**Reign with the Sceptre**

This open-source & open-hardware project aims to be more than just a little board with a big microcontroller and a few useful peripherals—it seeks to be a 32-bit ARM7 fast prototyping system. To justify this title, in addition to a very useful little board, we also need user-friendly development tools and libraries that allow fast implementation of the board’s peripherals. Ambitious? Maybe, but nothing should deter you from becoming Master of Embedded Systems Universe with the help of the Elektor Sceptre.

*PCB, populated and tested, test software loaded*

Art.#090559-91 • $143.60

**InterSceptre**

Recently, Elektor introduced Sceptre, a fast prototyping system fitted with a 32-bit microcontroller. Even on its own, this board will let you produce some great results, but if you add an extension board to make it easier to access all its peripherals, the Sceptre platform becomes downright powerful. What’s more, if you fit this extension board into a suitable case, you’ll be able right from the start to develop a prototype that you can use 'properly' in a installation, with no trailing wires or bits of sticky tape holding everything together. Now that’s what you call fast, convenient prototyping!

*Kit of parts, contains PCB and components*

Art.#100174-71 • $187.10

---

**Modern technology for everyone**

**CD FPGA Course**

FPGAs have established a firm position in the modern electronics designer’s toolkit. Until recently, these ‘super components’ were practically reserved for specialists in high-tech companies. The nine lessons on this courseware CD-ROM are a step-by-step guide to the world of Field Programmable Gate Array technology. Subjects covered include not just digital logic and bus systems but also building an FPGA webservice, a 4-channel multimeter and a USB controller. The CD also contains PCB layout files in pdf format, a Quartus manual, project software and various supplementary instructions.


**Elektor more than just your favorite electronics magazine. It’s your one-stop shop for Elektor Books, CDs, DVDs, Kits & Modules and much more!**

www.elektor.com/shop

---

**The Elektor DSP radio**

Many radio amateurs in practice use two receivers, one portable and the other a fixed receiver with a PC control facility. The Elektor DSP radio can operate in either capacity, with a USB interface giving the option of PC Control. An additional feature of the USB interface is that it can be used as a source of power for the receiver, the audio output being connected to the PC’s powered speakers. To allow portable 6 V battery operation the circuit also provides for an audio amplifier with one or two loudspeakers.

*PCB, assembled and tested*

Art.#100126-91 • $240.40
Universal Prototyping Board

Having trouble prototyping with microprocessors? Tired of spending too much time cramming and wiring even the simplest projects? Here you learn to build your own universal prototyping board that you can customize for most of your projects.

For years I have used RadioShack’s 6” Experimenter PC Board (P/N 276-170) for prototyping projects, both for business and personal projects. I’ve found them ideal for electronic designs involving integrated circuits, especially digital designs. Using wooden supports, I can easily cascade these boards together using ribbon cables to prototype large complex circuits that are modular and thus easier to work with. I use these boards so much that I often buy out the store. The major drawback comes when using 40-pin DIP devices, such as microprocessors. It quickly becomes difficult to connect the microprocessor’s various ports to adjacent board modules; I rapidly run out of room for connectors. Many of my projects involved devices like Microchip’s PIC16F877 (an example is shown in Photo 1), which was the impetus for doing this project. Each time I used such microprocessors, I spent considerable time and tedious effort trying to cram everything onto the board and then wiring it together. Parts such as the dual inline header or the resonator don’t readily fit the hole pattern (the upper center of the board, see Photo 1), necessitating cutting of lands or jerry rigging parts in odd positions. And there is always the fear of running out of board before all the ports are connected. In Photo 1, note the limited space where connectors for the remaining four ports must still go.

Then came a bit of inspiration. Why not make a universal circuit board with the same outline as the prototype board and having the options needed for most of my applications? In this article, I’ll describe how I planned and then built a universal prototyping board.

GETTING STARTED

Once I knew I wanted to build a universal board, I took pen and pad in hand and decided what I wanted for options. Foremost would be headers for each port, as well as RS-232 and PC/SPI, the pulse-width modulation (PWM), and a header to reprogram the device using Microchip Technology’s PICkit 2 development programmer/debugger.

The most important question was the pinout configuration for connecting the ports to other boards. This was solved by using the same pinouts used by MikroElektronika (www.mikroe.com) on its EasyPIC4 development board. That way, I could use my EasyPIC4 to test and develop circuits and software and then add them to the prototype by merely plugging in the module. To divert a bit, for those of you who haven’t heard, MikroElektronika’s development system is probably the very best buy ever in microprocessor development systems. For just $250, you get a super universal development board that allows you to use most of Microchip’s F-series microprocessors. It has a system that allows you to plug in various peripherals such as LCD displays, buttons, RS-232, LEDs, and even USB and PS/2 keyboards. Plus, their software has the libraries supporting these various goodies.

For software you have a choice of three languages: Pascal, BASIC, or C. Being a long-time BASIC man, my choice was obvious. I’ve used a few other PIC BASICS (including one that cost as much as Mikro’s software and development board combined) and this one is, by far, the best. It has true procedures including local variables that allows you to easily build your own procedure libraries to use in different programs without worrying about duplicate variable names. Unlike my other BASIC compiler, Mikro’s easily handles interrupts in real time. Also included is full floating-point math capability and a nice integrated development environment (IDE) with a programmer on the development board. You just write your
program, compile it, and then program the device you are using on the development board.

To interface to external circuits, Mikro's development board uses 10-pin dual-inline headers with eight data bits, plus the ground and 5 V. Therefore, circuits built on a separate proto board can be easily plugged into the microprocessor and also have the required power and ground connection. Each port (A through E) has the same pinout. Usually, I build a circuit module and then test it out using short programs to test or exercise it. These exercisers usually develop into the calling procedures used in the main program. And RadioShack's PC board is ideal for prototyping most circuit modules with this system.

ADDITIONAL FEATURES

I chose other features as well. All connections to the board should be pluggable so the board can be quickly and easily removed for rework. There should be an on-board voltage regulator with a solder jumper allowing either an unregulated DC voltage or 5 V to be connected via a screw terminal plug, plus a power switch and a "power on" indicator LED. It needs a single inline pin (SIP) header so Microchip's PicKit-2 in-circuit programmer can be connected, allowing the reprogramming of the microprocessor without removing it. You should be able to use either the 40- or 28-pin narrow (0.3" pin spacing) microprocessors with or without sockets. Port headers are connected to 5 V and ground, so power can be easily distributed to the remaining prototype circuit boards.

The FC and SPI ports each connect to SIP headers and each also has 5 V and ground. The PWM output is connected to a two-pin SIP header, which includes ground. The serial ports (one hardware and one software driven) connect to Maxim Integrated Products's MAX232 level translator chip with solder jumpers, allowing either the standard RS-232 or TTL levels. The main serial port (hardware) goes to one four-pin SIP header and the auxiliary serial port goes to another. Both of these headers include 5 V and ground. A ground bar allows easy connection of test instrument grounds, while a "test point" output of the 5 V allows the easy connection of logic probes. It has a couple of test point outputs with LEDs that can be used in debugging and for triggering. Finally, an array of pads on any remaining blank board space allows some on-board prototyping of circuits.

The use of 10-pin dual headers with ribbon cables presented a drawback. Installing dual inline
headers on RadioShack's PC board requires cutting five traces in half—a rather tedious job with a razor knife. The alternative is to use a DIP ribbon cable header (Digi-Key part number HDP10S), which means soldering the cable to the proto board and having it flop around so it's always getting in the way when you're trying to do rework. Instead, I also designed a small circuit board to adapt a DIP header to the 0.3" spacing of the proto board. More about that later.

With the basic requirements now defined, I drew up the schematic [see Figure 1]. Note that microprocessor pin numbers are for the 40-pin devices. Refer to the pinout diagram posted on the Circuit Cellar FTP site for the conversion of pin numbers for the 28-pin devices. Although designed for PIC16F8x7 devices (I have a bunch of PIC16F877 devices from past projects), I also designed it for PIC16F7x7 devices, which have an internal oscillator. The two pins used for a crystal or ceramic resonator (the board allows you to use either) are also used as I/O pins for port E.

I laid out the board with the same outline as RadioShack's board [see Photo 2] so it can be easily mounted on the same carrier as the prototype boards. More about that later, too. The microprocessor is placed in the approximate center with connectors around the board's perimeter so cables will not be in the way when probing with instruments. To allow mounting on the wooden carriers, each end of the board must have the first half inch free of components on the underside—that is, no through-hole components are allowed. Two additional features are pad layouts for using either a TO-220 or a TO-92 voltage regulator and pad layouts for using either a resonator or a crystal (including loading capacitors). Also, enough vertical space must be left for clearance of the PICkit 2 programmer. I placed the ground bar off to the right end by itself so ground clips can be easily connected with a minimum obstruction of the board.

With the major components placed, I laid out the circuit, keeping the area inside the 40-pin DIP as clear as possible so the narrow 28-pin DIP could be placed in the center. I then completed the layout by connecting the appropriate pinout of the 28-pin DIP to the 40-pin pad. I also tried to keep the area near the ground bar as clear of circuit lands as possible to make a prototype area. As luck would have it, I had a second smaller area below the PICkit programmer that can be used if the components are not very high. I filled the main area with pads spaced at 0.1" in both directions, then connected clusters with lands on the top side so each cluster is visible. For the small area, I set some 0.25-W resistor and capacitor pads with joining pads to allow easy wiring. Each prototype area has a set of pads connected to ground to further facilitate prototyping.

POWER SUPPLY

Power is connected to the board using a pluggable terminal block at P1 as shown in Photo 3. Wires are connected to the board using these screw terminals (Digi-Key part numbers ED1613 and ED1682), the positive lead connecting to the pin near the center, while the ground connects to the other pin. If the power is regulated 5 V, then no regulator is required, so solder across the jumper pads marked "t" to the left of the voltage regulator. If a regulator is used, then a 78L05 [TO-92 outline] can be installed at A1 if current is less than 0.1 mA or a 7805 [TO-220] for current up to 1 A. The regulator can be installed with sufficient lead length to be bent over on its back and out of the way. In this position, a small heatsink can be installed. You can install a large filter capacitor before the regulator at C7 and a high-frequency suppression capacitor after the regulator at C8.

If you want a power switch (top center of Photo 3), install a PC-mount slide switch (Digi-Key part number EG1903) in S1; otherwise, solder a jumper wire between the center and top pads. For a power-on indicator, install an LED in LED1 (marked with just a 1) and a 330-Ω resistor in R1. This will give you a visual indicator that power to the prototype is on.

PROGRAMMER & PROCESSOR

A single inline header (center right of Photo 3) is installed at PICkit-2 using six pins from a breakout header strip (Digi-Key part number S1022-36). To program the microprocessor, just plug the PICkit-2 programmer into the header with its white arrow adjacent to the white arrow on the circuit board.
Note there is no key. The programmer can be plugged in either way, so make sure you get it right.

As previously stated, either a 40- or a 28-pin device can be used with this circuit board. While designed specifically for the PIC16F877, it can also be used with the devices listed in Table 1.

A 10-kΩ pull-up resistor is installed in R2 for MCLR and the programmer. A 0.1-μF capacitor is installed at C1 for noise suppression of the microprocessor. I have found through bitter experience that, if at all possible when doing prototype work, always use sockets for ICs, because there is nothing worse than desoldering a chip only to find out that it wasn’t the problem! For devices using an external clock, install a crystal or ceramic resonator in X1 and, if required, install loading capacitors in C9 and C10.

**I/O PORTS**

I use ribbon cable and 10-pin headers to connect the microprocessor’s ports to outlying prototype boards (except for Port E). Each port has its full eight I/O bits, ground, and 5 V. Since the dual-inline headers will not directly install on RadioShack’s proto board, I designed a small adapter board, which accepts a single 10-pin DIP header that conducts the signals to two single-inline header pins (same as used on PICKit 2 header) that are 0.3” apart, thereby allowing the pins to be inserted into RadioShack’s board. Now, port ribbon cables are easily plugged into the adapter board, which is soldered on the outlying proto boards.

Where port pins are needed on more than one proto board, I designed a “daisy chain” adapter board with two headers. They are electrically in parallel, so a second ribbon cable can be plugged in to connect a second proto board. The port pinouts on RadioShack’s proto board are the same as on the microprocessor board—that is, the pads on the proto board match the pinout pattern of the microprocessor board. A short, bare jumper wire is then soldered from the ground pads under the adapter to the bus strip on the upper edge of the proto board. A second jumper wire connects the 5-V pad to the lower bus strip. The individual port pins are connected to the circuit using insulated solid wire soldered to the appropriate pads. I prefer AWG #26 wire and use different colors to better navigate through the circuit.

I recommend using keyed headers and plugs for interconnecting port cables (see Photo 4) so there is only one way to plug in the cables. This feature can be a life saver. Imagine you’re working on it again months later and rushing to put it back together, and when you try to insert a plug in backwards, it doesn’t go in because of that key. And if you had put it in backwards, it would have blown a chip, which would have taken you two weeks to replace, because it’s almost always a chip that you don’t have a second one of.

The RadioShack proto boards are spaced at about 0.25” to allow cables to pass between them and run under the proto boards. This makes for a less-cluttered prototype and easier-to-connect test probes.

**TEST POINTS & LEDs**

I often add test points and LEDs to a microprocessor circuit to aid in debugging, especially at the systems level. Test points give you trigger signals for oscilloscopes, DSOs, and logic

<table>
<thead>
<tr>
<th>Function</th>
<th>Solder jumpers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main serial port</td>
<td>a, b</td>
</tr>
<tr>
<td>Aux serial port</td>
<td>a, b, c</td>
</tr>
<tr>
<td>No A1 regulator</td>
<td>d</td>
</tr>
</tbody>
</table>

Table 2—Solder shorts
analyzers that can be placed in the software exactly where you want them instead of trying to find just the right on-board signal. Test point 1 is bit 1 of port C, which is also the receive line for the auxiliary serial port, so you can easily check if the signal is present and of the right sense. Test point 2 is bit 0 of port D and test point 3 is bit 0 of port B.

Often, a simple visual indication is all that is needed to show that a process or function has started or finished, so I included two LEDs, driver by bit 0 and 1 of port E. Since 100Ω limiting resistors (R3 and R4) are in series with the LEDs, each LED can be quickly and simply removed from the circuit by cutting either lead of the resistor. Of course, the LEDs and resistors are only installed if needed.

Finally, since port E has only two or four I/O pins (depending on the microprocessor version), a six-pin SIP header is used. Like the other ports, both the 5-V and ground lines are included.

**SERIAL PORTS**

The serial ports (main and auxiliary) may be used with or without a level driver by shorting or not shorting solder pads "a" through "e," as shown in Table 2. In addition to the hardware serial port built into the chip, I made provision for a software serial port to be used as the auxiliary. I find having an auxiliary serial port very useful in debugging, particularly at the systems level. However, if the driver chip [to give true RS-232 levels] U2 is used, then both serial ports are connected. You can’t have one port with RS-232 levels and the other TTL.

I have some equipment that uses a noninverting TTL serial connection, therefore, I designed the circuit so the serial ports could be either TTL or RS-232 levels. If TTL is desired, then driver chip U2 and capacitors C2–C6 are left off the board. For the main serial port, solder across pads "d" and "e," and for the auxiliary serial port, solder across pads "a," "b," and "c." When using driver chip U2 with the auxiliary serial port, solder across pad "a" to connect the chip’s output to the microprocessor. This pad is included so you can have the driver chip for the main serial port, but leave port RC1 open for use if the auxiliary port isn’t being used.

Driver chip U2 is Maxim’s MAX232, which provides inversion and level driving for two serial ports. The chip uses capacitors C3–C6 to create the positive and negative output voltage swings. This gives an RS-232–level output between ±5 and ±8 V. Connect the serial ports to the outside world (DE9 or DB25 connectors) using single-inline header P8 for the main serial port or P9 for the auxiliary serial port.

**SPI/PC BUSES**

Since the SPI and PC synchronous serial buses use the same hardware resources in the microprocessors, you can use one or the other, but not both. Because of this dual function, both use the same I/O lines, and hence the headers are connected in parallel. And since you can only use one or the other, you need only install one header (or none if you aren’t using either). Like the ports, the SPI/PC headers also have ground and 5-V lines, so you can put synchronous serial devices on a proto board and connect them to the microprocessor board without any other ports having to be connected, too. But note that the I/O lines (port C) remain connected to both the serial headers and the port C header, so if you connect port C to another proto board, you cannot connect those pins to other circuits without risking corruption of the serial functions.

Also note that none of the recommended resistor terminations are included on the microprocessor board, so you must put them on your proto board. Keep in mind that when using multiple boards with synchronous serial devices, the last board in the wiring chain should be the board with the termination resistors.

**PULSE-WIDTH MODULATION**

Another often-used special-function device of microprocessors is the PWM used to drive things such as DC motors. Since such devices often need high current or require voltages greater than 5 V, the positive power supply is not carried. This is considered as strictly a signal output line. Like the synchronous serial buses, it is connected to port C, so if used, then bit 2 cannot be used on the port C header.

**MAKING THE BOARDS**

I wanted to have about six boards made so I would have a ready supply waiting when I needed one, but when I checked with my usual circuit board maker, I found that six would cost me almost $400. That was somewhat more than I wanted to spend, although I would have them in just a few days. I decided to check around with other board fabricators and see if I could find a better price. I quickly found that PCB-pool.com would make 10 boards for just $160, much better at $16 each instead of $60 plus. The tradeoff was 10 days for production instead of two, but
since I didn’t have a project waiting for them, I could wait.

For those of you who haven’t ordered your own circuit boards, you need something called Gerber plot files, which is the electronic artwork used by PCB fabricators to make your circuit boards. People are surprised to learn that Gerber plot files are just text files, which any editor or word processor can read and display just like any ASCII text file. This text is actually a set of data and codes that tells the plotter where to draw lines (X and Y positions), pad shapes, and their sizes. The drill file is another text file telling where to drill holes and what the hole sizes are.

Once you have designed your circuit board, you use your circuit board CAD program to generate the Gerber plot files for the top, the bottom, the silkscreen [white writing on circuit board], and the solder mask. This gives you a real professional circuit board just like you find in commercial equipment. With a set of plot files, it’s just a matter of choosing a vendor, making an order with your credit card, and sending the files as an email attachment. In as little as two days [premium price and overnight delivery], you have your circuit boards ready to stuff and solder.

To save money, I combined the artwork of the adapter boards with the main board and, having a shear, cut them apart to make separate circuit boards. But not everyone has a shear in their shop, so I’m including a file package with separate Gerber plots for each circuit board. It’s more expensive this way, but then a shear will cost you hundreds of dollars. These file sets are available on Circuit Cellar’s website. [The file names are COMBINED.zip for the one circuit board with all boards combined and SEPARATE.zip for individual board files.] Individuals may make and use as many of these circuit boards as desired, but may not sell or resell, either as an individual or business, any or all of the circuit boards.

MOUNT & CONNECT

The one thing I really like about RadioShack’s PC board is the ease of making a wooden carrier, which allows mounting several proto boards in line to prototype large, complex circuits. Photo 5 shows the mounting of RadioShack’s proto boards with my Universal Pic prototype board (this is an alarm project I’m doing for my home shop). For a back plate, I cut a common 1" × 8’ pine board to the length I thought I’d need. Then, using 0.5” square haswood strips [from Hobby-Lobby], I created two rails down the length of the pine board by gluing these wood strips onto the pine board, spaced apart by the width of the RadioShack proto board. That way, the proto boards can easily be mounted using #4 × 0.375” panhead screws. Just lay the proto board across the wooden rails so the board’s mounting holes on each end are on the wooden rails. Then put a screw through each hole into the soft wood. This is why we need the half inch on each end of the board free of any circuit devices; the solder bump underneath would keep the board from laying flat.

The nice thing is, when you need to do work on a proto board, you just unscrew the four screws and lift it up. That’s why we want to use pluggable cables to interconnect multiple boards. The carrier, in addition to being cheap to make, lies flat on the workbench or you can put some cross pieces on each end so it can sit upright. It easily goes out in the field to connect to real equipment and lets you fully test your circuit before committing it to a printed circuit board.

Finally, Photo 6 shows the development setup using a cable to connect Mikro’s EasyPIC4 development board to my project prototype. This is an easy way to program the universal board’s microprocessor with Mikro’s IDE. Cheap and easy to build and use—that’s why I’ve used this method of prototyping for years. And now, with my universal microprocessor board, it’s even easier to use.

James Lyman (jclyman@pobox.com) owns Labs Plus (www.labsplusdesign.com), a Texas-based company that designs and builds custom electronic instrument systems for security applications and nondestructive testing. Previously, he worked for Cyltek, Y-COR, The Southwest Research Institute (San Antonio, TX), Datapoint Corp., and Bendix Corp. James holds a BS in Aerospace Engineering, a BS in Electrical Engineering, and an MS in System Management.

PROJECT FILES


SOURCES

PIC16F877 Microcontroller and PICkit 2 programmer/debugger
Microchip Technology, Inc. | www.microchip.com

EasyPIC4 Development board
MikroElektronika | www.mikroe.com
A Tour of the Lab (Part 2)

The Frequency Domain

In the first part of this series, you learned about the time domain equipment a professional engineer needs to succeed in the ever-evolving embedded design industry. This article is about essential frequency domain equipment.

In the first part of this article series, I covered the topic of classical test equipment, from oscilloscopes to logic analyzers, frequency counters, and reflectometers. All such equipment has one thing in common: it works in the time domain. OK, I admit this is a complicated and probably pedantic way to say that such devices simply allow you to measure how a given signal evolves over time—which is usually what an engineer is looking for—but this enables me to introduce this second article. There is another kind of engineer who feels more comfortable in the frequency domain: the RF designer.

As you may know, frequency and time domains are basically two views of the same physical reality, thanks to the well-known Fourier transform. For example, a quick pulse in the time domain corresponds to a wide-band noise signal in the frequency domain. A sine wave in the time domain is a frequency “pulse” in the frequency domain. And any periodic signal in the time domain is a set of equally spaced harmonics in the frequency domain with given amplitudes and phases. So why aren’t oscilloscopes enough? The first answer could be that multi-gigahertz oscilloscopes are still quite expensive, but, as I’ll explain in this article, looking directly at the frequency domain could be a terrific tool for wireless designs and more. For example, a spectrum analyzer helps you to discover small perturbations buried in the noise and to obtain clues as to their root cause.

Now it’s time to wrap up my lab tour with a presentation of the two most common frequency-domain instruments: the spectrum analyzer and the network analyzer. I hope to show you both their main characteristics and what they are used for.

SPECTRUM ANALYZERS?

A spectrum analyzer is nothing more than a radio receiver. Get an FM radio with a built-in reception level indicator (or connect an AC voltmeter somewhere before the FM demodulator), gently sweep the frequency button, plot the signal strength against frequency, and you’ve got a spectrum analyzer. Figure 1a shows the internal architecture of a classic swept RF spectrum analyzer. The input signal first passes through an attenuator in order to avoid any saturation in the initial stages, and then it’s low-pass-filtered to reject out-of-band signals. Then the signal is down-converted to a fixed intermediate frequency through several heterodyne mixing stages. [Remember that multiplying two signals enables you to get the sum and difference frequencies. Then one of the two can be filtered out.] These down converters must be driven by very low-noise local oscillators in order to get a clean measurement. That’s why spectrum analyzers are not as simple as they look. The first local oscillator is swept to get a plot around the frequency of interest and with a frequency span set by the user.

The next critical component of a spectrum analyzer is the resolution bandwidth filter (RBW). It