The Nuts and Volts of BASIC Stamps

Parallax, Inc.
www.parallaxinc.com

Nuts and Volts
www.nutsvolts.com

Column #73, May 2001 by Jon Williams:

Lookin' For The Light

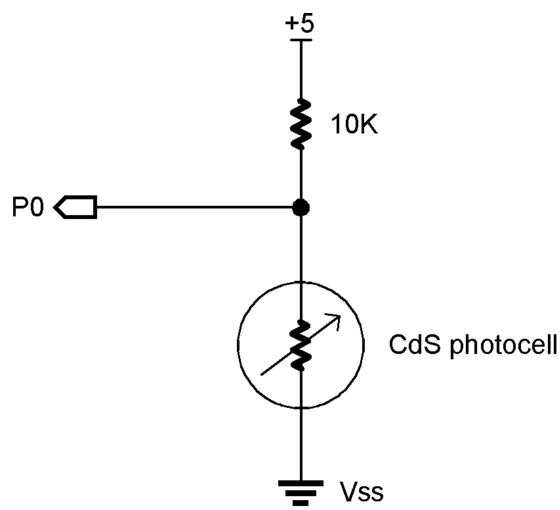
The kinds of sensors we could connect to the BASIC Stamp are as varied and interesting as the people programming them. That said, light sensors seem to be gaining in popularity or, perhaps, I've just noticed. So this month we're going to have a bit of fun with a couple of types of light sensors. Nothing earth-shaking or human-race-saving (as if we've ever done that...), just good clean experimental fun.

This column is geared a bit toward beginners, although you old dogs may learn a few new tricks too. To that end, you'll need very little: a BS2, a breadboard and a few common parts. There will be no long code listings to type in this month; just fragments to play with. I'll leave it to your intelligence and imagination to incorporate these sensors and code fragments into your projects. Let's rock....

The CdS Photocell

First up is the ubiquitous Cadmium Sulfide (CdS) photocell. They're very easy to get – in fact, you can buy a pack of five at Radio Shack for about two dollars. From the Stamp's point-of-view, the CdS photocell looks like a resistor that changes its value in response to light. The photocell resistance is inversely proportional to the light falling on it; as the light gets brighter the resistance gets smaller.

Figure 73.1: CdS photocell circuit with BASIC Stamp



Okay, let's do something with it. Our first experiment will be a bright light detector that will use the CdS photocell and a 10K resistor. Figure 73.1 shows the schematic.

Since the CdS photocell is, essentially, a resistor, we can combine it with another to create a voltage divider. The "output" of the voltage divider will depend on the amount of light striking the CdS photocell.

With my Radio Shack units, I measured the resistance at about 1 Meg when dark, about 10K in the ambient light of my office and down to about 200 ohms with bright light shining on them. Let's see how this works when we connect the circuit to the Stamp.

What we must first understand is that the Stamp is a little more flexible in its definition of zero and one than we are. For inputs, we think of zero being zero volts (ground) and one being five volts. The Stamp has a logic threshold of about 1.5 volts. This means that anything from ground to about 1.5 volts will be considered a 0; anything from about 1.5 to 5 volts will be considered a one. By applying Ohm's Law, we can see how the circuit works:

Dark	(5 volts x 1 Meg) / (1 Meg + 10K) = 4.95 volts
Ambient	(5 volts x 10K) / (10K + 10K) = 2.5 volts
Bright	(5 volts x 200 ohms) / (200 ohms + 10K) = 0.1 volts

By these calculations, we should see a one when the photocell is dark or exposed to ambient light. When a bright light is shined on the cell, we should see the input change to a zero.

Okay, then, here's a bit of code to find out:

```
CdS     VAR     In0

Loop:
  DEBUG Home,BIN1 CdS
  PAUSE 100
  GOTO Loop
```

Indeed, it does work.

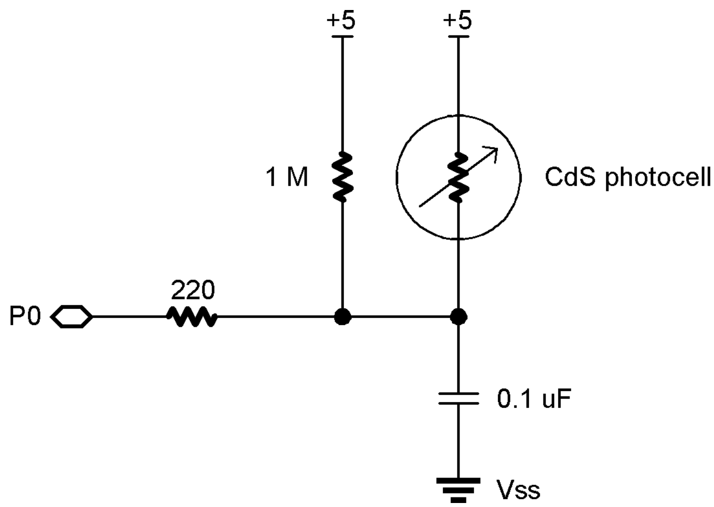
About now you may be thinking, "That's great, Jon, but what do I do with it?" Well, take a look around. Light/dark sensors are used everywhere: security systems, emergency lighting control, robotics. On a diet? How about building a talking refrigerator alarm that reminds you of your diet goals when the refrigerator door is opened? (This assumes that the light in your refrigerator works...)

We can "tune" this simple circuit by changing the value of the fixed resistor. By increasing the value of the fixed resistor we will reduce the amount of light needed to cause a "0" output. If we changed it to 50K, for example, ambient light would read as a zero. Finally, we could install a 50K potentiometer in place of the fixed resistor to create a tunable circuit.

Or...we could choose to create tunable software. To do this, we need to read the photocell as an analog value. Knowing that the CdS photocell behaves like a resistor, we'll use RCTIME and a couple of common parts to read its value.

To use RCTIME, we need to modify the circuit as shown in Figure 73.2.

Figure 73.2: Resistor/capacitor circuit with photocell



The astute among you (That's everyone, right?) have probably checked out RCTIME in the Parallax manual and are thinking to yourself, "Hey, dude, that is NOT the recommended circuit for RCTIME! And you added a resistor...." You're right, dude. Here's the story:

The RCTIME function measures the time it takes for the capacitor to discharge. In the "recommended" configuration, the voltage at the Stamp-side of the capacitor will have to swing from 5 volts (discharged) through 1.5 volts; a span of 3.5 volts.

We're using the "other" configuration. In this circuit, the capacitor voltage will swing from 0 to 1.5 before RCTIME ends; a span of 1.5 volts. So you can see that the "recommended" circuit will give us better resolution.

The problem for us, though, is that the dark value of the CdS photocell is so large that RCTIME might return 0 (due to roll-over). This would, of course, be erroneous. By using the resolution-reduced circuit and reducing the maximum value of the CdS (by added a parallel resistor) we can measure the photocell without problems. On my breadboard I saw readings from 0 (bright light) to about 17,000 (dark).

Here's the code:

```
CdS      CON      0
```

```
light   VAR     Word

Loop:
 LOW CdS
 PAUSE 1
 RCTIME CdS,0,light
 DEBUG Home,DEC5 light,CR
 PAUSE 100
 GOTO Loop
```

When you run this program you'll see the value bounce around quite a bit. We can smooth out the bouncing by increasing the PAUSE between readings or we could digitally filter it. A simple method of applying a digital filter is to add a portion of the last reading to a portion of the current reading. The ratio of last-to-first values will determine the behavior of the filter. The larger the last reading portion, the slower the light reading will climb toward the current value. We can use this technique to filter out glitches (so our sunrise detector isn't affected by a passing police car).

Try this:

```
CdS     CON     0

light   VAR     Word
last    VAR     Word

Loop:
  LOW CdS
  PAUSE 1
  RCTIME CdS,0,light
  light = (light */ $40) + (last */ $C0)
  last = light
  DEBUG Home,DEC5 light,CR
  PAUSE 100
  GOTO Loop
```

In this version we slow the response of the photocell by taking 25% of the current reading (light */ $40) and adding it to 75% of the last reading (last */ $C0). Experiment with different ratios (make sure that they add up to 100%) to see how the circuit responds. What you'll see is that

Okay, let's wrap-up the CdS photocell. We've seen that it's easy to use with just few parts and built-in PBASIC functions. For you hardcore types, you could insert the CdS in a 555 timer circuit and read the value using PULSIN or COUNT (see below). Or you could connect the first circuit to an ADC and read the value that way. Do whatever works best for you – just keep the goal in sight and don't let the method get in the way of results.

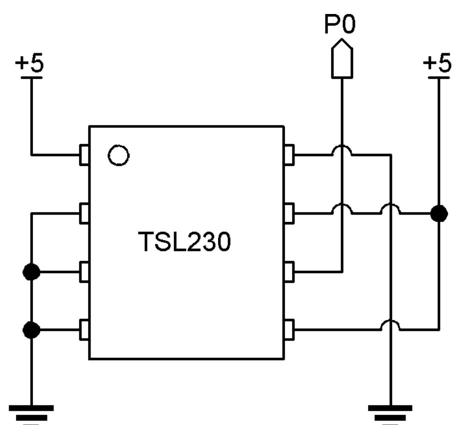
The TAOS TSL230

At a recent DPRG (Dallas Personal Robotics Group), one of our members shared a new light sensor from Dallas-based Texas Advanced Optoelectronic Solutions (TAOS). The TSL230 looks like any other 8-pin PDIP, except that the body is clear.

This neat little sensor combines a configurable silicon photodiode and a current-to-frequency converter on the chip. The output is a pulse train with frequency directly proportional to light intensity.

Connecting the TSL230 to the Stamp is very easy. Figure 73.3 shows the basic connections.

Figure 73.3: TSL230 hookup to BASIC Stamp



I wired-up the circuit on an INEX-1000 breadboard and used this bit of code to test the output of the TSL230.

```
TSL230  CON     0

light   VAR     Word

Loop:
  COUNT TSL230,100,light
```

```
DEBUG Home,DEC5 light,CR
PAUSE 100
GOTO Loop
```

The output of the TSL230 is a pulse stream so the code uses COUNT (with a 100 ms window) to read the light value. In this case, the value is directly proportional to the amount of light falling on the sensor (more light = bigger values).

You may be tempted to use PULSIN to get a quicker measurement. You can do this when there is sufficient light falling on the sensor. At low light levels, the time between pulses exceeds the PULSIN window parameters and you could get erroneous readings.

Aside from being very easy to use, the TSL230 has a couple of neat features:

Programmable Sensitivity
Scalable Output Frequency

The connections to Pins 1 and 2 (S0 and S1) control the sensitivity of the photodiode. These inputs are TTL compatible so advanced applications could actually control the TSL230 sensitivity on-the-fly.

Pins 7 and 8 (S2 and S3) control the output scaling. When the output frequency is scaled by 2, 10 or 100 it will appear as a square wave. When not scaled, the low-going portion of the output pulse train carries the light value. Like S0 and S1, these pins could be connected to the Stamp to dynamically control the output scaling. Why? The idea is to allow the best possible resolution for a given processor speed for a given light value. Go ahead and play with it. You'll learn a lot in the process.

Be sure to download the TSL230 specs from TOAS.

Hopefully you found this month a nice break from the big projects we usually do. I believe it's good to experiment with new parts, even if there is no specific goal in mind at the time. Play time doesn't always bring immediate results, but I promise, it brings results. I have pulled myself out of many project holes with information gleaned from play-time experiments.

Let's Sound Off Again

Of the 20 or so articles I've written for Nuts & Volts, the speech and sound articles (SP0256-AL2 and ISD2560) have been the most popular. Clearly, speech and sound are hot topics for Stamp uses. Next month I'll have some really neat stuff to share. See you then.

Texas Advanced Optoelectronic Solutions, Inc.
800 Jupiter Road, Suite 205
Plano, TX 75074
(972) 673-0759
www.taosinc.com