Photoresistor, Transistor, and LED's

Learning Objectives:

By the end of this laboratory experiment, the experimenter should be able to:

- Explain how a photoresistor works
- Describe the voltage-current relationship for an LED
- Describe how to capture analog voltage samples with an ADC using the Arduino library
- Build a circuit that includes an LED, photoresistor, and transistor and interface the circuit to a microcontroller to create a light controlled switch
- Write a program for the Arduino to control and modify the functionality of a light controlled switch
- Develop a light-tracking device with Arduino, a servo and photoresistor.

Components:

Qty.	Item
2	10 kΩ resistor
1	Solderless Breadboard
1	470 Ω resistor
1	Photoresistor
1	220 to 270 Ω resistor
1	red or green LED
1	Arduino Microcontroller Board and USB Cable
1	hobby servo with attached photoresistor board

Introduction:

A photoresistor is simply a resistor whose resistance depends on the amount of light that falls upon it. Photoresistors are used to make light-sensitive devices, and are often made from cadmium sulfide (CdS). The resistance of a CdS photoresistor varies inversely to the amount of light incident upon it. In other words, its resistance will be higher at low light levels (in the dark) and lower at high light levels. Two advantages of photoresistors are that they respond to wavelengths of light similar to those of the human eye, and that they are very easy to interface. A disadvantage is that they are a relatively slow-to-respond light changes compared to other light-detecting devices, such as photodiodes.

A light emitting diode (LED) behaves like an ordinary diode except that when it is forward biased, it emits light. The forward voltage drop for an LED is higher than that of a standard diode. Typical LEDs (the two-wire leaded "jelly bean" type) require 5 to 15 mA to reach full brightness, but are not designed to handle more than about 20 mA of current[1]. You will therefore ***always*** need to provide a resistor in series with an LED to limit the current to about 20 mA or less, or else you will burn it out. Also, do not make the mistake of trying to substitute an LED where a standard diode is called for! Examine the schematic diagram closely to see which kind

[1]. However other LEDs, specifically HBLEDs (high-brightness LEDs) are designed for lighting applications can handle upwards of 1.0-1.5A or more.

of diode is needed. The schematic symbol for an LED is the almost the same as for a standard diode, but with the addition of two lightning-bolt arrows pointing outwards signifying the emission of light.

LEDs come in many different colors depending on the particular compounds with which they are manufactured. The most common (because they are the easiest and cheapest to make) are red LEDs, followed by green and yellow. Outside of these three colors, LEDs are significantly more expensive. Blue LEDs are important because they're used in the manufacture of white LEDs, because the latter doesn't actually exist. White LEDs are blue LEDs that emit a frequency of light that is used to excite a phosphor that is coated on the LED that, in turn, emits the white light. The use of white LEDs has increased substantially in the last decade, because they are energy-efficient in general lighting applications.

Procedure

1. Use the DMM to measure the photoresistor's resistance in the ambient light of the lab. How stable is its value? After recording this value, repeat and record the measurement while covering the cell with your hand. These values at these two extremes will be used in calculations later on.
2. To investigate the voltage-current behavior of the LED, construct the circuit shown in Figure 1. Measure the actual resistance of the 470 Ω resistor, and record your reading. Vary the supply voltage from 1 to 5 volts in 1-volt increments. **At each voltage, measure the voltage across the LED and the 470 Ω resistor using the DMM, and enter the values into the following table.** (**Lab Report Q1**) **The LED current can be calculated by applying Ohm's law across the resistor.** A similar table should be entered into the lab report (**Lab Report Q2**) with all voltage values and comments. Remember Ohm's Law for calculating the current through a resistor: $I = V/R$, where V is the voltage *across* the resistor.

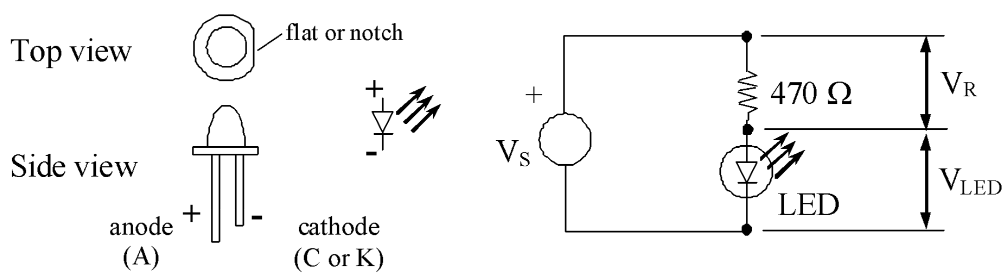

Figure 1. LED and typical circuit. Note that the anode lead is longer than the cathode. Sometimes there is a flat on the cathode side of the LED to help you distinguish anode from cathode. With voltage sources above the maximum forward voltage of the LED, ***you must always use a resistor in series to limit the current through the LED***.

Table 1. LED circuit measurements (Refer to Figure 1)

V_S, V	V_{LED}, V	V_R, V	Current, mA	Comment on LED brightness
1				
2				
3				

4				
5				

Figure 2 shows a simple light-controlled LED circuit. The circuit should turn the LED off as the photoresistor is covered. (**Lab Report Q3**) **<u>Explain the theory of operation of this circuit</u>**. Based on the information obtained above, what is a good supply voltage to use (**Lab Report Q4**)? (Hint: V should be high enough so that enough current flows through the LED when the photoresistor has low resistance, and yet should be low enough so that the current is not enough to turn on the LED when the photoresistor has high resistance.) **Build the circuit in Figure 2, and check its function.** (**Lab Report Q5**) **<u>Describe its operation.</u>**

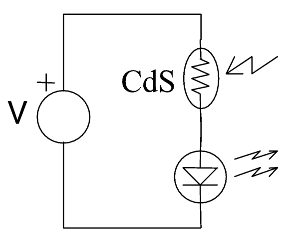

Figure 2. Light-controlled LED

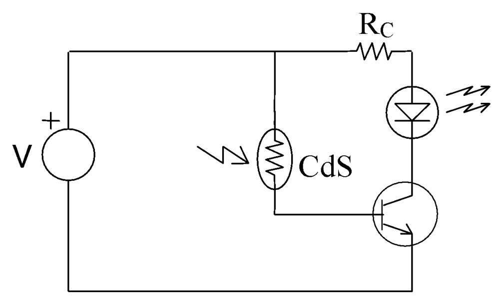

Figure 3. Light-controlled LED using a transistor "switch"

<u>The Light-Controlled Switch Using a Transistor</u>

A transistor can be added to the light-controlled-LED circuit to improve its sensitivity and to eliminate the 'half-on-half-off' state of the LED. A rudimentary circuit to do so is shown in **Figure 3 (you do *not* have to build this circuit).** Here the photoresistor controls the amount of current flowing into the base of the transistor, which in turn controls the collector current of the transistor, thus controlling the current through the LED. Unfortunately, this circuit may not function properly, because when the photoresistor is in the dark state, and the LED is supposed to be turned off, the base current may still be large enough for the LED to remain lit. (**Lab Report Q6**) **<u>Prove this in your report, by calculating the collector current for the circuit in Figure 3 when V=10 V, R_{CdS}=100 kΩ, R_c=220 Ω and h_{fe}=100.</u>**

Figure 4 shows an improved circuit. This is the circuit that you will build and experiment with next.

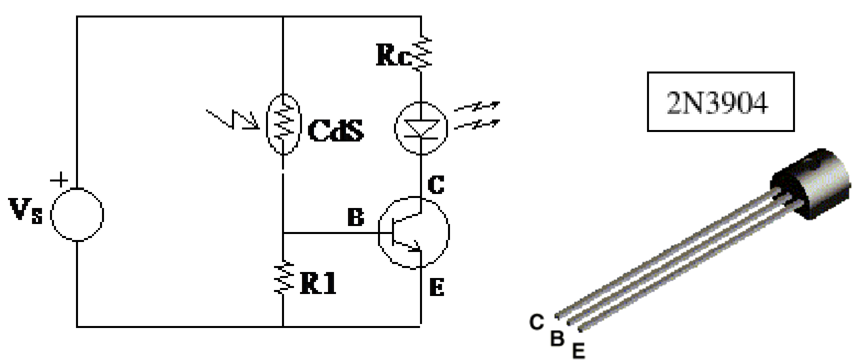

Figure 4. Improved light-controlled switch using a 2N3904 transistor. With the flat side of the transistor facing you, the pins from left to right are: emitter, base, and collector.

With a properly selected resistor R_1, the voltage at the base of the transistor in the dark state is less than 0.7 V, putting the transistor in its cutoff state. Since the transistor is in cutoff, no current flows from its collector to its emitter, keeping the LED off. As the photoresistor's resistance decreases due to increasing light intensity, the voltage at the base increases due the voltage divider formed by R1 and the photoresistor. Once the base voltage reaches 0.7 V, the base current starts to flow, and any further decrease in the photoresistor's resistance causes an increase of base current. This base current increase will be amplified by the current gain of the transistor up to the point that the transistor saturates.

Procedure

The following procedure explains how to select the resistance values for R1 and R_C in Figure 4. **(Lab Report Q7) Document with calculations in your report how you arrived at the resistance values that you ultimately used to build the light-controlled transistor switch in Figure 4 using the procedure below.**

1. (**Lab Report Q8**) **Choose the supply voltage, V_S**. The supply voltage is often the first design choice made in a project. For example, if the supply is chosen to be five alkaline batteries in series, then V_S = 5 * 1.5V = 7.5V.

2. (**Lab Report Q9**) **Select R1**. First, choose the value of the photoresistor's on-resistance (call it R_{on}) within the range that you measured earlier at which you would like the LED to be turned on. It is recommended that you use the resistance that you measured with the photoresistor uncovered. The value of R1 should be such that $\mathbf{0.7V = V_S*R1/(R1+R_{on})}$.

   Choose a nominal resistor value closest to the value you calculated for R1, or you can use a variable resistor (potentiometer/trim-pot). Depending on the resistor value you choose, you may want to check the equation above for R_{ON} to make sure it's still within the range that you want.

3. (**Lab Report Q10**) **Select R_c**, the current limiting resistor. With this resistor, the collector current is limited to $\mathbf{I_{max} = (V_S - V_{LED} - V_{Sat})/R_C}$, where V_{LED} is the forward voltage drop across the LED (measured above) when it's on, and V_{Sat} is the saturation voltage across the transistor's collector and emitter. A reasonable value for V_{Sat} for the 2N3904 transistor is about 0.4 V. *Select R_C so that the LED current is limited to be less than 20 mA (preferably 10-15 mA).* You may need to iterate on this calculation depending on the limited choice of resistor values that are available. Using R_c, **construct and test the circuit**.

Using the Arduino to Make a Programmable Light-Controlled Switch

The circuit in Figure 4 is very simple, but it suffers from the limitation that once R1 is chosen and the circuit is constructed, you are stuck with its performance unless you physically remove R1 and replace it with a different value. That is not a serious issue if you are dealing with one circuit on a breadboard, but suppose this circuit were part of a product that you were manufacturing by the thousands (or hopefully millions!). If you wanted to tune the performance of the device, you would have to modify the assembly drawings, bill of materials, perhaps the circuit board artwork, component inventory, rework the entire work-in-process, etc. All of this would be a huge deal in a commercial setting!

Here we will use the Arduino/ATmega328 to make a light-controlled switch whose performance can be modified by simply *reprogramming* the microcontroller.

Procedure

1. Build the part of the circuit in shown in Figure 5 below labeled A. We are going to use the Arduino to supply power to this circuit. However, in general, do not assume that you can always power circuits from just the power regulator on the Arduino board without doing some calculations regarding the current requirements of the circuit(s) that you are connecting! Loads that draw more than 30 mA per pin, or more than a few hundred milliamps from the Arduino's power regulator should be powered instead by an external supply that shares a common ground with the Arduino. (**Lab Report Q11**) **Considering the circuit labeled "A" in the figure, how should pin D9 be configured in setup(), as an INPUT or OUTPUT?**

Connect the 10k resistor to pin D9 on the Arduino board using a jumper from the pin to the solderless breadboard. Also, you will need to take both power (5V) and ground (GND) from the Arduino using the appropriate pins from its (bottom-left 6-pin) POWER header. The voltage on Pin D9 will either be 0 or 5 V. (**Lab Report Q12**) **Which of these voltages will turn the transistor on? When the transistor turns on, is it saturated? Based on your earlier measurements on the LED and the characteristics of the transistor, what are i_b, and i_c when pin D9 is at 5 V?**

2. Create a sketch using the code in Appendix A. Check to see if the LED is blinking at 1Hz. (**Lab Report Q13A**) Examine how the program works. (**Lab Report Q13B**) **Explain how the output on Pin D9 is toggled by using the C-language '^=' operator. Also, explain the use of the "static" keyword with the ledIsOn variable. Is there another way to accomplish what the static variable does here?**

3. When you have *successfully* completed step 2, build the circuit in B shown in Figure 5. Modify the `AnalogReadSerial` example code provided with the Arduino IDE to read the input voltage on Pin A0, and monitor this value using the Serial Monitor. (**Lab Report Q14**) **What happens when you cover the photoresistor? What range of values are shown in the communications window? What voltages do these correspond to?**

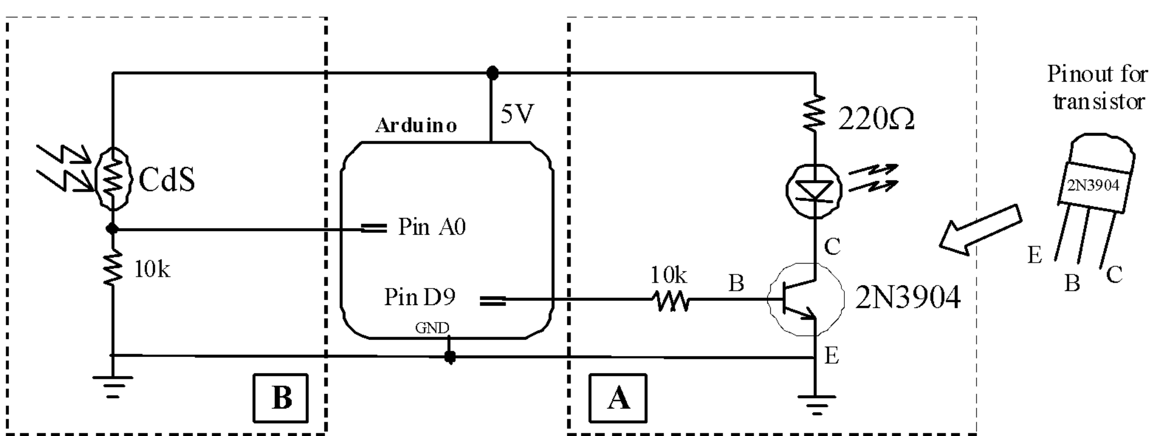

Figure 5. Light-controlled switch using the Arduino. The photoresistor (in B) is part of a voltage divider, the output of which is connected to pin A0 (Analog 0). Arduino D9 (Digital Pin 9) is connected to the base of a transistor (in A), which is used as a switch for illuminating the LED when the light level on the photoresistor is beyond a certain threshold value.

4. After you have *successfully* completed step 3, write a program that will turn the LED on when you cover the photoresistor with your hand. Hint: It is best/easiest to choose a threshold value above which the LED is in one state, and below which the output is reversed. The upper and lower `analogRead()` values that you collected in the previous steps should be used to

determine your threshold value. (**Lab Report Q15A)** Include the code with your threshold value clearly documented.

(**Lab Report Q15B)** **What changes need to be made to the software (note: no need to change any *hardware*) if you want to have the LED stay ON under ambient light conditions and turn OFF when a shadow falls on the photoresistor (i.e. the opposite function to what you programmed in step 4)?**

5. Write a program called *photo_blinkie* that blinks the LED, where the LED blinks faster when the photoresistor is covered more completely, and blinks slower when more light falls on the photoresistor. Hint: You might consider using the `map()` function (again) for this exercise. (**Lab Report Q16)** **Include the completed code in your lab report.**

6. Detach the breadboard circuit from your Arduino and attach the servo-photoresistor combo device to your Arduino. The interface to this setup is four wires, red and black for 5V and GND, white for servo control and yellow for photoresistor sensing. Using all of the testing procedures that you employed above (upper and lower analog light sensor range determination), write a program that tracks a light source. If it's a daytime lab you can use the light from the windows as the light source. If not, the overhead lights may be used, or the light from the display of your phone/tablet with acceptable results. Write a program to track the light source by tracking it with the photoresistor and the servo. **Demonstrate your working light-tracking system to your TA and** . (**Lab Report Q17)** **include your final code in your lab report.**

Appendix A – Blink Program – starting point for photoresistor night-light sketch

```
/*
 * Blink LED at 1000/(DELAY_TIME_MSECS * 2) Hz.
 * This code assumes that the LED is active-high.
 */

#define PORTB_LED_MASK 0b00000010 // LED is on PB1 (Arduino D9)

#define DELAY_TIME_MSECS 500      // msecs between toggles

/*
 * initializations - just setup the one LED in this case.
 */
void setup()
{
  DDRB  = PORTB_LED_MASK;     // LED on PORTB is an output
  PORTB = 0x00;               // set initial LED state
}

void loop()
{
  static byte ledIsOn=0;

  if (ledIsOn) {
    PORTB &= ~PORTB_LED_MASK;   // LED -> off
  }
  else {
    PORTB |= PORTB_LED_MASK;    // LED -> on
  }
```

```
    ledIsOn ^= 1;                   // toggle ledIsOn
    delay(DELAY_TIME_MSECS);
  }
```