A Serial Interface for Scotty's Spectrum Analyzer

Introduction:

Scotty's spectrum analyzer is computer controlled via the LPT parallel printer port. Most current computers do not feature this way of sending data to a peripheral device – the USB port reigns supreme. This creates problems if you need to operate the device with a new computer.

A number of issues arise from a conversion to serial control:

1) First and foremost all data sent to and received from the spectrum analyzer must be in "bit" form. The most common standard is to send a start bit, followed by 8 data bits and one stop bit. So to send one byte (eight bits), ten bits are actually transmitted.

The LPT interface does not have this constraint – it can send 12 bits (8 data bits and 4 control bits) at once. Scotty designed his control board around this feature and uses various "AND" and "OR" gates to send the correct signals to the right places. For this to work properly ALL twelve control lines need to be active at the SAME time. This is clearly not possible with a serial interface.

There are a number of ways to accumulate the individual bits and the most widely used method is to use a microprocessor. It reads (or writes) 8 bits at a time and stores them in its memory. I have chosen a PIC 18F4550 for this purpose. It is widely available, inexpensive and has good support from its manufacturer. The uP has a serial interface and several analog to digital converters (ADC) hardwired in its silicon structure.

2) Secondly, Liberty Basic has fairly limited support for serial interfaces. It only supports simple commands for RS232 programming. It is important to note that both USB and RS232 use serial bit "streams" that is, data is sent and received one bit at a time. As pointed out above, RS232 sends 10 bits at a time, USB can send any number of bits one after the other but the basics are the same. This is not the place to go into details of packet structure and error correction and all the other issues associated with data communication. There are plenty of useful papers available on the internet and if you want to learn more about that, please consult theses sources.

The most common way to handle these problems is to create a command structure and then write the appropriate code for uP to execute them. An example:

Basic sends the letter 'A' to the uP via the serial interface. The uP is programmed to recognize the letter 'A' (decimal 65 == hexadecimal 41 == start bit, then 0100 0001, then stop bit). The uP next executes a series of instructions just like Liberty Basic. This may be: set pin 9 to +5V for 10uS and then return pin9 to 0V.

Likewise, the uP can be instructed to convert an analog voltage created by the log detector to a number that is then sent to Liberty Basic.

To summarize, this project creates a program for the PIC18F4550 that recognizes and executes a series of commands to send appropriate signals to the DDS(s) and the PLL(s) and the ADC(s). Scotty's Liberty Basic program is modified to handle this.

To implement the serial interface you need:

1) A programmed PIC 18F4550
2) A voltage level converter from RS232 levels (+15V, -15V) to uP levels (0V, 5V) i.e. MAX232
3) The modified Liberty Basic program
4) A USB to serial converter (Ebay, maybe $4)

OooOOOooo

Details for the PIC18F4550

The serial control board is very simple and can be bread boarded. There are only two IC's, the PIC and a MAX232 level converter. The board is powered by 5V and consumes about 100mA. A handful of other passive components and a 9pin RS232 socket completes the parts list. I am sure if the demand is there, Cash will produce a circuit board for this project.

The PIC connects to the spectrum analyzer via control lines as shown in the circuit schematic. There are differences between Scotty's control boards and the serial board described here but they are minimal. The only major difference is the reset pin on the AD9850 (pin 22) which is connected directly to pin 7 on the PIC. The diodes in Scotty's design are not needed.

SPIbusCLK (PICpin 15) and SPIbusDTA (PICpin 16) connect to all (3) PLLs and PLL1LE, PLL2LE, PLL3LE actually shift the data into the selected PLL. So if you send data on the SPI BUS and toggle PLL1 then only PLL1 will be programmed. PLL2 and PLL3 ignore the data. In practice the Liberty Basic program does this directly with the command 0x2 – the command instruction to program any PLL followed by the command instruction 0X1 – the command instruction to toggle PLL1LE (i.e. hex 2 followed by hex 1)

The DDS uses parallel control (PICpins 40 … 33) connected to the AD9850 pins D7 to D0. The same connection as Scotty's control board from the LPT port. WCLKDSS1 (PICpin 19) connects to W_CLK (AD9850 pin 7) and FQUDDDS1 (PICpin 20) connects to FQ_UD (AD9850 pin 8).

There are two LED indicators shown in the schematic (DS1, DS2). DS1 lights when a "overrun" error occurred (i.e PIC to fast for Liberty). DS2 lights for a "framing" error. The latter should never happen but if it does, reduce the baud rate in Liberty Basic to less than 115,200. (Line 1859)

Details of the instruction set for the PIC are specified in the Liberty Basic program. The PIC program is written in C18, the (free) compiler available from Microchip who manufactures the PIC series of micro controllers. Copies of the C programs for this project are available to anybody who can use them. Please Email me Apart from the compiler you need MPLAB IDE, also available as a free download from Microchip.

To program the PIC you need hardware and software as well as the included HEX file. This procedure is not hard – there are a number of software programs available (I prefer the WIN PIC programmer available as a free download from:

http://freenet-homepage.de/dl4yhf/winpicpr.html

You have to read the instructions for this program carefully – setting it up can be a pain in the neck. The hardware is simple enough and can be bread boarded for occasional use. Download the software and check the instructions how to make the hardware connections to program the PIC. IF you really, really can't do it, mail me a PIC18F4550 and I'll program it for you!!!!!

At this stage the PIC supports the original configuration of Scotty's spectrum analyzer and tracking generator. That is, one POS2120, two ROS1500, three LMX2326 and one AD9850. I plan to expand this once I have the hardware to test.

Details for the serial version of Scott's Liberty Basic program

A number of changes from Scotty's V110 will be noted. At the top of the program (line 12) is a con trol variable that permits 3 different debugging options. Set to 0. If you use debugging, don't forget to comment out the 'nomainwin' instruction.

Next (line13) gives a choice whether the PIC is connected or not. This allows the program to run without a PIC to check the execution. Set to 1 for normal use.

Line 20 (SERIALdelay=10) specifies a global delay in program execution speed. The program has NOT been tuned for speed at this stage and should run on just about any computer as is. Experiment with lower delays (<10) to speed the program.

Line 23
PICcom$="com3"' SPECIFY the COM port used by your system (check with Device Manager)
Must specify the Com port the program is connected. If you use the USB adapter you need to install the driver that comes with the USB adapter and the check with the device manager the port it uses. Mine uses com3 but com5 or com6 are possible.

All other parameters are same as in the original program from Scotty.

My email address is:
frankwinter@optusnet.com.au

have fun!
Frank VK4BLF QTHR