Column #48, April 1999 by Lon Glazner:

One Step Beyond the Application Note

Application notes are generally a good starting off point for many designs. It's always nice if you can learn from example. There's nothing wrong with learning from your own mistakes, but it's usually beneficial if you can avoid as many as possible. More importantly, there's no reason to re-invent the wheel, especially in electronic engineering. I'd like to share with you a process that I use when starting a design, and then walk you through an example. When I'm embarking on a design that utilizes an IC or processor that's new to me, I try to ensure that I do three things before any actual design work begins.

First, I download or order the data sheets for the part in question. If the part is not going to meet your design requirements, you will probably find this out by reading the first few pages of the data sheet.

The general description on the first page is usually a glowing report detailing all of the wonderful things that the device can do. This description overtly avoids any discussion of a particular part's shortcomings. Rest assured the information is available, it's just not on the front page. What you can find out from the general description are the key features of a product. If these features seem to meet your design's needs, then you've cleared the first hurdle.

Next, check the maximum ratings of the part in question. Supply voltages, operating current, and operating speeds can all be found either in the maximum ratings section or in the electrical characteristics that usually follow in table form. When dealing with families of parts that are all fairly similar, the manufacturer usually has a selector guide with a cross-reference. This selector guide can be useful in weeding out undesirable parts. Second, once a cursory overview has shortened the parts list, I always search for application notes that reference the parts I am interested in. Often times, the list of application notes is pretty large. This is the case with the BASIC Stamp. Not only does Parallax provide application notes and kits for use with the BASIC Stamp, there are also third party sources for application notes.

Sitting down and actually reading all of the application notes for a particular product would be time consuming and inefficient. I definitely don't recommend that path to anyone. But it is usually a good idea to familiarize yourself with the application notes that are available. Being able to reference source code or connectivity information from an application note can cut hours out of your design time, and reduce the possibility of errors in the final product.

The third — and final — step in the process of selecting a chip is to look for an errata sheet for the part. An errata (error-data) sheet will usually state errors in the original data sheets and/or provide updates due to bugs in the part's silicon. These can KILL you. Primarily because they won't be listed in hard copies of the data sheet that you might already have. This allows functional problems to seep into your design or create hours of misguided debugging attempts. Always look for an errata sheet! Most manufacturers make these available as soon as the problem is defined. I also check to be sure that any part I am selecting has not been placed on an obsolete part list. There's nothing worse than completing a design that can't be produced.

For the Stamp enthusiast, some of this stuff doesn't matter. For example, if you're designing a pump controller for your fish tank, you're better off using what you have on hand. In other words, who cares if you're using an obsolete part, if it's free and already in your tool box, you might as well use it.

Defining The Design

I just started designing a new product. In our company, we usually go through a complete paper design, cost estimate, and functional overview prior to any parts hitting the breadboard. I'm still in that paper design phase but, of course, I'm looking ahead at the testability of my prototype. And that's where the BASIC Stamp comes in.

I need a device that will take voltage measurements of a single AA NiCad battery. Initially, the accuracy of the measurement doesn't need to be that stellar. But later it would be great if I could fine tune the analog-to-digital conversion for very precise readings. I would like the tester to run off of a 9Vdc battery, and be semi-portable. It should also be expandable. For starters, I need to measure at least two voltages; future expansion of up to six voltages is desirable. The voltage data should be transferable to a PC for graphical analysis. And finally, I would like to be able to adjust the period between voltage measurements (the sample period). If I were to list the functions I'm looking for they would appear as they do below.
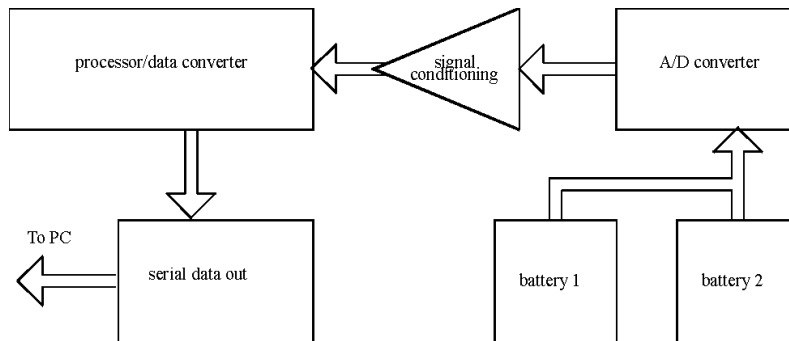
• Eight-bit analog-to-digital (A/D) conversion
• 2-6 A/D measurements
• 0-2Vdc range for each measurement
• Single supply derived from 9Vdc
• Serial data capability
• Adjustable sample period

Beyond the functional description, I need to be a little concerned about pricing. This is a test set-up for a product that may or may not go into production. So spending too much on the set-up would be a bad idea. In that same vain, spending too much time on the design would tend to eat up company resources. To me, this design has BASIC Stamp written all over it. It's usually good to have a block diagram of all functionality in a design. Figure 48.1 is the initial block diagram that I came up with.

There are a couple of things that immediately become apparent from the block diagram. The first is that the BASIC Stamp can suffice as both the processor/data converter and the serial data out block. I want the data to be sent to a PC in ASCII format. In this way, I can easily read it as a terminal program displays the data. This should also be something that the BASIC Stamp can accomplish.

I also realized from the block diagram that I will either need multiple A/D converters or some means of multiplexing the analog signals onto a single A/D. The latter concept is the least costly in parts so, for now, I'll be leaning towards that method. This device will likely require some conditioning of the analog voltage to be measured. The signal conditioning may be filtering, amplification, just plain buffering, or all of the above. To further minimize pricing, I'll be using the BASIC Stamp I (BS1). Most, if not all, of the functionality I desire can be derived from the BS1. So, at this point, I went to work scribbling furiously and perusing data sheets, and this is what I came up with.

**Figure 48.1: Voltage Tester Block Diagram**

Analog Multiplexing

Here's where the application notes come in. I know from being familiar with my Parallax BASIC Stamp manual that there is an application note for using the National Semiconductor ADC0831 eight-bit analog-to-digital converter (pages 77-79 application note #2). This is a single supply, single channel, eight-bit A/D. I just happen to have a few of these lying around, so that definitely fits my budget.

These parts utilize a serial data protocol that is compatible with NSC MICROWIRE, and is easily interfaced to the BS1. Since these parts can share serial data and clock lines, I could easily just hook-up six ADC0831 and, with a little "glue logic," I could select the A/D I want to read a voltage from with the BS1. But not only would this require multiple A/Ds, I would also have to duplicate my signal conditioning for each A/D. There is a better way. With a low-cost chip — the MM74HC4051 from Fairchild Semiconductor — I can multiplex up to eight analog signals onto a single A/D device.

Figure 48.2 is a functional diagram of how the MM74HC4051 fits into the circuit. With this chip, you can easily multiplex analog signals to a single A/D converter. There is an internal resistance that is associated with each switch. This creates the need for the signal conditioning circuit to possess at least a voltage follower as its first element. More discussion on this comes later. The logic family that I chose allowed signal levels from +6Vdc to -6Vdc to be passed by the analog switch. There were other forms of this chip (different logic families) that could operate off of +15Vdc, which I have to say intrigues me. But I'll leave that for a later date.

**Figure 48.2: MM74HC4051 Functional Diagram**
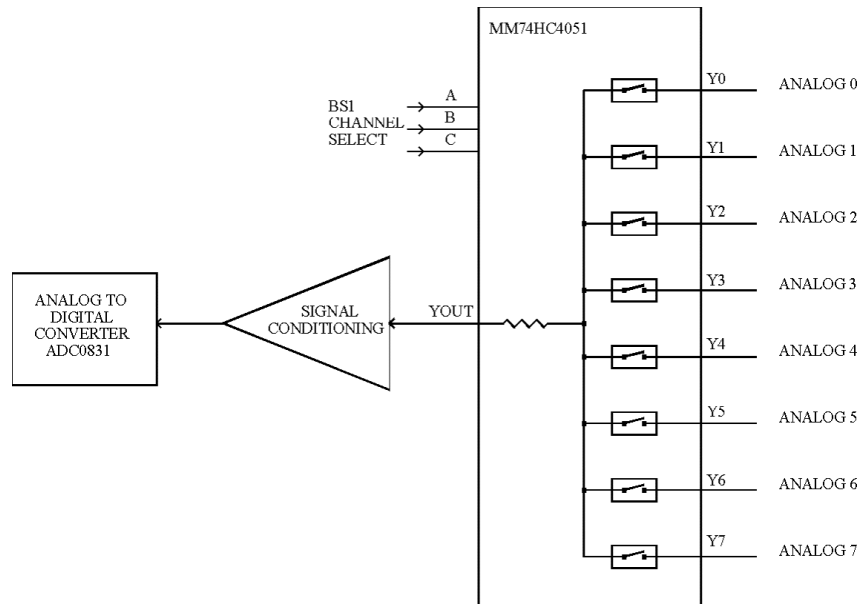
Signal Conditioning

As mentioned earlier, the analog multiplexer requires a voltage follower, or buffer, to ensure that the next section of my signal conditioning is not affected by the analog switch resistance. In reading through the data sheets for the MM74HC4051, I noticed that the resistance of the analog switch varies with the input voltage, as well as the voltage differential between the lower rail of the analog switch and its upper rail. This kind of characteristic is what you should look for in the data sheet. It tells me that unless I isolate this resistance from any filters or amplifiers, I will probably lose some accuracy in my measurements.

I chose the signal conditioning circuit in Figure 48.3 for a couple of reasons. The previously mentioned analog switch resistance explains the voltage follower. But the voltage follower is also in place to prevent the analog switch resistance from affecting the gain of my amplifier block. I realized early on that if the eight-bit A/D was measuring over a 5Vdc range, and my input signal was less than 2Vdc, about half of the A/D's measurement range would be unused. For that reason, I included a non-inverting amplifier stage. The gain for this amplifier can be defined as 1+R2/R1.

Figure 48.3: Signal Conditioning Circuit
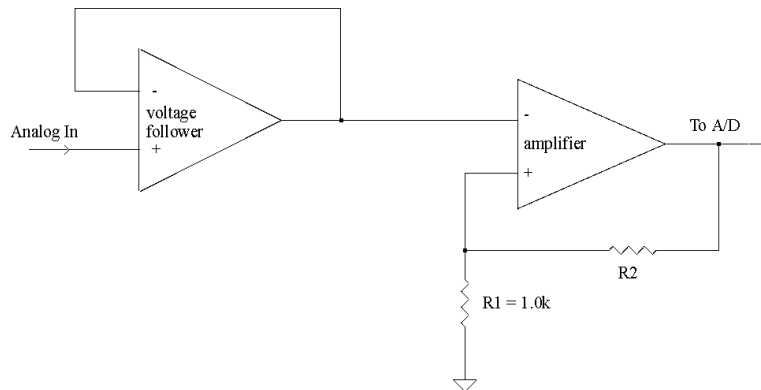
For instance, if I set R2 to 1.0K ohms, then my gain would be 1+(1000/1000) = 2. So if my input signal was 1.5V, the signal I would be measuring would be 3.0V.

It was at this point in the design that I re-called a BASIC Stamp application note written by Scott Edwards, of Scott Edwards Electronics fame. The application note that I'm speaking of described the Dallas Semiconductor DS1267 digital potentiometer, which just happens to use a serial interface similar to the ADC0831. Since both of these devices use the same communication interface, it's feasible to place them on the same bus (share the clock and data lines). Furthermore, it dawned on me that by replacing R2 in the signal conditioning circuitry with a digitally-controlled potentiometer, I could have an intelligent amplifier. This amplifier could scale its gain up or down for more accurate measurements. Now things started to get interesting.

Before going any further, I needed to select a low-cost operational amplifier (op-amp) for my signal conditioning circuit. Ideally, the op-amp should come in a dual or quad pack (two or four op-amps per chip). It should also be capable of single supply operation. Finally, an op-amp with rail-to-rail output (output ranges from roughly 0Vdc to +5Vdc) would be desirable to give me access to the full range of the A/D's eight-bit measurement.

There are a handful of other important characteristics useful in selecting an op-amp. These range from operating current to gain-bandwidth. Since I'm just amplifying DC voltages, most other characteristics are not entirely relevant to my design. I found a National Semiconductor part — the LM358 dual op-amp — that seemed to fit the bill for this design. In choosing this part, I opted to live without a rail-to-rail output.

This is just another example of the kind of trade-off you typically make during a design. I made sure that this part was pin-for-pin compatible with other rail-to-rail output op-amps in case I changed my mind at a later date.

The DS1267 digital potentiometer that I used was the 10K ohm version. There's a bit of a trick to using these parts for a design like this. When I first used this part, I measured the resistance across the part at both of its end points (with the pot set to 0 and 255 by the BS1). I found the resistance ranged from 500 to 9500 ohms. This led me to deduce that the wiper resistance was roughly 500 ohms. In reading the data sheet, I found that the wiper resistance for these parts can range from 400-1000 ohms. The actual resistance exhibited is related to the voltage at the wiper, which is related to the setting of the digital pot. But guess what, there's no place in the data sheet that actually described how this resistance was related to the voltage on the wiper. This required a little lab work.

I modified the DS1267 application note to step through resistance settings in steps of 10, and I measured the resulting resistance with a digital multimeter (DMM). This data was then graphed. The result is displayed in Graph 1. As you can see, the resistance remains linear over the range of settings for the digital potentiometer. This is good news, but since the DMM presents a high impedance load to the DS1267, it may not be representative of my circuit. This would still have to be verified empirically. Which means I could verify it during the software design.

Designing the Software

Before writing any software, I always create a schematic. The schematic is usually in pencil. There's no reason to generate it with a computer assisted design (CAD) program just yet. Chances are your schematic will change as you write the software. If you're going on to design a printed circuit board (PCB), then you'll probably end up modifying both your software and schematic to ease layout requirements and reduce noise.

It's also a good idea to generate a flow chart of some sort for your software prior to sitting down to write it. Quite often my BASIC Stamp programs are relatively short. For this reason, I often forego the flow chart in lieu of a list of subroutines. If your program is long and has quite a few branches, gosubs, or gotos, then flow charting before and after you write the software is something I would recommend.

Originally, I had envisioned being able to set the gain of the amplifier block in my software with an equation. But, in the end, I opted for maintaining the gain settings by storing them in EEPROM. The BS1 Write and Read commands make it simple to implement indirect addressing to access stored constants.

This also allowed me to modify the potentiometer settings to exactly match desired gain values.

During initial testing, I've found that the voltage measurements are typically within 2-3% of the actual voltage. It would be possible to utilize both of the potentiometers in the DS1267-010 in order to increase the resolution of the gain setting function. Doing this would remove some of the error exhibited by this voltage measurement technique. It would also be possible to have a reference voltage at one of the analog inputs.

By using a known reference voltage, a subroutine could easily be written that would adjust the gain values to be stored in EEPROM to the correct values. For instance, with a reference voltage fixed at 250mA, the routine could set the gain to 2 and adjust the potentiometer value up or down until the A/D reading was closest to 500mV. This new value could then be stored in EEPROM. This could continue until all gain values have been modified. Afterwards, the regular program would commence. This kind of self-adjustment would be useful in removing errors induced by temperature fluctuations or component tolerances.

This kind of system can be easily devised with the versatile BASIC Stamp. The source code used for this design is listed in Code Listing 48.1: NV_499.BAS.
The software was simplified considerably due to the inclusion of both of the application notes provided by Parallax and Scott Edwards. Other than a little research, and some short subroutines, this design was painless. There's still some work to be done as far as getting the voltage measurements into a PC. And guess what? There's an application note for that described in the Parallax BS1 SEROUT command (pages 63-65 of the BASIC Stamp manual). I haven't yet implemented the serial communication routines, although this should not pose a problem. It is most likely that I will use a quickBASIC program to store and convert the incoming data. The resulting data file will then be opened with Microsoft Excel and the data will be graphed.

Figure 48.4 provides an oscilloscope screen capture of both the gain setting and the A/D measurement. You can actually see the communication to the DS1267-010 and the resulting increase in the amplifier's gain. This is followed closely by the A/D measurement. The analog channel 0 input is measured by the oscilloscope as 293.8mV. The output of the op-amp amplifier is 10 times the voltage at analog channel 0, or 2.938V. These measurements are displayed at the bottom of the screen capture.
The oscilloscope used was the Hewlett Packard 54645D - MS0. This oscilloscope has some neat features that allow you to monitor digital lines like a logic analyzer would, as well as display the analog channels.
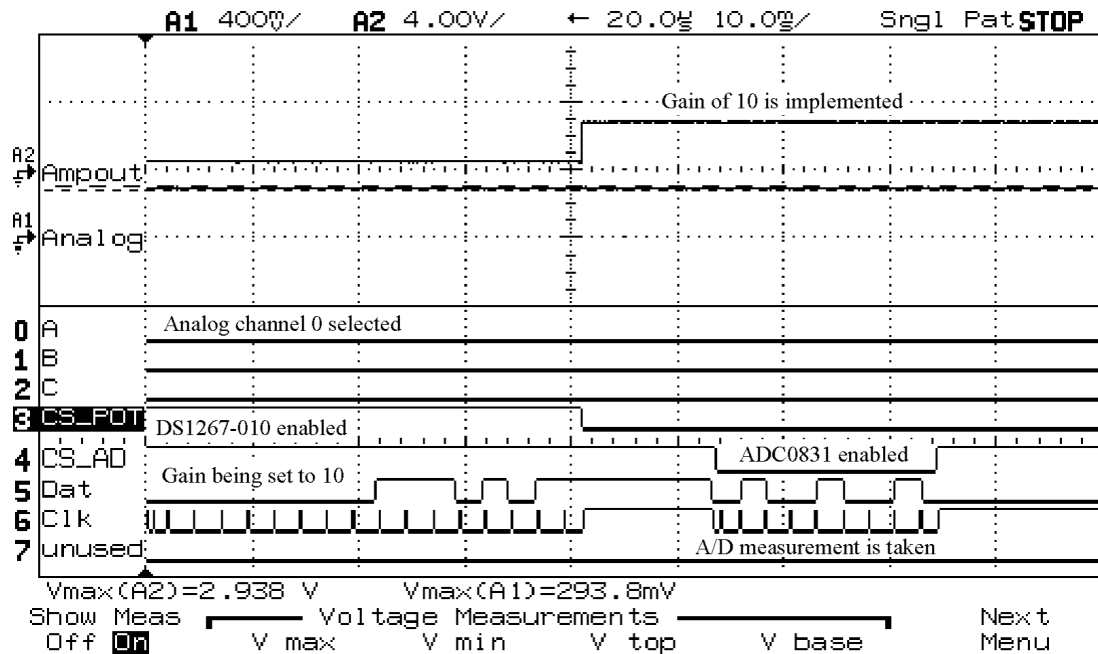
Figure 48.4: HP-54645D MSO Screen Capture

Figure 48.5 is a schematic of the finished project. Keep in mind that the serial output has not been completed, but I left a pin available for this function. The sample rate is also not adjustable at this point, but simply adding a PAUSE command, or increasing the ones already in the program would suffice as an adjustable sample rate.

In Closing

It was my hope to show a couple of things in this month's column. Primarily, I wanted to emphasize the use of application notes in engineering. Usually you can find an application note that can answer some of your questions, and point you in the right direction. I run into people now and again that expect to find an application note that will fit their needs, and solve all of their design woes. That's just plain unrealistic.
When working with electronics, you can expect to face certain hurdles. In many cases, somebody has already cleared the hurdle that you can't seem to get over. If it's a common problem, there may be an application note for solving it. If it's an uncommon problem, there's only one thing you can do. Get out your tools, engineering resources, pencil, and paper, and get to work. Whether your efforts culminate in success or failure, you're bound to learn something, and that's not a bad thing.

Figure 48.5: Schematic of finished project
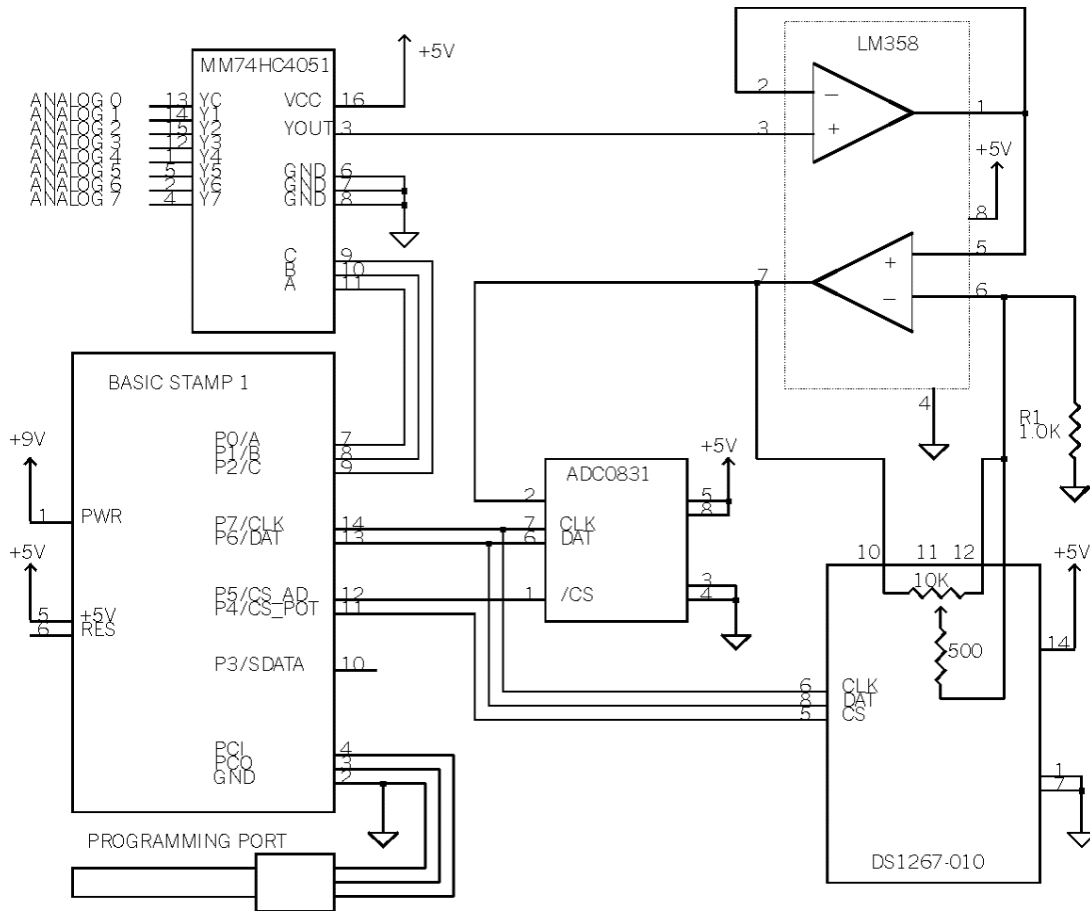
The design completed here was accomplished with two application notes, both of which were nearly identical. It took a little creativity, and a few hours of research but, in the end, it's something that I can use in the lab. There were no revolutionary concepts derived from integrals or squeezed out of linear algebraic functions. I'm not going to show up at the next COMDEX with 10,000 units of BS1-based battery voltage testers and make a mint with this design.

What I did do was build something that would help me get my design work done, and hopefully allow me to build a better product. And I think that's the true power of the BASIC Stamp. It allows you to put technology to work at a minimum cost. Not to

mention I tend to get a kick out of the idea that that little BASIC Stamp is making decisions and cranking out precise analog measurements while I type away at the PC. And I don't have to pay it a dime.

```
' Program Listing 48.1: NV_499.BAS Self Scaling 8 Channel A/D
'***********************************************************************
'NV_499.BAS: Automatically scaling variable gain amplifier. This software
'multiplexes 8 analog input values to one serial ADC0831. The multiplexing
'is done with a single low cost MM74HC4051 analog switch. It also allows
'for the control of a self scaling variable gain amplifier based on an
'LM358 as well as a DS1267-010. This amplifier exhibited less than 5%
'error when using it's maximum gain of 10. Gain settings for the amplifier
'are stored in the BS1's EEPROM memory, and were determined in the lab
'empirically.
'
'***********************************************************************
'BS1 Pin Definitions:
SYMBOL          CLK    = 7      'Serial clock line
SYMBOL          DAT    = 6      'Serial data line
SYMBOL          DATpin = pin6   'Symbol for data line pin
SYMBOL          CS_AD  = 5      'Chip select for ADC0831
SYMBOL          CS_POT = 4      'Chip select for DS1267-010
SYMBOL          SDATA  = 3      'Pin reserved for serial data output
SYMBOL          C      = 2      'A/D channel select MSB pin
SYMBOL          B      = 1      'A/D channel select pin
SYMBOL          A      = 0      'A/D channel select LSB pin
'
'BS1 Variable Definitions
SYMBOL          DSpots = W0
'Word variable for DS1267-010 dual pot settings
SYMBOL DSpots0              = B0
'Pot 0 settings variable
SYMBOL DSpots1              = B1
'Pot 1 settings variable(not used in this code)
SYMBOL          AD_sel = B3
'A/D channel select variable
SYMBOL          Clocks = B4
'Clock bit counter variable for serial communication
SYMBOL          AD_in  = B5
' A/D input variable used to store A/D measurements
SYMBOL          Gain   = W3
'Stores EEPROM address of pot settings for various gains
SYMBOL          AD_data     = W4
' Displays voltage in 100's of micro-volts
'
'
'***********************************************************************
START:
WRITE  8,235  'Pot 0 setting for a gain of 10 stored at EEPROM address 8
```

```
WRITE   7,214    'Pot 0 setting for a gain of 9 stored at EEPROM address 7
WRITE   6,186    'Pot 0 setting for a gain of 8 stored at EEPROM address 6
WRITE   5,158    'Pot 0 setting for a gain of 7 stored at EEPROM address 5
WRITE   4,126    'Pot 0 setting for a gain of 6 stored at EEPROM address 4
WRITE   3,98     'Pot 0 setting for a gain of 5 stored at EEPROM address 3
WRITE   2,66     'Pot 0 setting for a gain of 4 stored at EEPROM address 2
WRITE   1,38     'Pot 0 setting for a gain of 3 stored at EEPROM address 1
WRITE   0,10     'Pot 0 setting for a gain of 2 stored at EEPROM address 0

GOSUB   ANALOG           'Load first analog measurement
AD_SEL = 0              'Pre-set A/D channel to 000
PAUSE   1000            'Allow a little time for power up
'
'
'***********************************************************************
BEGIN:
GOSUB   SELECT_AD       'Select A/D channel with pins A,B,and C
GOSUB   SET_GAIN        'Set gain and take A/D measurement
GOSUB   TRANSLATE_DATA
'Translate data into 100's of micro-volts and display
GOTO    BEGIN
'
'
'***********************************************************************
'SELECT_AD: This subroutine cycles through the 8 different analog
'channels. The pin values are reset with each pass through to ensure that
'the correct A/D is selected. The channel being accessed in located in the
'variable AD_sel.
'
SELECT_AD:
  PINS = %11101000     'Clear lowest three bits, pins A,B, and C
  PINS = PINS + AD_sel'Add AD_sel(0-7) to PINS register
  DEBUG AD_sel         'Display current channel selected
  AD_Sel = AD_sel + 1
'Increment channel number for next time through
  IF AD_sel = 8 THEN Zero_AD_sel     'Make sure AD_sel never exceeds 7
  RETURN
Zero_AD_sel:
  AD_sel = 0           'Return AD_sel to 0
RETURN
'
'
'***********************************************************************
'TRANSLATE_DATA: This subroutine translates the A/D data into a value in
'100's of micro-volts. This is done to retain maximum resolution of the
'measurement. Simply divide the result returned in AD_data by 10,000 to
'determine the measurements value in volts.
'
TRANSLATE_DATA:
LET AD_Data = AD_in * 196
'Scale up A/D measurement by 10E3*8 bit A/D step(19.6mV/step)
```

```
LET AD_data = AD_data / Gain
'Divide measurement by Gain value (10 to 2)
DEBUG   AD_data,CR    'Display result to debug screen
RETURN
'**************************************************************************
'SET_GAIN: The SET_GAIN routine cycles through the gain values from 10 to
'2. Any measurement that is returned that is below the upper rail of the
'op-amp is considered valid. The LM358 upper rail is 3.5V. With a minimum
'gain of two the greatest voltage that can be measured is 1.75Vdc. By
'changing to a rail-to-rail output op-amp such as the LMC6132 this range
'can be increased to 2.5Vdc. A voltage divider could also scale down
'voltages, and allow larger voltages to be measured.
'
SET_GAIN:
Gain = 9                        'Start gain test at (9-1+2) or 10
Gain_test:
IF Gain = 0 THEN Done_with_gain 'Check for minimum gain setting
LET Gain = Gain - 1
'Subtract one from exiting Gain register
READ    Gain,DSpots0
'Use result as address pointer for EEPROM READ
GOSUB   OUTPOT                  'Set potentiometer to desired gain level
GOSUB   ANALOG                  'Read analog to digital converter
IF AD_in > 178 THEN Gain_test       '
Test for value greater than 3.5V(op-amp upper rail)
Done_with_gain:
LET Gain = Gain + 2          'Actual gain is Gain reg. + 2
DEBUG   Gain,CR               'Display Gain selected
PAUSE   10
RETURN
'**************************************************************************
'OUTPOT: This subroutine was taken directly from a Scott Edwards
'application note. It is also similar to the analog subroutine except that
'17 bits are transmitted.
'
OUTPOT:
LET DIRS = %11111111         'Make all pins outputs
HIGH CS_POT                  'Select DS1267-010
LOW CLK
LOW DAT
PULSOUT CLK,10
FOR CLOCKS = 0 TO 15
        LET DATpin = BIT15
        PULSOUT CLK,10
        LET DSpots = DSpots * 2
NEXT
LOW CS_POT                   'De-select DS1267-010
HIGH CLK
PAUSE   10
RETURN
'**************************************************************************
```

Column #48: One Step Beyond the Application Note

```
'ANALOG: This subroutine was taken directly from the Parallax application
'note #2 from their data sheet. Notes on this application note can be
'found on pages 77-79 of the Parallax BASIC Stamp data book.
'
ANALOG:
LET DIRS = %10111111            'Make DAT pin an input
LOW CLK
LOW CS_AD                       'Select ADC0831
PULSOUT CLK,10
LET AD_in = 0
FOR CLOCKS = 0 TO 7
        LET AD_in = AD_in * 2
        PULSOUT CLK,10
        LET AD_in = AD_in + DATpin
NEXT
HIGH    CS_AD                   'De-select ADC0831
HIGH    CLK
RETURN
'********************************************************************
END:                            'End of program space
```