The Nuts and Volts of BASIC Stamps
Parallax, Inc.
www.parallaxinc.com
Nuts and Volts
www.nutsvolts.com

Column #84 April 2002 by Jon Williams:

Sonic Sight – Seeing with Sound

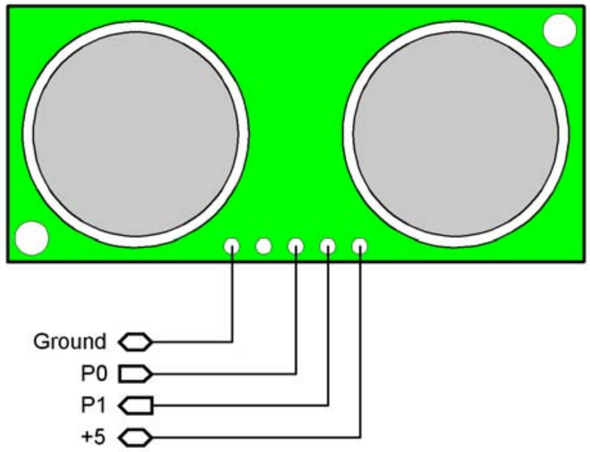
By now, many of you know that I live in Dallas and really the city. For a wanna-be geek, it's a great place to live in with so many technology companies located here. Where this helps a regular guy like me is when geeks [and I mean that in the best way] gather. My favorite local gathering of geeks is the Dallas Personal Robotics Group.

The DPRG is one of the bigger robotics clubs in the country. We have members from very young to very ... uh, "mature" (that's you, Ralph) – male and female. The diversity of the DPRG's membership is a real strength, and no matter what a person's experience level, the DPRG has something to offer.

One of my favorite aspects of club meetings is "show and tell." Members are allowed to show off their handiwork, whether it be hardware, software, a mix of both or just some bolt-out-of-the-blue idea. Everything is enthusiastically welcomed by the club.

Several months back, club member Bill Boyer – an experienced BASIC Stamp user – demonstrated the Devantech SRF04 ultrasonic range finder. Bill did some really neat things with it. His robot is able to scan an area and locate a target as narrow as a broom stick. I've used IR object detection for avoidance, but it's not really suitable for object location. I had to try the SRF04.

Figure 84.1: SRF04 to BASIC Stamp 2 Schematic



Well, it took a while, but I finally called Acroname (see sources) and ordered a sensor. It arrived a few days later, complete with documentation that included a demo program for the BS2. We'll expand on that program here.

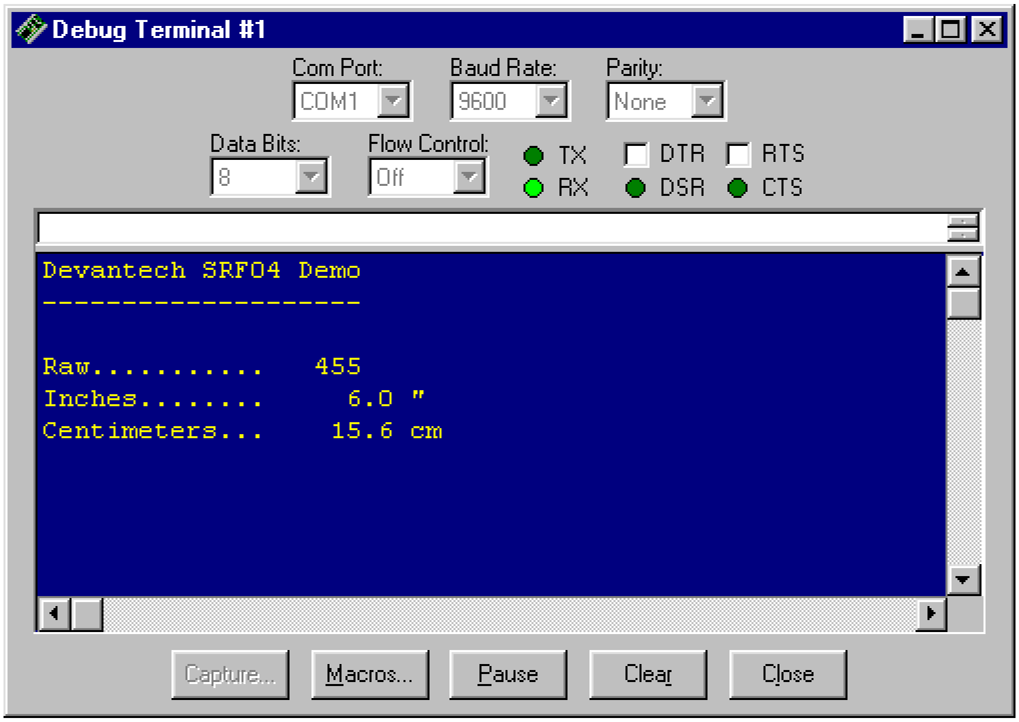
Blind Like A Bat

I find it interesting that we refer to bats as blind. I grew up in the desert of southern California and used to play a game with the local bats: dodge this rock. When the bats would come out in the early evening I would casually toss a small stone into their flight path. With amazing dexterity and precision, the bats would avoid the rock and continue on their quest for insects. Clearly, bats aren't blind; even if their eyes don't work particularly well. (Note: no bats were ever harmed in the course of my little game.)

The SRF04 "sees" the same way a bat does: by emitting a short burst of sound and "listening" for the echo. Under control of the Stamp (or other micro), the SRF04 emits an ultrasonic (40 kHz) sound pulse. This pulse travels through the air at about 0.9 feet per second (the speed of sound), hits something and then bounces back. By measuring the time between the transmission of the pulse and the echo return, the distance to the object can be determined.

The SRF04 outputs a high-going pulse that corresponds to time required for the echo to return. We can, of course, use **PULSIN** to measure it and determine the distance to the target. And ... there's a convenient side-effect with **PULSIN** on the BS2: the value returned for the round-trip is in two microsecond units – the same as a one-way trip (sensor to target) in one microsecond units.

Figure 84.2: SRF04 Debug Output

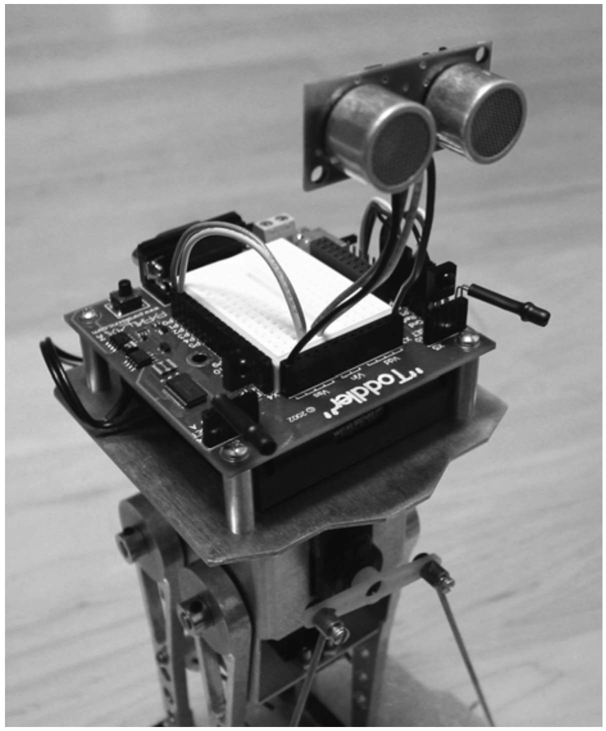


Build An Ultrasonic Tape Measure

There are a couple of small technical details to be aware of, but otherwise, coding for the SRF04 is a snap. The trigger pulse must be at least 10 microseconds wide. That's easy; we can do it with **PULSOUT**. The other requirement is that we must wait 10 milliseconds between measurements. No problem there, either.

The heart of this program is a subroutine called **Get_Sonar**. This routine started with the code sample that came with the sensor. What I found, however, was that the reading seemed to bounce around a bit. I felt like a little filtering (averaging several readings together) would smooth things out and make the output more useful.

Figure 84.3: Toddler Robot with Sonic Sight



The value returned by the routine will be stored in *rawDist*, so the code starts by clearing it. Then, within a loop, the code takes five readings from the sensor and averages them together. This may look a bit odd because most us think about adding numbers and then dividing to get an average. We do the dividing first and then add the result into the return value because we could have an overflow if we do all the addition first. Yes, the dividing first technique can lead to rounding errors, but only if the values were very small. I have not seen this when using the sensor. Since one inch (the minimum range of the sensor) is about 74 microseconds, dividing by five (loop value) each time through causes no problem.

With the raw measurement complete, we can display it and convert it to units that make more sense to us humans. To convert the raw measurement to inches, we need to divide by 73.746. If we change this to 7.3746 then we'll have tenths of inches. Remember that we can't divide by a fractional number, but we can multiply by using the star-star operator. Dividing by 7.3746 is the same as multiplying by 0.1356 (the reciprocal). To get the star-star parameter, we multiply 0.1356 by 65,536 to get 8886 (Note: Using 8886 with star-star is actually equally to multiplying by 0.13558 – pretty darned close). The same technique is used to convert the raw value to centimeters.

To keep the display neat I used a simplified version of Tracy Allen's right justification technique. Since the units are in tenths, the value is divided by ten before sending to **RJ_Print**. On return the decimal place is printed, then the tenths digit by using the DEC1 modifier. How's that for an easy program?

Now that I've got my SRF04 working, I'll start on the "search" routine. Maybe Bill will help me with some code.... If all goes well, my BOE-Bot will be able to locate a retrieve empty soda cans in some future DPRG Roborama. I'm a long way from that though. When I get there, I'll share it with you.

More Robot Goodies

Our friends at Quadravox have been busy and they've come up with a couple of cool new modules that are useful for robotics.

The first is the QV316M4-TTS. This module is identical to the QV306M4 sound module, except that is does speech synthesis (male voice) through in entry of allophones. For those of you have been scratching to locate an SP0256-AL2 – forget about it. Go get a QV316M4. It has much better quality than the SP0256-AL2, an onboard amplifier and an easy-to-use serial interface. It will buffer up to 40 allophones so that your micro isn't tied up while the unit is "talking." As I indicated a moment ago, if you're using a QV306M4 to do [pre-recorded] speech, you can pop a QV316M4 into its place and have an unlimited vocabulary.

The second new product is called the QV356M4 – the BOE-Bot Speech Board. This product was designed to mount right onto the Parallax BOE-Bot chassis. The QV356M4 combines the elements of the QV306 sound module and the QV430 programmer into one pcb; then it adds a whole host of new features.

In addition to its pre-recorded robot vocabulary, the QV356M4 can record a user message under control of the Stamp. The message is similarly played back. You can even change the sound of the prerecorded speech by changing the sampling and playback frequency. The feature that I particularly like is the ability to mix sounds generated by the Stamp (**FREQOUT**, **DTMFOUT**) through the QV356M4 amplifier. This means there's no need to record simple beeps that can be generated in code by the Stamp.

Both units are available from Parallax. Check the web site for additional details, documentation and demo source code.

Oops...

It came to my attention that another contributor to Nuts & Volts suggested (in the February issue) that the BASIC Stamp is not capable of decoding and analyzing Sony IR signals. I got several calls and e-mail notes from confused readers regarding this statement.

Since I work for Parallax and have more Stamp experience than the other author, please allow me to set the record straight: the BS2sx and BS2p are great at Sony IR decoding – all twelve bits of the consumer protocol. And in case you missed it, be sure to check out my article called "Control From the Couch" that was published in this column in August of 2001. The article includes IR signal analysis and complete 12-bit decoding using the BS2sx or BS2p.

Even the "elder statesman" (the stock BS2) is capable of basic IR decoding and can handle simple remote commands like channel up and down, volume control and power on and off. We've entertained thousands of school children and trade show visitors the last couple of years with BS2-powered, IR-controlled BOE-Bots and other Parallax demos. We love IR control! Come visit us at a trade show or educational course and you'll see just how much.

Thanks for allowing me to clarify. Until next time, Happy Stamping.

Resources

Acroname
www.acroname.com

Devantech
www.robot-electronics.co.uk

Dallas Personal Robotics Group
www.dprg.org

```
' =========================================================================
'
'   Program Listing 84.1
'   File...... SONIC SIGHT.BS2
'   Purpose... Devantech SRF04 Ultrasonic Range Finder
'   Author.... Jon Williams
'   E-mail.... jonwms@aol.com
'   Started...
'   Updated... 06 MAR 2002
'
'   {$STAMP BS2}
'
' =========================================================================


' -------------------------------------------------------------------------
' Program Description
' -------------------------------------------------------------------------
'
' This program uses the Devantech SRF04 to measure the distance between the
' unit and a target.  Display is raw value, inches and centimeters.
'
' Conversion formulas:
'
' inches = echo_time / 73.746           (use 7.3746 for tenths)
' centimeters = echo_time / 29.033      (use 2.9033 for tenths)


' -------------------------------------------------------------------------
' Revision History
' -------------------------------------------------------------------------


' -------------------------------------------------------------------------
' I/O Definitions
' -------------------------------------------------------------------------

Trigger         CON     0
Echo            CON     1


' -------------------------------------------------------------------------
' Constants
' -------------------------------------------------------------------------

MoveTo          CON     2                       ' cursor position control


' -------------------------------------------------------------------------
' Variables
```

```
' -------------------------------------------------------------------------------

pWidth          VAR     Word                    ' pulse width from sensor
rawDist         VAR     Word                    ' filtered measurment
distance        VAR     Word                    ' converted value
blips           VAR     Nib                     ' loop counter for measurement

temp            VAR     Word                    ' value for RJ_print
digits          VAR     Nib                     ' used by RJ_Print


' -------------------------------------------------------------------------------
' EEPROM Data
' -------------------------------------------------------------------------------


' -------------------------------------------------------------------------------
' Initialization
' -------------------------------------------------------------------------------

Init:
  PAUSE 250
  DEBUG CLS
  DEBUG "Devantech SRF04 Demo", CR
  DEBUG "--------------------", CR, CR
  DEBUG "Raw........... ", CR
  DEBUG "Inches........         ", 34, CR
  DEBUG "Centimeters...         cm", CR


' -------------------------------------------------------------------------------
' Program Code
' -------------------------------------------------------------------------------

Main:
  GOSUB Get_Sonar                               ' take sonar reading
  DEBUG MoveTo, 15, 3
  temp = rawDist
  GOSUB RJ_Print                                ' display raw value

  DEBUG MoveTo, 15, 4
  distance = rawDist ** 8886                    ' divide by 7.3746
  temp = distance / 10
  GOSUB RJ_Print                                ' display inches
  DEBUG ".", DEC1 distance

  DEBUG MoveTo, 15, 5
  distance = rawDist ** 22572                   ' divide by 2.9033
  temp = distance / 10
  GOSUB RJ_Print                                ' display centimeters
  DEBUG ".", DEC1 distance
```

```
  PAUSE 200                                     ' delay between readings
  GOTO Main

  END


' -------------------------------------------------------------------------------
' Subroutines
' -------------------------------------------------------------------------------

Get_Sonar:
  rawDist = 0
  FOR blips = 1 TO 5
    PULSOUT Trigger, 5                          ' 10 uS trigger pulse
    PULSIN Echo, 1, pWidth                      ' measure distance to target
    rawDist = rawDist + (pWidth / 5)            ' simple digital filter
    PAUSE 10                                    ' minimum period between pulses
  NEXT
  RETURN


RJ_Print:                                       ' right justify
  digits = 5
  LOOKDOWN temp, <[0,10,100,1000,65535], digits
  DEBUG REP " "\(5 - digits), DEC temp
  RETURN
```