

# Learning to PIC with a PIC-EL — Part 1

*This easy to duplicate prototype board for peripheral interface controllers will make it easy to get to know them!*

**Craig Johnson, AA0ZZ**



ave you ever wondered what makes all the little devices around your house that seem to have a mind of their own tick? You set the controls for your electric coffee pot, microwave oven, conventional oven, heating/cooling thermostat, DVD player, car's temperature controls, mileage indicators, trip odometer, etc. You look at your fancy new radio that you just bought and realize that there must be a lot of "brain power" hidden inside of it, too. How can you change bands, change operating modes, save a frequency in memory and change the speed of the CW keyer so quickly and effortlessly? In the past, devices like these were controlled by discrete electronics and later by integrated logic devices. Today many of these devices are controlled by little microcontrollers. They are simple and inexpensive. With a little effort you can learn how these microcontrollers work and how to put them to work in a new application that you dream up.

## Ready to Try?

Peripheral interface controllers (PICs) have been around for many years and are still very popular, in spite of the arrival of a number of new microcontroller families. One of the reasons PICs are so popular is because there are so many examples of projects that use them. Unfortunately, much of the code is complicated and hard to understand, especially for beginners. On the other hand, with a bit of determination and perseverance, trademarks of amateurs today, many have found PICs to be very useful in a variety of applications such as CW keyers, frequency counters, direct digital synthesis (DDS) controllers, receivers and transmit-

ters controllers, repeater controllers, power meters, antenna controls and station control. The list seems endless.

What is the difference between a microcontroller such as a PIC and a microprocessor such as a Pentium in a personal computer (PC)? One major difference is that a microcontroller could be thought of as a complete stand-alone computer on a chip, since it contains a central processing unit (CPU), memory and input/output (I/O) interface. On the other hand, a Pentium microprocessor in a PC requires external memory modules and I/O processing components before it can function as a computer or controller. This is not to say that the PIC is better or more powerful, of course, but each type must be evaluated and used for its intended purpose. A PIC microcontroller is very sufficient and capable of controlling many wonderful Amateur Radio applications.

A wide variety of powerful, inexpensive PIC microcontrollers is readily available. They come in a variety of sizes, capabilities and configurations. Some have *flash* memory that can be programmed multiple times, while others are intended to be programmed just once. If there is a reason to protect the code from being read from the PIC by users, it can be protected by setting a configuration bit during programming.

This article is intended to show how easy it is to understand how to use a PIC microcontroller. When you buy a PIC microcontroller it has no instructions or code stored in it. You need to write the code (usually with an ordinary text editor) in one of several special programming languages, process the text with an *assembler* or *compiler* to create PIC machine instructions and store these machine

instructions in the PIC.<sup>1</sup> The scope of how to write the PIC code is beyond the scope of this article. Later we will discuss an online course by John McDonough, WB8RCR, that shows how to write PIC code. It's called *Elmer 160*.

Once the PIC code is written it can be loaded into the PIC in the PIC-EL board. After you have compiled or assembled your text file into PIC machine instructions you are ready to load it into the PIC microcontroller in the PIC-EL board. This requires a computer program (called programming software) as well as specific hardware, connected to the computer via some sort of hardware interface, to control the PIC as the code is being loaded into it. A variety of programming software packages are available and described in Section 3 or in the *Elmer-160* course.

By looking at the various PIC-EL hardware components and the sample code or *Elmer-160* lessons, the amateur can learn how to use microcontrollers in many interesting applications. The rewards are great.

## PIC-EL Hardware Description

Figure 1 shows the schematic of the PIC-EL board. The PIC-EL board has two parts — a PIC programmer and a test/demonstration portion.

### PIC programmer

The PIC-EL board contains the hardware functionality that is required to interact with the programming software to transfer the machine language instructions from the computer to the PIC microcontroller. In the case of the PIC-EL, an RS-232 (DB9F) serial port connection connects them.

<sup>1</sup>Notes appear on page 42.

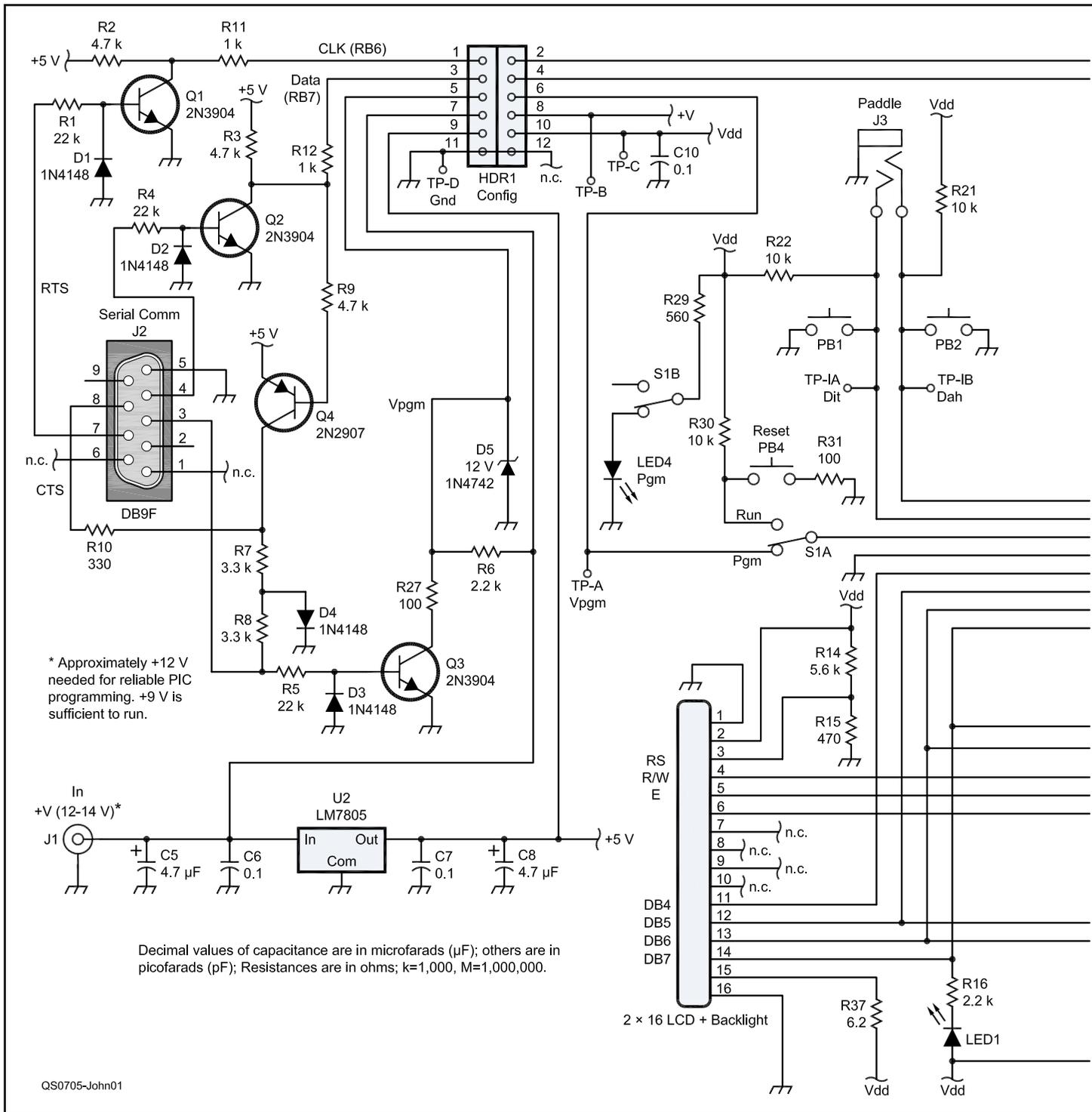


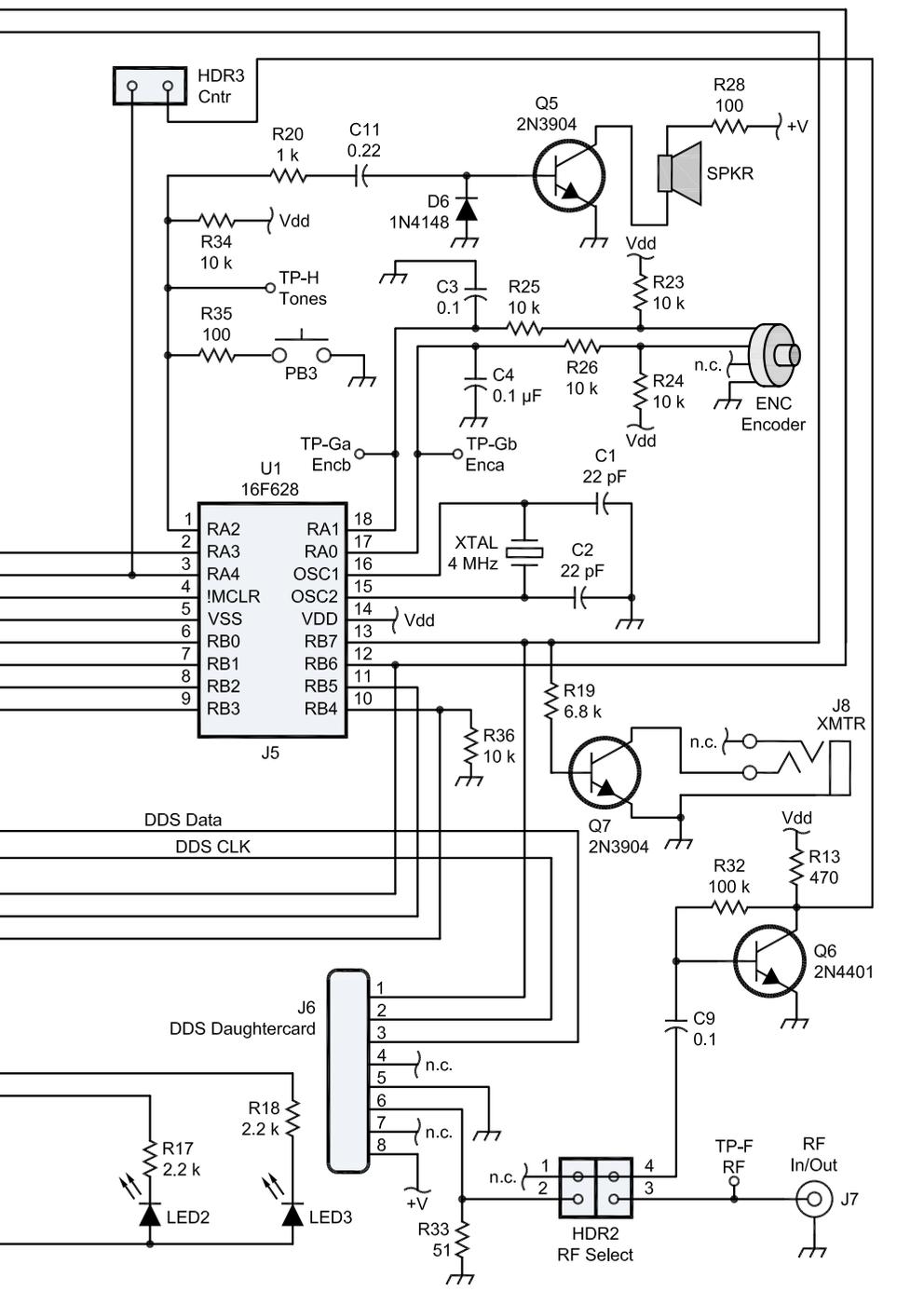
Figure 1 — Schematic of the PIC-EL board. The parts are listed in Table 1.

The PIC-EL's serial port programmer design is an adaptation of the classic programmer by David Tait. See Tait's PIC-related home page on the Internet, including downloadable software at [people.man.ac.uk/~mbhstdj/piclinks.html](http://people.man.ac.uk/~mbhstdj/piclinks.html). The software is also provided on the ARRL Web site with permission of the developer.<sup>2</sup> Details of how the RS-232 signals drive the PIC-EL hardware interface appear in Appendix A.<sup>3</sup>

No provision is made at this time for connecting a USB port or a parallel port to the programmer. However, external USB or parallel port programmers can be attached to the PIC-EL board via HDR-1. The PIC-EL board cannot be run directly from a PC parallel port because of the difficulty of getting the DATA IN signal to work properly. The parallel port's voltages are 0 V for a low level and +5 V for a high level. Since the TD

signal never goes negative, the circuitry that was used to pull the Q4 collector below zero for a low level does not work. In fact, the low level only goes down to approximately 3.2 V when TD is asserted (0 V). Additional circuitry could be added, but it is not included in this PIC-EL board.

How about a USB to serial converter between the computer and the PIC-EL's serial interface? It would be nice, but so far



applications. It allows the person to progress from controlling very basic components to more advanced components and projects.

In RUN mode, PIC experimenters have an opportunity to use and understand the following hardware functions:

- An 18-pin PIC microcontroller (16F84/A, 16F628/A, 16F88, etc). Includes a 4 MHz crystal.
- A 2x16 liquid crystal display (LCD with two lines of 16 characters).
- A rotary encoder (ENC-1).
- Three general-purpose push buttons, PB1 through PB3.
- A dedicated push button (PB4) for MASTER CLEAR (reset) of the PIC microcontroller.
- Three LEDs, LED1 through LED3.
- A speaker (SPKR-1) with transistor driver.
- All connections necessary to drive the NJQRP DDS daughtercard (DDS-30 or DDS-60).<sup>4</sup>
- A stereo jack for connection to CW paddles.
- A stereo jack with transistor driver for transmitter keying.
- A transistorized level conditioner for converting low-level signals to levels required for PIC input detection.
- A multi-purpose BNC connector.
- Selectable via a jumper at header HDR-2.
- Allow DDS output to be routed to the BNC
- Allow DDS output to be routed to a "conditioner" and then to a PIC input pin.
- Allow an outside signal source to be brought in to the "conditioner" and then to the PIC input pin.
- A 2x6 pin header block (HDR1 - CONFIG).
- Allows attachment of a "foreign programmer" to this PIC project board.
- Allows attachment of this programmer to a "foreign project board."

The PIC-EL schematic (Figure 1) may look quite complicated because many of the PIC pins have multiple usages. However, we can break down the schematic into its core pieces to understand the individual functions. This will also show how to use these basic components in other projects.

In Part 2 of this article we will discuss in detail the components connected to the PIC.

### PIC-EL Computer Interface

Now you are ready to program a PIC using the PIC-EL. Before you can do this you need to have a program installed in your PC. There are many good options available (including *IC-PROG* and *WinPIC*; see Figure 2) for the *Windows* operating system but we will concentrate on one good one, *FPP* by David Tait,

we have not found one that works in this application. The reason is that RS-232 ports are intended to perform asynchronous serial communications while we are directly controlling the serial port's control lines and doing synchronous communications to the PIC-EL. The PIC-EL requires a perfect one-to-one correspondence between signals generated by the computer programmer software and the signals presented to the PIC-EL serial

interface. Since USB-to-serial interfaces do not allow this closely controlled synchronous activity, it cannot interface properly with the PIC programmer and does not work.

### Project/Demonstration

The project/demonstration portion of the PIC-EL board was designed to allow the experimenter to understand how a PIC microcontroller can be used in a variety of

**Table 1****Parts for PIC-EL Board Project**

Resistors are all  $\frac{1}{4}$  or  $\frac{1}{2}$  W. Digi-key is at [www.Digi-key.com](http://www.Digi-key.com) and Mouser is at [www.mouser.com](http://www.mouser.com).

C1, C2 — 22 pF, ceramic disc capacitor (Digi-key P4841-ND, Mouser 140-50N2-220J).  
 C5, C8 — 4.7  $\mu$ F, electrolytic capacitor (Mouser 140-XRL16V4.7).  
 C3, C4, C6, C7, C9, C10 — 0.1  $\mu$ F, monolithic capacitor (Digi-key P4910-ND, Mouser 80-C317C104M5U).  
 C11 — 0.22  $\mu$ F, monolithic capacitor (Digi-key BC1149CT-ND).  
 D1, D2, D3, D4, D6 — 1N4148 diode (Digi-key 1N4148FS-ND, Mouser 625-1N4148).  
 D5 — 1N4742, 12 V Zener diode DO-41 (Digi-key 1N4742ADICT-ND, Mouser 78-1N4742A).  
 ENC — Rotary encoder (Digi-key P10860-ND).  
 HDR-1 — Pin header, 0.1", 2 x 6 position (Mouser 571-1032406).  
 HDR-2 — Pin header, 0.1", 2 x 2 position (Mouser 571-1032402).  
 HDR-3 — Pin header, 0.1", 1 x 2 position (Mouser 571-1032392).  
 J1 — Coaxial power jack, 2.1 mm (Digi-key SC1153-ND, Mouser 163-5004).  
 J2 — DB9F serial connector (Jameco-104951).  
 J3, J8 — Audio jack,  $\frac{1}{8}$ ", stereo (Mouser 161-3501).  
 J4 — SIP header, 16-pin (for PCB) (Mouser 571-16404526).  
 J5 — DIP socket, 18-pin (for PIC) (Digi-key ED3118-ND).  
 J6 — SIP socket, 8-pin, 90° (SamTec, SSQ-108-04-T-S-RA).  
 J7 — BNC jack, PCB mount (Mouser 523-31-5538-10-RFX).  
 LCD — Liquid crystal display, 2 x 16 character (08LCD9) ([www.ElectronicExpress.com](http://www.ElectronicExpress.com)).  
 LED1, LED2, LED3, LED4 — Red LED, T1-3/4 (Digi-key P374-ND).  
 P4 — SIP socket, 16-pin (for LCD) (Mouser 517-974-01-16).  
 PB1, PB2, PB3, PB4 — SPST push button, momentary — P8079SCT-ND.  
 PCB — PC board (KangaUS).  
 Q1, Q2, Q3, Q5, Q7 — 2N3904 NPN transistor, TO-92 (Digi-key 2N3904D26ZCT-ND).  
 Q4 — 2N2907A PNP transistor, TO-18 (Digi-key 497-2577-5-ND, Mouser 610-2N2907A).  
 Q6 — 2N4401 NPN transistor, TO-92 (Digi-key 2N4401-ND, Mouser 610-2N4401).  
 R1, R4, R5 — 22 k $\Omega$  (Digi-key 22KEBK-ND, Mouser 291-22K).  
 R2, R3, R9 — 4.7 k $\Omega$  (Digi-key 4.7KEBK-ND, Mouser 291-4.7K).  
 R6, R16, R17, R18 — 2.2 k $\Omega$  (Digi-key 2.2KEBK-ND, Mouser 291-2.2K).  
 R7, R8 — 3.3 k $\Omega$ , (Digi-key 3.3KEBK-ND, Mouser 291-3.3K).  
 R10 — 330  $\Omega$  (Digi-key 330EBK-ND, Mouser 291-330).  
 R11, R12, R20 — 1 k $\Omega$  (Digi-key 1.0KEBK-ND, Mouser 291-1K).  
 R13, R15 — 470  $\Omega$  (Digi-key 470EBK-ND, Mouser 291-470).  
 R14 — 5.6 k $\Omega$  (Digi-key 5.6KEBK-ND, Mouser 291-5.6k).  
 R19 — 6.8 k $\Omega$  (Digi-key 6.8KEBK-ND, Mouser 291-6.8k).  
 R21, R22, R23, R24, R25, R26, R30, R34, R36 — 10 k $\Omega$  (Digi-key 10KEBK-ND, Mouser 291-10K).  
 R27, R28, R31, R35 — 100  $\Omega$  (Digi-key 100EBK-ND, Mouser 291-100).  
 R29 — 560  $\Omega$  (Digi-key 560EBK-ND, Mouser 291-560).  
 R32 — 100 k $\Omega$  (Digi-key 100KEBK-ND, Mouser 291-100K).  
 R33 — 51  $\Omega$  (Digi-key 51EBK-ND, Mouser 291-51).  
 R37 — 6.2  $\Omega$  (Mouser 291-6.2).  
 S1 — Slide switch, DPDT (Digi-key SW102-ND).  
 Shunts for HDR1, shunt-1 shunt, 0.1", 2 position (Mouser 571-3828115).  
 SPKR — Speaker (Digi-key 433-1028-ND).  
 U1 — PIC16F628 microcontroller, with diagnostics pre-loaded (Digi-key PIC16F628-04/P-ND).  
 U2 — L7805 5 V voltage regulator (Digi-key 497-1442-5-ND, Mouser 511-L7805 ABV).  
 XTAL — Crystal, 4 MHz (Digi-key X405-ND, Mouser 520-HCU400-20).

G0JVY, since it is readily available and very easy to use. Similar packages are available for the Linux and Macintosh operating systems.

**Installing FPP**

You can download *FPP* directly from David Tait's Web site or obtain it from the ARRLWeb.<sup>5,6</sup> If you are using *Windows 95* or *98* you can run *FPP* without any additional drivers. However, if you are using *Windows 2000* or *Windows XP* you need to install two drivers, *directio* and *loaddrv*, to allow *FPP* to access the COM port. These two drivers are available at many Internet sites including

the AmQRP Elmer-160 Web site.

Detailed instructions for installing *FPP* are available by downloading John's *Elmer-160* Lesson 10 from the AmQRP Elmer-160 Web site.

**Using FPP**

A serial cable is needed to connect the PC to the PIC-EL board. You need a standard straight-through 9-pin DB9M (male) to 9-pin DB9F (female) connector. One end of the cable plugs into the RS-232C serial (COM) port on your PC and the other end plugs into the PIC-EL board. You may need

to disable other programs also using that same serial port.

Be sure that you have a sufficiently high voltage power supply connected to your PIC-EL board. Although the board can run off a 9 V battery, you will need to have at least +12 V at the PIC-EL CONFIG header (HDR1) in order to generate the minimum programming voltage, called  $V_{pgm}$ .

**Setup**

Once the serial cable is connected, you've got *FPP* loaded and turned on, your PIC-EL board is powered by at least 12 V, and you've manually wiggled the lines as described in Lesson 10, it should be a piece of cake to program a PIC on the PIC-EL board.

**Obtain the .HEX Program**

The HEX file is the new software you will be burning into the PIC. You can download the *PIC-EL V2* diagnostic program from my Web page or the FILES section of the PIC-EL Yahoo group. The *PICELv2diag.HEX* file contains the test program in HEX ASCII format. This file format is the standard output produced by PIC assembly programs and is a specific data format that is expected by the *FPP* program.

**Load the PICELv2diag.HEX File into FPP**

Click on the LOAD button and navigate to wherever you unzipped the PIC-EL V2 diagnostic files on your PC. You will see the *PICELv2diag.HEX* file listed there. Double-click on it and the HEX ASCII code will load into the *FPP* buffer. You will see that code in the *FPP* window as shown in Figure 3.

**Slide Mode Switch S1 Down to PGM MODE**

You need to move the slide switch S1 to the down position in order to put the PIC-EL board into the PGM mode. LED4, next to the switch, will turn on when you do this.

**Erase the PIC Currently Plugged into the PIC-EL Board**

You may want to erase, or clear out the software program currently in the PIC's flash memory before you burn a new program into the PIC. Click the ERASE button to do this, and if successful you will see a simple message pop up saying "PIC is erased."

**Burn the New Code into the PIC**

Now that the PIC memory is empty and you have the new program's HEX file in the *FPP* buffer window, you are all set to burn the program into the PIC. Click PROGRAM on the *FPP* application window and con-

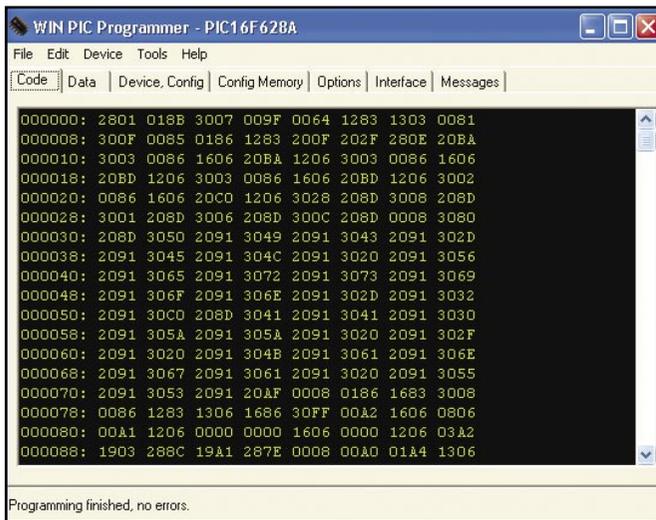


Figure 2 — PC screen snapshot running WinPIC software by DL4YHF to program a PIC in the PIC-EL II.

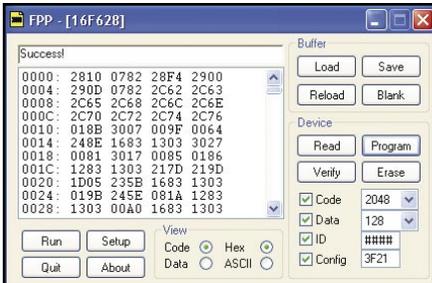


Figure 3 — PC screen snapshot running FPP software to program a PIC in the PIC-EL II.

firm your desire again in the pop-up window. It will take a few moments for this short program to be burned into the PIC, but when complete, FPP will display DEVICE PROGRAMMED! If it says PROGRAMMING FAILURE, you obviously have a problem such as the PIC was not first erased, the power supply wasn't connected or of sufficient voltage or the cable was not plugged in.

### Slide the Mode Switch Up to Go Into RUN Mode

Now that the programming is complete, you next need to put the PIC-EL board back into RUN mode. Do this by sliding MODE switch S1 up, and the PGM LED will turn off once again.

### Reset the Board to Start Up the New Program

Although not always necessary, just press the RESET button on the PIC-EL board to start the new program you just programmed into the PIC.

### PIC-EL Diagnostic Program

A PIC-EL diagnostic program is pre-

loaded on the 16F628 PIC that comes with the PIC-EL kit. The test symbolics file is also available on my Web site and on the PIC-EL Yahoo group FILES section. By looking at these symbolics and running the various tests you will be able to get an idea about how to activate the various elements of the PIC-EL. Note that the diagnostic program was written for demonstration purposes and not necessarily the way a finished program would be written.

When the test is started, the various test items will appear, one at a time, on line 2 of the LCD. When the test you want to run is displayed, press push button PB3. When you want to end a particular test you again press PB3 and hold it for ½ second. The main menu of tests will again start to appear, one at a time.

### Test LEDs

When this test is selected the diagnostic will light the three LEDs in this sequence.

- LED1 turns on and then off.
- LED2 turns on and then off.
- LED3 turns on and then off.
- LED1 and LED2 and LED3 turn on and then all turn off.

The PGM/RUN LED is not exercised in this test. It is turned on when switch S1 is in PROGRAM mode and off when in RUN mode.

### Test Push Buttons

This test will verify the operation of the three general-purpose push buttons. Pressing the push buttons in sequence will result in the following results:

- Pressing PB1 will cause LED1 to light. It will remain on until PB1 is released.
- Pressing PB2 will cause LED2 to light. It will remain on until PB2 is released.

- Pressing PB3 will cause LED3 to light briefly, but the test will then end.

This test does not exercise the RESET push button, PB4. Pressing PB4 resets the PIC and makes the test start over.

### Test Speaker

Selecting this test will cause a series of tones to come from the PIC-EL speaker. The test plays the eight tones of a one-octave diatonic scale starting with C and ending at the next higher C. By the way, this test uses classical music theory, starting with middle C, ending with the next higher C and tuned such that the A above middle C is tuned to 440 Hz. The accuracy is limited by the tolerance of the 4 MHz crystal. By my ear it's pretty close.

### Test Encoder

When this test is selected, the mechanical encoder is exercised. The test will start at zero with an eight digit number displayed on the LCD. Then, turning the encoder clockwise will increment the displayed number and turning the encoder counterclockwise will decrement the displayed number. The test uses the gray code signals generated by the encoder (see Figure 8) and determines the direction by the mechanism described.

### Test Paddles

CW paddles can be connected to the PIC-EL via a patch cord that plugs into J3. The jack connects one of the paddle connections to the tip and the other to the ring. In the absence of CW paddles, you can exercise the equivalent functionality by pressing PIC-EL push buttons PB1 and PB2. When the DIT paddle (or PB1) is pressed the test will light LED1. When the DASH paddle (or PB2) is pressed the test will light LED2.

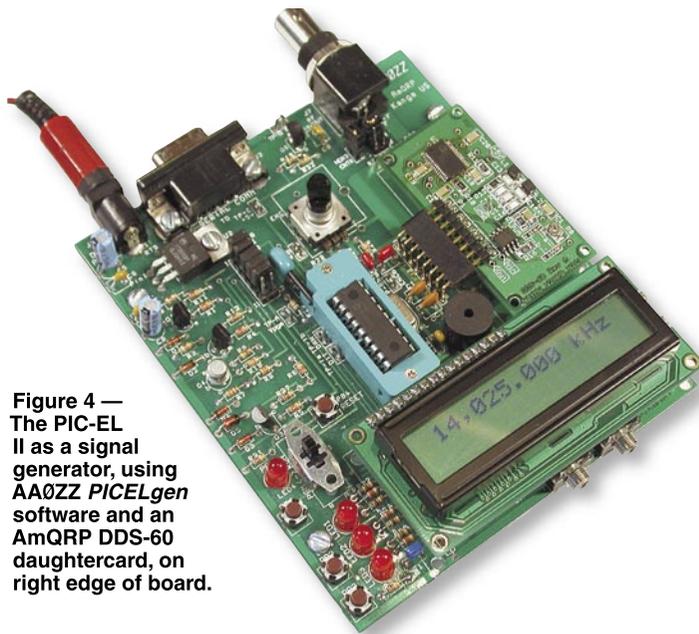


Figure 4 — The PIC-EL II as a signal generator, using AA0ZZ PICELgen software and an AmQRP DDS-60 daughtercard, on right edge of board.

## Test Transmitter Keying

This test will demonstrate how CW keying is accomplished via a PIC microcontroller. An oscillator or transmitter is connected to the PIC-EL board via a patch cord with an 1/8 inch mono phone plug. One end is plugged into J8 on the PIC-EL and the other is plugged into the oscillator/transmitter. The test activates transistor Q7, which effectively shorts the KEY LINE to ground and activates the oscillator/transmitter.

## Test DDS-30

If you have a DDS-30 DDS daughtercard from AmQRP organization, you can test it by selecting this test. The DDS-30 is connected to J6 of the PIC-EL. The test program generates three frequencies — 7.040, 7.041 and 7.042 MHz. (They are displayed on the LCD in Hz as 7040000, 7041000 and 7042000.) The duration of each frequency is about 1/2 second and then the sequence repeats. You can observe the output on a spectrum analyzer (or oscilloscope) or by tuning your HF receiver to 7.040 MHz. See Figure 4.

## Test DDS-60

The DDS-60 DDS daughtercard from AmQRP is slightly different from the DDS-30 in that the DDS-60 uses an AD9851 DDS while the DDS-30 uses an AD9850. The DDS-60 uses a 30 MHz clock oscillator and expects the 6x multiplier to be activated by driver software. Once again, the DDS-60 is installed in J6 of PIC-EL. When selected, this test generates the same three frequencies.

## PIC-EL for Elmer-160 Lessons

I developed the PIC-EL board in late 2004, with consulting help from AmQRP club members, George Heron, N2APB; Joe Everhart, N2CX; John McDonough, WB8RCR; Earl Morris, N8ERO, and Jim Kortge, K8IQY. This was soon after John McDonough volunteered to teach an online course to help beginners learn how to use PICs. We wanted to make a project board the students could use to load the code they just wrote into the PIC and then to immediately try it out. (Immediate feedback does wonders in keeping the experimenter motivated!) Within a three-month period I developed several prototypes, tested them, and finally produced a printed circuit board that the AmQRP club could use in a kit. Hundreds of these kits were distributed by the club in early 2005 and the reception was very positive.

Since John called his online PIC course *Elmer-160*, I named this board the PIC-EL. The standard PIC for the PIC-EL is a 16F628, a very common, mid-range PIC, selected for this project because of its balance of architectural simplicity, power and low cost. It is one of the most common PICs used in ama-

teur applications these days. The *Elmer-160* lessons start with an even lower-end PIC, the 16F84, because it is even easier for beginners to understand. The 16F628 is nearly compatible with the 16F84 in that only a few lines of code need to be changed. The PIC-EL can use either of these PICs interchangeably. However, the diagnostic program, preloaded on a 16F628 that is supplied with the kit, is slightly too large for a 16F84.

PIC-EL kit sales by the AmQRP club ended after a few months. Recently I updated the PIC-EL board to replace obsolete parts, correct a few deficiencies, and to make the kits available on a long-term basis. The result is the PIC-EL Version II. The kit is available for \$55 plus \$5 shipping from Bill Kelsey, N8ET, at Kanga US.<sup>7</sup>

If you are interested in the *Elmer-160* lessons from John McDonough, WB8RCR, you can find them at the AmQRP Web site.<sup>8</sup> John's lessons are available for downloading and you can contact him by using links provided on that site.

## Questions and Support

For up-to-date details and documentation regarding this project, please see the author's Web page. For additional support questions, see the Yahoo group PIC-EL or e-mail the author directly at the address below.

## Conclusion

It's very satisfying to be able to develop a PIC program that performs a task in exactly the manner you want it to. Yes, it takes a bit of effort, but the result is well worth it.

In the next part we will discuss in greater detail how to use the various hardware components attached to the PIC microcontroller in the PIC-EL.

## Notes

<sup>1</sup>Assemblers or compilers are software programs provided to convert human readable instructions into the firmware needed by the PIC.

<sup>2</sup>[www.arri.org/files/qst-binaries/Johnson0407.zip](http://www.arri.org/files/qst-binaries/Johnson0407.zip).

<sup>3</sup>See Note 2.

<sup>4</sup>[www.njqr.org/](http://www.njqr.org/).

<sup>5</sup>[www.people.man.ac.uk/~mbhstj/piclinks.html](http://www.people.man.ac.uk/~mbhstj/piclinks.html).

<sup>6</sup>See Note 2.

<sup>7</sup>[www.kangaus.com](http://www.kangaus.com), [n8et@arri.net](mailto:n8et@arri.net) or Kanga US, 3521 Spring Lake Dr, Findlay, OH 45840-9073.

<sup>8</sup>[www.amqrp.org/elmer160](http://www.amqrp.org/elmer160).

*Craig Johnson, AA0ZZ, holds a degree in electrical engineering from the University of Minnesota, and an MBA from the University of St Thomas. Since graduation he has worked on large-scale, mainframe computer design and development at Unisys Corporation in Roseville, Minnesota. He now works at Minnetronix, Inc in St Paul, Minnesota, developing microprocessor-based medical devices. He holds six patents related to computer disk technology and I/O architecture and has three more patents pending.*

*Craig got his first amateur license in 1964 at*

*the age of 14 and his General class license six months later. He credits Amateur Radio as being a key factor in his desire to become an electrical engineer. After college he let his license lapse and concentrated on computers. In 1995, two of his three children expressed an interest in Amateur Radio. Today, all five family members have ham licenses.*

*Craig has a variety of ham radio interests including DXing and contesting. He operates almost 100% CW but he enjoys the digital modes as well. He enjoys the excitement of building and operating homebrew QRP radios. He has designed and developed DDS VFOs using PIC microcontrollers that are being used by hundreds of hams throughout the world.*

*You may contact Craig at 4745 Kent St, Shoreview, MN 55126 or [aa0zz@cbjohns.com](mailto:aa0zz@cbjohns.com). His Web site is [www.cbjohns.com/aa0zz](http://www.cbjohns.com/aa0zz). *

## New Products

### W2DGJ WIRE ANTENNA WIND DAMPER

◇The W2DGJ Wire Antenna Wind Damper is said to work like an automotive shock absorber, putting pressure on the damper and not the wire when the wind blows and the antenna supports move. The Wind Damper attaches between the insulator at the end of your antenna wire and the supporting tree or pole and is said to allow up to 10 inches of movement before stressing the wire. One Wind Damper is required for each end of a dipole antenna. Price: \$29.95 each. For additional information, visit your favorite dealer or contact W2DGJ Insulators, 1300 Fatio Rd, Deland, FL 32720; email [tenpin11@hotmail.com](mailto:tenpin11@hotmail.com).

### QSO WIZARD SE AND PRO SOFTWARE FROM ORCASTAR

◇*QSO Wizard* ham radio operating assistance software for the Macintosh is now available in *Pro* and *SE* versions. Both versions include logging features, a custom QSL card maker, beam heading path plots, multiple clocks, a schedule editor and other features for the active operator. The software is built around images of the Earth. Worldwide prefixes, time zones, grid locators and other information is available by clicking on the image or searching included databases. The *SE* version uses graphics centered on the middle of your state. The *Pro* version uses custom graphics centered on your city or town, has a contact time clock that shows local time anywhere in the world, displays a real time "night shadow" showing which parts of Earth are in darkness, and other features. Available for Mac OS X only, *QSO Wizard* is customized for the user's call sign and QTH (North American locations only) at time of order. Price: *Pro* version, \$69.95; *SE* version, \$39.95. A 30-day demo version is available for a \$6.95 shipping/handling fee. More details are available at [www.orcastar.com](http://www.orcastar.com).