

Fourth Program:

For this program, two new peripherals will be interfaced with the CEENBoT Light Follower Program: a micro RC Servo and the PS2 controller. To do this, the PSXC and ATTINY[4] modules need to be correctly opened and two buttons need to be programmed to change the value on the servo as well as have a third button that will center justify the servo. In general, if you use a servo that has not been tested, the physical limits of the servo will need to be found experimentally. There are currently two tested RC servos that can be used, a Parallax servo and the Radioshack Micro RC Servo:

- Parallax: 400 to 2100
- Radioshack: 700 to 1500

The use of the servo is so that you can attach an LED light to the end of the arm of the servo so that when it is placed behind the breadboard on the BoT, the light shines on the photoresistors. So, in addition to constantly running through the ADC channel testing, the program needs to be constantly checking the value input from the PS2 controller so that the RC servo is moved *before* the ADC channels are checked.

It might also be in the interest of functionality to create a 'backward' function so that when a particular button is pressed, the BoT moves in reverse and then turns away from the obstruction.

The following code is used to show the functions that could be used for this program:

```
/*
 * Light_Follower_Final
 * DESC: This program is an extra addition to the Light Follower program that
 *       adds an RC servo that is controlled by the PS2 controller which shines
 *       an LED light onto the photoresistors, which controls movement.  Added
 *       are also enhanced visuals on the LCD.
 * Created: 3/30/2016
 * Author: Walker Arce
 */

#include "capi324v221.h"
#include "tmrsrvc324v221.h"

//===[PROTOTYPES]===
void CBOT_main( void );
ADC_SAMPLE check_ADCl( void );
ADC_SAMPLE check_ADCr( void );
void controller_in( void );
ADC_SAMPLE algol( ADC_SAMPLE left );
ADC_SAMPLE algor( ADC_SAMPLE right );
void motor_move( ADC_SAMPLE left, ADC_SAMPLE right );
void motor_back( unsigned char backl, unsigned char backr );
//===[PROTOTYPES]===
```

That is the Light Follower program, when it is completed, the program should make it so that the BoT movement is controlled by the controller, which moves the LED on the servo so the motors move to turn the BoT and also have the BoT move forward.

References:

[1]www.digital-brain.info/ceenbot/

[2]*Note about resistors: It doesn't matter what resistance you use for the circuit per se, as long as they are the same resistance. If they aren't the same, then the values will be drastically different and your final program won't work correctly.*

[3]Photo credit: http://educationalrobotics.wikispaces.com/Flashlight+Control+Project

[4]The ATTINY subsystem module controls the system peripherals such as the RC Servos, IR sensors, and switches.