Column #43, September 1998 by Jon Williams:

Complementary I/O

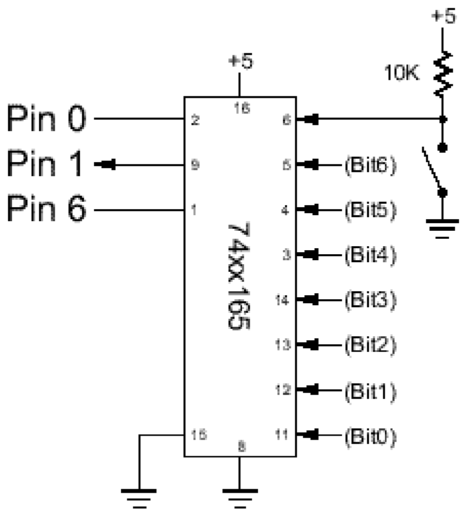
How many times have you been on the verge of finishing a very cool project only to "need" just one more input? Don't worry, it has happened to the best of us. This month, we're going to take it kind of easy and look at a couple of serial-driven chips that can expand your project's I/O capabilities without a lot of code.
Gimme Them Bits!

In the last couple of issues, Robert Nansel in the Amateur Robotics Notebook column has been building a neat little BS1-controlled robotic platform. Included in the circuitry are three bumper switches. Well, how would you like to look at more bumper switches? Say ... six, seven, or even eight? It's no problem with the use of a $1.00 part.

The 74HC165 is a parallel-in, serial-out shift register that complements the 75HC595 (serial-in, parallel-out) that we've talked about in the past. And using the HC165 with the Stamp is brutally easy. Take a look at Program Listing 43.1. The inputs of the HC165 are loaded when the Parallel Load pin (Stamp pin 6) is pulsed low. What this means is that you'll get a snapshot of the inputs when you strobe this line. Even if the inputs change while you're moving the data into the Stamp, your "snapshot" data will remain intact.

With the inputs secure, they are moved into the Stamp through synchronous shifting. The first thing we do is shift our data byte left. This is accomplished by multiplying by two. The reason we must shift first is that we want the last bit clocked-in to reside in Bit0.

Figure 43.1: 74xx165 connections to BASIC Stamp to add I/Os



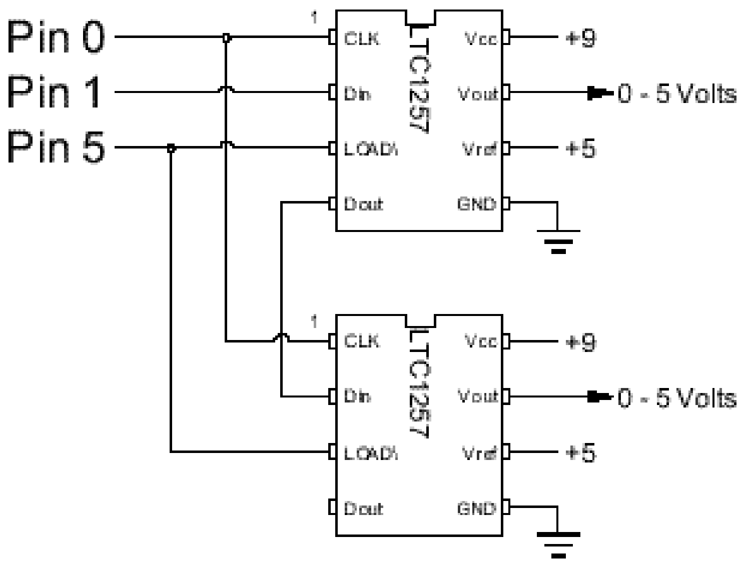
If we shift after retrieving this bit, our data will not be correct. Then we grab the bit from Pin 1 and send a clock pulse to get the next bit.

Just a reminder for new Stamp programmers: bytes B0 and B1 are the only Stamp 1 variables that allow bit-level access. My habit is to reserve these two bytes when I'm designing my programs — just in case I need to do something with individual bits. For you robot builders (and others), the bits of B0 can be renamed SYMBOLically. You could, for example, give the bits names like FrLft, FrCntr, FrRt, RrLft, RrCntr, and RrRt. This will make your control code easier to read and debug.

I've included code for the 75HC595 in the listing for your convenience. If you look closely you'll see that the code follows chip function: a mirror image of the 74HC165.
Precision Analog Output

Our next chip is an output device, a digital-to-analog converter (DAC) to be precise. While the LTC1257 is more sophisticated than the HC165, it's no more difficult to use. The LTC1257 from Linear Technology is a 12-bit DAC that complements the LTC1298 12-bit analog-to-digital converter (ADC) that Scott Edwards covered in this column back

Figure 43.2: LTC1257 connections to the BASIC Stamp



in June '95 (Stamp Application #4). Twelve bits over a five-volt range means that each bit is worth just over 1.2 millivolts. This is a lot of precision.

The only tricky part of the LTC1257 is Vref. Internally, the 1257 sets Vref to 2.048 volts. Since it's likely that we'd want to take out output all the way up to five volts, we can connect five volts to the Vref pin. Here's the trick: In order to get Vout all the way up to five volts, Vcc must be 2.7 volts greater than Vref (while remaining within the maximum parameters for Vcc). If you connect five volts to Vref and Vcc, you'll only be able to get 3.8 volts out of the DAC. This was an easy problem to solve. I simply connected my nine-volt, unregulated supply to the Vcc of the 1257. Presto, five volts out.

Listing 43.2 shows the code. Note the similarity to the HC595 code. The reason is that the LTC1257 contains a 12-bit shift register. Strobing the Load\ line causes the bits to be latched into the DAC section and the corresponding voltage appears on Vout.
In the event that you need more than one precision analog output, you can chain two or more 1257s together. Just connect Dout from one chip to Din of the next. Figure 43.2 shows the connections. Make sure that you output your "last" data first. What I mean is that the 1257 closest to the Stamp should be the last chip to which you send information. Once all your bits are shifted, pulse the strobe line to output the analog voltage.

The second section of Listing 2 shows how to deal with two daisy-chained LTC1257s. The LTC1257 is available from Digi-Key for a little over $8.00.

Help From The Real Experts

I've never claimed to be an expert at anything except getting into trouble with my soldering iron and a wild idea for a project. Thankfully, there are real experts among us, and two of my favorites have just released new books that are "must-haves" for any experimenter's bookshelf.

The first is Programming And Customizing The BASIC Stamp Computer (ISBN 0-07-913684-2) by well-known Stamp guru Scott Edwards. This book is — without a doubt — the most complete and comprehensive document ever published on using BASIC Stamps. In addition to several projects, Scott's book covers electronic fundamentals, construction and experimentation practices, related circuits and calculations, and much, much more. If you're new to Stamps and electronics project building, GET THIS BOOK! And if you've been using Stamps for a while, GET THIS BOOK! It's really that good. The book comes with a CD with all the source code and tools you'll need to build the book's projects. The CD also includes all of Scott's "Stamp Applications" articles (1-29) from Nuts & Volts Magazine.

The next is Serial Port Complete (ISBN 0-9650819-2-3) by Jan Axelson. Jan has written for many technical publications and has been one of my favorites for a long time. During my recent move, I actually found in my desk one of Jan's articles that I'd clipped in 1990!

Serial Port Complete is an accurate title. This book covers everything you'd ever want to know about serial ports and network projects. And it's not all theory. This is a very practical book, full of detailed examples — including Stamps and Visual Basic. No doubt some of you noticed that I've put off part three of the StampNet articles. I want to spend more time with Serial Port Complete before I wrap up that project.

Both books are professionally published and put together exceptionally well. Each contains detailed code listings, clear and concise diagrams, tables, and schematics. Let me reiterate: these books are "must-haves" for serious experimenters. And for you Web-shoppers, both books are available from Amazon.com.

Follow-Up On Past Projects

I recently moved back to the Dallas area and, boy, am I happy about that. Among the many cool things in the Dallas/Fort Worth metroplex is my favorite store, Tanner Electronics. You know that look of absolute glee that crosses a child's face when he enters a toy store? Well, that's how I look when I go into Tanner's.

Unless you live in the area, you've probably never heard of Jim Tanner or his store. So why do I bring it up? Two reasons: Jim regularly carries the SP0256-AL2 and 3.12 MHz crystals and will supply them to his friends at B.G. Micro if they run out. So those of you that were having trouble getting crystals for the voice project we did a few months ago, you can get them from B.G. — Jim promises me he will take care of the crystals for B.G. if they don't have them.

Jim's not fully prepped for mail order yet, but he will be soon. Stay tuned to Tanner's Web site at www.tannerelectronics.com for details.

Have fun and I'll see you next time.

```
' Program Listing 43.1
' Stamp Applications: Nuts & Volts, September 1998

' ----[ Title ]-----------------------------------------------------------
'
' File...... DIGIO.BAS
' Purpose... 74HC165/74HC595 Demo
' Author.... Jon Williams
' E-mail.... jonwms@aol.com
' WWW....... http://members.aol.com/jonwms
' Started... 02 AUG 98
' Updated... 02 AUG 98


' ----[ Program Description ]---------------------------------------------
'
' Demonstrates digital I/O expansion using readily available shift
' registers.


' ----[ Revision History ]------------------------------------------------
'
' 02 AUG 98 : Rev 1 complete


' ----[ Constants ]-------------------------------------------------------
'
SYMBOL Clk     = 0                    ' clock
SYMBOL DPin    = 1                    ' data pin
SYMBOL Data    = Pin1                 ' data bit
SYMBOL _165    = 6                    ' HC165 strobe
SYMBOL _595    = 7                    ' HC595 strobe


' ----[ Variables ]-------------------------------------------------------
'
SYMBOL temp    = B0                   ' using for shift in/out
SYMBOL shift   = B2                   ' shift counter


' ----[ EEPROM Data ]-----------------------------------------------------
'


' ----[ Initialization ]--------------------------------------------------
'
Init:   Pins = %11000000              ' _595 & _165 = H, Clk = L
        Dirs = %11000001
```

```
' ----[ Main Code ]----------------------------------------------------
'
Main:   GOSUB DigIn                    ' get HC165 inputs
        DEBUG #%temp,cr                ' display
        PAUSE 100
        GOTO Main
        END


' ----[ Subroutines ]--------------------------------------------------
'

DigIn:  PULSOUT _165, 5                ' load the inputs
        INPUT DPin                     ' prepare for data in
        FOR shift = 1 TO 8             ' shift 8 bits
          temp = temp * 2              ' rotate left (MSB first)
          Bit0 = Data                  ' get a bit
          PULSOUT Clk, 5               ' clock out the next bit
        NEXT
        RETURN


DigOut: OUTPUT DPin                    ' prepare for data out
        FOR shift = 1 TO 8             ' shift 8 bits
          Data = Bit7                  ' MSB first
          PULSOUT Clk, 5               ' clock the bit
          temp = temp * 2              ' get next bit
        NEXT
        PULSOUT _595, 5                ' bits -> outputs
        RETURN
```

```
' Program Listing 43.2
' Stamp Applications: Nuts & Volts, September 1998

' ----[ Title ]-----------------------------------------------------------
'
' File...... LTC1257.BAS
' Purpose... LTC1257 DAC Demo
' Author.... Jon Williams
' E-mail.... jonwms@aol.com
' WWW....... http://members.aol.com/jonwms
' Started... 02 AUG 98
' Updated... 02 AUG 98


' ----[ Program Description ]---------------------------------------------
'
' Demonstation program for the Linear Technologies LTC1257 12-bit DAC
' (digital to analog converter). Note that Vcc must be 2.7 volts greater
' than Vref in order to take the analog output all the way up to Vref.
' For conveniece, the 9v raw supply to the Stamp is used for Vcc while the
' regulated outpuIt from the Stamp is used for Vref.
'
' Vout = Vref / 4096 * dacValue
' Output resolution at 12 bits is 1.2 millivolts


' ----[ Revision History ]------------------------------------------------
'
' 02 AUG 98 : Rev 1 complete


' ----[ Constants ]-------------------------------------------------------
'
SYMBOL Clk     = 0                    ' clock
SYMBOL DPin    = 1                    ' data pin
SYMBOL Data    = Pin1                 ' data bit
SYMBOL _DAC    = 5                    ' LTC1257 Load\


' ----[ Variables ]-------------------------------------------------------
'
SYMBOL temp    = W0                   ' using for shift in/out
SYMBOL tempL   = B0
SYMBOL tempH   = B1
SYMBOL dac1    = W1                   ' output data
SYMBOL dac2    = W2                   ' output data
SYMBOL shift   = B6                   ' shift counter


' ----[ EEPROM Data ]-----------------------------------------------------
'
```

```
' ----[ Initialization ]--------------------------------------------------
'
Init:  Pins = %00100000
       Dirs = %00100011


' ----[ Main Code ]-------------------------------------------------------
'
Main:  dac1 = $07FF                   ' about half
       dac2 = $0FFF                   ' max output

       ' output to one DAC
       temp = dac1
       GOSUB AnOut
       PULSOUT _DAC, 5                ' output the voltage

       END                            ' remove for two units

       ' output to multiple DACs
       temp = dac2                    ' last DAC first
       GOSUB AnOut
       temp = dac1                    ' closest DAC last
       GOSUB AnOut
       PULSOUT _DAC, 5                ' output the voltage(s)

       END


' ----[ Subroutines ]-----------------------------------------------------
'

AnOut: OUTPUT DPin                    ' prepare for output
       FOR shift = 1 TO 12            ' shift 12 bits
         Data = Bit11                 ' output the bit (MSB first)
         PULSOUT Clk, 5               ' clock bit into LTC1257
         temp = temp * 2              ' get next bit
       NEXT
       RETURN
```