Experiment #5: Measuring an Input

As we've learned so far, each of the BASIC Stamp's 16 "real world" pins can be configured as either an input or an output. If the pin was configured as an input, there are 7 different instructions in the PBASIC language that can be used. Each of these commands is suited for specific types of input conditions.

For example, in Experiment #2 we learned how to use a command called "**input**". If this command is used in your program, it causes the specified pin to become an input (gee, that makes sense!). Then, anytime we wanted to check the status of the pin (whether it was "high" or "low") we used the statement **`if in2=0 then blink`**.

What's a...

7 different "input" instructions:

They are:

- Button
- Count
- Input
- Pulsin
- Rctime
- Serin
- Shiftin

We've already used "input" earlier. In this experiment, we'll be exploring "**Pulsin**"

This line of code "looked" at the pin, and returned a value. If the value on that pin was "0", then the code would branch off to another point in the program where it would blink the LED. If the value of the input was a "1" then it would continue program execution with the next line of code.

In any event, the values that could be detected with this code were binary - either a "1"or a "0". This is suitable for detecting whether or not a switch has been pushed (Exp. #2), or even to detect light or dark on a photocell, as we did in Experiment #4.

The PBASIC language has some other commands that offer a greater level of sophistication when it comes to detecting inputs. If you haven't already, download a free copy of the BASIC Stamp Manual from www.stampsinclass.com. In it you'll find a complete description and application information of all the commands available in the PBASIC language.

In this Experiment we're not only going to take an advanced look at "input detection", but we're also going to use a popular integrated circuit named the '555 timer.

Parts Required

For this experiment you will need the following:

(1) BASIC Stamp II
(1)"Board of Education"
(1) Programming Cable
(1) LED
(1) CMOS 555 timer IC
(1) 10 microfarad, 25 volt electrolytic capacitor
(1) 1 microfarad, 25 volt electrolytic capacitor
(1) 470 ohm, ¼watt resistor
(1) 100K potentiometer (variable resistor)
(1) 15K ohm, ¼watt resistor
(1) 1 K ohm, ¼watt resistor
(1) 9 volt battery or wall transformer
(misc.) connecting wires
(1) Personal Computer running DOS 2.0 or greater, with an available serial port.
(1) BASIC Stamp Editor program

Sources for these materials are listed in Appendix A.

Build It!

This hardware circuit uses an Integrated Circuit called a "555 timer". The '555' is a actually a "bunch of electronic circuitry" (that used to fill up a large area on a printed circuit board) that has been miniaturized and encased into the little "8 pin dip" package that we're using today. Although it's a sophisticated array of circuits on the inside of the plastic case, the '555' is really quite simple to use for many different applications. In fact, in the many years since it's development, the '555' has been designed into an untold number and variety of devices because it can do so many different things. Although it's not "programmable" like the BASIC Stamp, the '555' can be configured, with different combinations of resistors and capacitors, to accomplish many different tasks.

What's a...

Integrated Circuit:

"IC's" as they're commonly called, are electronic circuits that have been miniaturized and combined into one small, convenient package. Many different types of IC's have been created for untold numbers of applications. The '555' timer that we're using in this experiment is a member of the "Linear" family of IC's. The Stamp's CPU is a "Digital" IC.

8 pin dip:

This refers to the package style of the IC. The '555' has 8 Pins, and these pins are arranged in a Dual Inline Package.

What's a...

Astable multivibrator:

A fancy name for a circuit that with "no outside intervention" (by other circuitry or devices), will continually output a stream of pulses. Remember when we created the pulse stream for the servo control in Experiments 3 & 4? Same thing here, except that the '555' will alternate high and low, without us having to write a program. It's a hardware version of the software "Pulsout" command.

An IC (**integrated circuit**) needs to have some sort of identification on its package to tell us where pin #1 is. The identifier is usually a notch or indentation located on one end of the plastic package. See Figure 5.1.

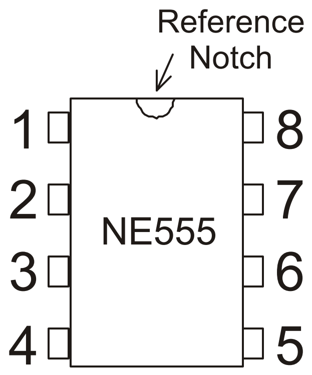


Figure 5.1: 555 Timer IC
Note the notch on one end of the chip package.

As with the BASIC Stamp, each pin on the '555' has a particular purpose. Although the '555' is fairly "bullet proof", connecting an improper electrical signal to the wrong pin can damage the device, so be careful and follow the diagrams closely.

The type of circuit that we're building here is called an "**astable multivibrator**". Don't be put off by its complicated name! All this really means is that the output of the '555' alternates from high to low. Recall that in Experiment #1 we used the 'high' and 'low' commands to blink an LED on and off. In reality, that's all that our '555' circuit is doing. It's just "oscillating" on and off. The '555' circuit that we're building, is the hardware equivalent of Experiment #1.

The rate at which the output (on pin #3 of the '555') blinks is controlled by the values of a resistor and capacitor. As the values of these devices change, the "blink" rate of the '555' changes.

We've used resistors in the past. They control the amount of current flowing through a given circuit. Since we want to (conveniently) change the rate at which the '555' blinks, we're going to use a variable resistor, also known as a **potentiometer**. If you've ever adjusted the volume on a radio, you've used one. By turning the dial on the "pot", you change the value of the variable resistor. This in turn changes the rate at which the '555' blinks.

What's a...

Potentiometer:

A "pot" is just a resistor that changes its value as you manually rotate (or in some cases slide) its shaft or dial. Recall that resistors have two "leads" or connections. A pot has three connections. The center lead is connected to a wiper that "wipes" across a resistive element. The closer the wiper gets to either end of the element, the lower the resistance between the wiper and the end it's approaching. Pots come in many different values, such as 5K, 10k, 100k, 1 meg ohms, and more. They also come in many different physical configurations to accommodate different product designs. But they all operate essentially the same way – mechanical movement of the wiper element changes the resistive value of the device.

As you're connecting the potentiometer, you'll notice that there are three terminals or connections available. One of these is the "wiper" and the other two are the ends of the resistive element. We only need to connect one end and the wiper contact to our circuit, as shown in Figure 5.2.

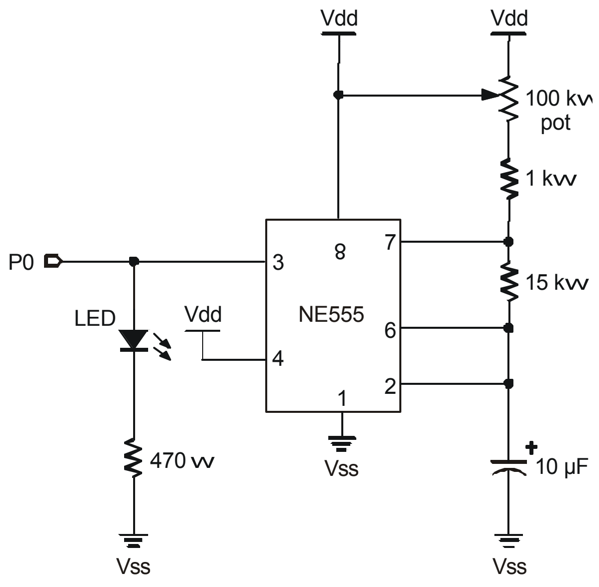


Figure 5.2: 555 Timer Schematic
Schematic for Experiment #5 on the Board of Education.

FYI:

About the schematics:

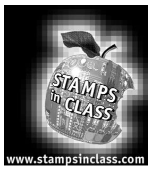
It is common practice on schematic diagrams to draw the pins of an IC wherever they make the diagram easiest to read.

When you insert the '555' timer IC into the breadboard area of the Board of Education, be sure to have the device "span" the "dividing trench", so that the pins are not shorted together. Once you've completed the circuit shown in Figure 5.2, go ahead and power up the Board of Education.

When you power up the Board of Education you should find your LED blinking. Turn the knob on the potentiometer. What happened? You're probably wondering why it's already working and you haven't even written a program! That's the beauty of using a 555 timer in your circuit – it doesn't necessarily need to communicate with the BASIC Stamp to operate on it's own.

You see – Pin #4 (on the '555') is a "**reset**" pin. It is an "input" to the '555' and as long as this pin sees a 'high' (1), the '555' will operate. In order for our circuit to work without interaction from the BASIC Stamp, we connected pin #4 (on the '555') directly to Vdd (high). This kept pin #4 in a high state, which allowed the '555' to blink the LED. Now, we are going to have the BASIC Stamp take control of the '555'.

Remove the power source from the Board of Education. Move the end of the wire from Vdd (Pin 4 on the 555) to P1 (on the BASIC Stamp) to give the BASIC Stamp control. Now we are going to write a program that controls the '555' from P1. Remember, when Pin 4 (on the '555') sees high, it will start working.



Program It!

Start the BASIC Stamp Editor. If you don't remember how to do this refer back to an earlier experiment. As mentioned above, we now want the BASIC Stamp to control the 555 circuit, so be sure that you've connected the 555's Pin 4 to BASIC Stamp P1.

Pin #4 (on the '555' is a "**reset**" pin. It is an "input" to the '555' and as long as this pin sees a 'high', the '555' will operate. In order for our circuit to work without interaction from the BASIC Stamp, we connected pin #4 (on the '555') directly to Vdd (high). This kept pin #4 in a high state, which allowed the '555' to blink the LED.

What's a...

Reset:
As mentioned, this is a control pin on the '555' timer IC. If we connect this pin to P0 on the Stamp, & P0 is configured as an *"input'*, then the '555' circuitry may in fact operate (although perhaps unreliably). In this situation we have two inputs (P0 on the Stamp and 'reset' on the '555') connected together. A pin configured as an input on the Stamp will tend to "float" high. This is a "floating condition, however and is not guaranteed to be a true "high". When we make P1 an output, and cause it to go "high", it does so, and *drives* pin 4 (on the '555") high, rather that just "floating" it up.

Type in the following program:

```
here:
high 1
pause 5000
low 1
pause 5000
goto here
```

Now while holding the "ALT" key down, type "r" (for "run") and press "enter".

What's happening, and why? If everything is working properly, you should see the LED blink on, off, on, off, etc. (for a period of 5 seconds) and then be completely off for 5 seconds. Then the program recycles and does it again.

Since P1 (on the BASIC Stamp) is connected to the reset pin on the '555', every time P1 goes 'high' it allows the '555' to blink the LED on and off. And whenever P1 goes low (under our program control), it shuts off the '555' circuit.

Ok you say, but so what?

Well, think about it this way. Microcontrollers are only capable of doing one thing at any one time. If we want to blink an LED on and off as a “warning indicator”, then while the BASIC Stamp is doing its “high - pause - low – pause - repeat” routine (to blink the LED), the BASIC Stamp is not able to do anything else.

Now, as shown in the following program, you can turn on the “LED blinker circuit” and continue on (in your program) and do something else “more important”. Try it.
‘

```
x var word
low 1
here:
high 1              ‘turn the blinker on
for x = 1 to 500
debug ? x           ‘count to 500 on the screen
next                ‘while the LED blinks
pause 3000
low 1               ‘turn the blinker off
pause 2000
goto here           ‘go back and do it again
```

What we’ve done here is “off-loaded” the task of actually blinking the LED (on, off, on, off, etc.) from the BASIC Stamp. The action of “blinking“ is accomplished by the ‘555’ timer circuit. All the BASIC Stamp needs to do is enable or disable the “blinker circuitry”. The BASIC Stamp can then go on and do other more important tasks. In this example, the “more important task is to count up to 500 and display the numbers on the screen. In the real world, however, you might be looking for other “input conditions” to be met (on some other pin on the BASIC Stamp).

What's a...
Microfarad:
A unit of measurement for the amount of “charge” that can be stored in a capacitor. Similar to the “ohm” value for resistors, the microfarad (for capacitors) is available in a wide range of values. 1 microfarad is equal to 1/1000000 of a farad. We’ll explore capacitors in an upcoming experiment, but for now, understand that the lower the value, the lower the charge that you can store on the capacitor, which results in faster oscillation of the 555 circuit.

In this circuit, so far, we’ve been using a 10 **microfarad** capacitor. Shut off the power and replace the 10 microfarad capacitor with a 1 microfarad cap. Be sure to observe the proper polarity on the capacitors. Go ahead & re-apply power.

By reducing the value of the capacitor (C1) we’ve increased the blink rate of the LED (in this case, 10 fold). Even though it may be difficult to see visually, the LED is still blinking, but at a much higher rate.

In Experiment #4 (where we controlled the rotation of a servo), we used a command called **`pulsout`**. Recall that **`pulsout`** generated a single pulse output of a length determined by one of the commands’ parameters.

For example, to create a pulse length of 1 millisecond (on P1), the command was: **`Pulsout 1, 500`**. The value of 500 is the number of two microsecond increments. Therefore 500 times 2 microseconds = 1000 to microseconds or one millisecond.

We're now going to use a new command called **pulsin**.

Pulsin is the input counterpart to **pulsout**. Rather than generating an output pulse of a predetermined length, **pulsin** looks at a particular input pin and measures the length of an incoming pulse and returns the length of that pulse in a variable.

Try the following program:

```
x var word

high 1

here:
pulsin 0,1,x
debug ? x
goto here
```

What's happening? Pin #3 of the '555' timer circuit is connected to the LED. The LED blinks at the rate determined by the value of the potentiometer (and the capacitor). We've also connected the output of the '555' to P0 on the BASIC Stamp.

x var word

This simply sets up a variable called "x", that is one word (or 16 bits) in size. This means that x can go as high as 65,536 (decimal).

high 1

This causes P1 to go high, which in turn (since its connected to the reset pin #4 on the '555'), allows it to oscillate.

here:

A label to jump back to . . .

pulsin 0,1,x

This single command tells the BASIC Stamp to:

Look at an incoming pulse on P1.

Wait for that pulse to go from low to high.

As soon as it does, start a "stopwatch", and continue to monitor the pin.

As soon as the pulse goes back to low, then stop the "stopwatch" and return a the value in a variable called "x". This value is in two microsecond increments.

`debug ? x`
This displays the value of "x" on the PC.

`goto here`
Do it again.

Now, try adjusting the value of the pot. What's happening to the value of "x"? Can you explain what's happening?

Whenever you change the value of the pot, the blink rate of the LED is changed. **`Pulsin`** can only measure up to a maximum of 131 milliseconds, that's why we increased the blink rate of the LED (by lowering the value of the capacitor). With the 10 microfarad capacitor, the blink rate was just too slow for **`pulsin`** to be able to measure it.

You can now actually measure the value of the blink rate of the LED. Take the value of "x" displayed on your screen, multiply it by two (remember that **`pulsin`** measures in 2 microsecond intervals) and you'll get the length (in microseconds) of each "blink".

The **`pulsin`** command is a significantly more advanced "input detector" command than a simple statement like "in" (or input), but they both have their appropriate uses – it just depends on the application.

Be sure and check out Appendix B and for a complete listing of PBASIC commands see the BASIC Stamp Manual Version 1.9. The Application Notes also show different ways to use the **`pulsin`** command.

Questions

1. What is a potentiometer, and what is one typically used for?

2. Why might we want to use a '555' timer to blink an LED instead of using the BASIC Stamp?

3. What does the **`pulsin`** command do?

4. What does the Reset pin on the '555' do, and how do you connect it to the BASIC Stamp?

5. Add appropriate remarks to the following program:

```
low 1              ____________________________________
here:              ____________________________________
high 1             ____________________________________
for x = 1 to 500   ____________________________________
debug ? x          ____________________________________
next               ____________________________________
pause 3000         ____________________________________
low 0              ____________________________________
pause 2000         ____________________________________
goto here          ____________________________________
```

Challenge!

1. Write a program (complete with remarks) that will allow the LED to blink whenever a switch is connected to P8. (You'll need to build this circuit in hardware – if you need a hint, refer to Experiment #2 on connecting a switch to an input pin).

2. Draw the schematic diagram of your circuitry from Challenge #1.

3. Replace the switch part of the circuit in Challenge #1, with the photosensor circuit from Experiment #4. Connect the photosensor to P10 and write a program that will enable the '555' blinker circuit for a period of 3 seconds, whenever the sensor "sees a shadow".

4. Write a program that will display (using debug) the measured value of **`pulsin`** and whenever the value falls below 10000, the reset pin is brought low, shutting off the blinking circuit.

What have I learned?

On the lines below, insert the appropriate words from the list on the left.

microfarads

output

volume level

simultaneously

values

.131 seconds

hardware

debug

variable

integrated

pulses

Pulsout

microseconds

resistance

LED

The '555' timer is an _____________ circuit that can be used for many different applications. In this Experiment, we used it to create a stream of _____________that caused an LED to blink. We then connected the _______________of the '555' to the BASIC Stamp and were able to measure the length of each pulse in _______________.

A potentiometer is a mechanical version of a___________ resistor. To increase or decrease the_____________ of the pot, you physically rotate the shaft, not unlike changing the_________________on your home stereo.

The "blink rate" of the '555' timer circuit is determined by the______ of a resistor (measured in ohms) and a capacitor (measured in ______________). A microfarad is equal to 1 / 1000000 of a farad.

The Pulsin command is the "input equivalent" to the______________ command. Pulsin can measure pulses up to _____________ in length. In our program, using the ____________ command, we could actually measure the length of each pulse that was blinking the___________.

Utilizing hardware to accomplish simple things is sometimes the best solution to accomplish a given task. If you have an application that needs to do two things ______________, you will need to weigh the benefits / disadvantages of adding additional _____________ circuitry to your design.

Why did I learn it?

This Experiment demonstrates the interfacing of other types of integrated circuits to a microcontroller. Microcontrollers are only capable of doing one thing at any one time. In many cases, this restriction isn't a problem because the microcontroller operates at such a high rate of speed. If however, you absolutely need to be doing more than one thing at a time, the challenge can be easily solved by using additional circuitry as we did with the " 555" timer integrated circuit.

The "555" timer has been used in innumerable applications and products throughout the years. In this experiment, we used the timer in what we call "astable multivibrator" mode. This was a relatively simple example of how to off-load some of the processing that ordinarily would have to be accomplished by the microcontroller. Many products that use a microcontroller as their central processing unit (CPU), rely heavily on additional circuitry to accomplish certain tasks.

This is not to say that a microcontroller cannot do the job, but rather it is sometimes quicker and more cost-effective to use additional circuitry, to accomplish a given task. As you design your own circuits, you'll need to make decisions between additional code or additional hardware to come up with the most appropriate solution.

In some upcoming experiments, we will be connecting many other types of IC's to the BASIC Stamp which significantly enhance its operation and capabilities to interact with the "Real World".

And of course, knowing how to interface different types of integrated circuits and components together is one of the foundational disciplines required of an electronics engineer.

How can I apply this?

As you continue to experiment with microcontrollers, you'll discover many different ways to interface or connect things. Some of these methods may be from some "application note" that was developed by a semiconductor company, still others might be your own creation. In any event, knowing the basic methods of connecting IC's together to form a reliable product is a very valuable skill.

Many of you have pagers or cell phones. These, as we've mentioned before have microcontrollers as their basic "brain". But in order for these devices to realize their true potential, they need "support circuitry" (not unlike our 555 timer). And the ability to design a suitable hardware solution will always be in demand.