RCTime Notes

dan trueman

There are a number of ways of setting up RCTime on a Basic Stamp, and getting familiar will all of them can help clarify a number of things about how the Stamp works and also just about basic electronics in general.

The idea behind RCTime is to measure how long it takes to charge up or discharge a capacitor. This will depend on the circuit the capacitor is in, and particular the value of a variable resistor that impedes flow to/from the capacitor. The following three examples (all drawn from the Basic Stamp manual) illustrate different ways to go about all this.

Example 1:

Here we begin by setting the I/O pin “low” (equivalent to ground, or Vss), which means that both sides of the capacitor will be seeing ground and the capacitor will discharge any charge it has accumulated. We then remove the I/O pin from the circuit (by putting it in “input” mode), at which point one side of the capacitor will see +5 volts (or Vdd) through the resistor R and begin to charge. We measure the time it takes at the I/O pin for the voltage to rise from 0V and cross a certain threshold (around 1.5V on the Stamp); this tells us how quickly the capacitor is charging, which depends on the value of the resistor R — the higher the resistance, the longer it takes for the capacitor to charge. (We put a resistor (220 Ohm here) between the I/O pin and the rest of the circuit to make sure we don’t create a short circuit when we set the I/O pin low (this could happen if the variable resistor R can go to 0 Ohms)).

Here is the code that would make this work:

```plaintext
result VAR Word
' discharge the cap by setting the I/O pin (pin 0 here) “low”
LOW 0
' wait for 1 ms to make sure the cap has discharged
PAUSE 1
' run RCTIME on pin 0
' RCTIME will put the pin (0, argument 1) in “input” mode and then measure how
```
long the pin remains in state 0 (< 1.5V, argument 2).
' it will put that measurement in the variable “result”
RCTIME 0, 0, result
' print out the result
DEBUG DEC ? result

The amount of time the capacitor takes to charge of course also depends on the size of the capacitor, so it is necessary to choose a capacitor that is not so large that the charging takes so long as to make the sensing uselessly slow. However, if the capacitor is too small, there won’t be much resolution since the capacitor will charge super fast.

Example 2:

Here we begin by setting the I/O pin “high” (equivalent to +5V, or Vdd) and wait briefly for the capacitor to charge (the “top” side of the capacitor in this diagram will be seeing +5V at this point, and will charge). After it has charged, we remove the I/O pin from the circuit and measure how long it takes for the capacitor to drain through the variable resistor R. Once again, we put a resistor between the I/O pin and the rest of the circuit to make sure we don’t create a short circuit (this time when we put the pin “high”) should R be able to go to 0 Ohms.

Here is the code that would make this work:

result VAR Word
' charge the cap by setting the I/O pin (pin 0 here) “high”
HIGH 0
' wait for 1 ms to make sure the cap has charged
PAUSE 1
' run RCTIME on pin 0
' RCTIME will put the pin in “input” mode and then measure how
' long the pin remains in state 1 (> 1.5V, arg 2). when the cap has gone < 1.5V
' it will stop timing and then put that measurement in the variable “result”
RCTIME 0, 1, result
' print out the result
DEBUG DEC ? result

Note that in this case, the cap has to drain from 5V to 1.5V, which is a much greater voltage change than in example 1, which only has to go from 0V to 1.5V. This will affect the overall timing, and some argue that example 2 is better because you will get higher resolution in a lower noise circuit. Probably doesn’t matter that much. However, this is
the circuit I’ve always used. I sometimes put a resistor in parallel with R, in the event that R is be super high (as with FSRs when they aren’t being pressed) and making the RCtime very slow; the parallel resistor will allow current to flow, keeping the RCtimes down, but still allowing a meaningful measurement if its resistance isn’t *too* low.

**Example 3**

![Circuit Diagram](image)

This example is similar to example 1, but can be a bit confusing. Here we begin by putting the I/O pin “high,” which will charge the “lower” plate of the capacitor in this diagram, so both sides will be at +5V (which means the capacitor is effectively “discharged” because there is no voltage difference across the plates!). We then remove the I/O pin from the circuit and measure how long it takes for the “lower plate” to discharge through the variable resistor, which can also be described as measuring how long it takes for the capacitor to charge (by creating a voltage difference across the plates).

Here is the code that would make this work:

```plaintext
result VAR Word
‘ “discharge” the cap by setting the I/O pin (pin 0 here) “high”
HIGH 0
‘ wait for 1 ms to make sure the cap has “discharged” (both “plates” at +5V)
PAUSE 1
‘ run RCTIME on pin 0
‘ RCTIME will put the pin in “input” mode and then measure how long the pin remains in state 1 (> 1.5V).
‘ it will put that measurement in the variable “result”
RCTIME 0, 1, result
‘ print out the result
DEBUG DEC ? result
```

See page 356 in the Basic Stamp Manual ([http://www.parallax.com/dl/docs/prod/stamps/BasicStampMan.pdf](http://www.parallax.com/dl/docs/prod/stamps/BasicStampMan.pdf)) for more about all this. Try them out on the HCI boards; it’ll be a good learning experience! Lemme know if you find any bugs in my code here.