# Pickle with USB I/O

Update the PIC-EL to use with a new PC with a USB interface.

Craig Johnson, AA0ZZ
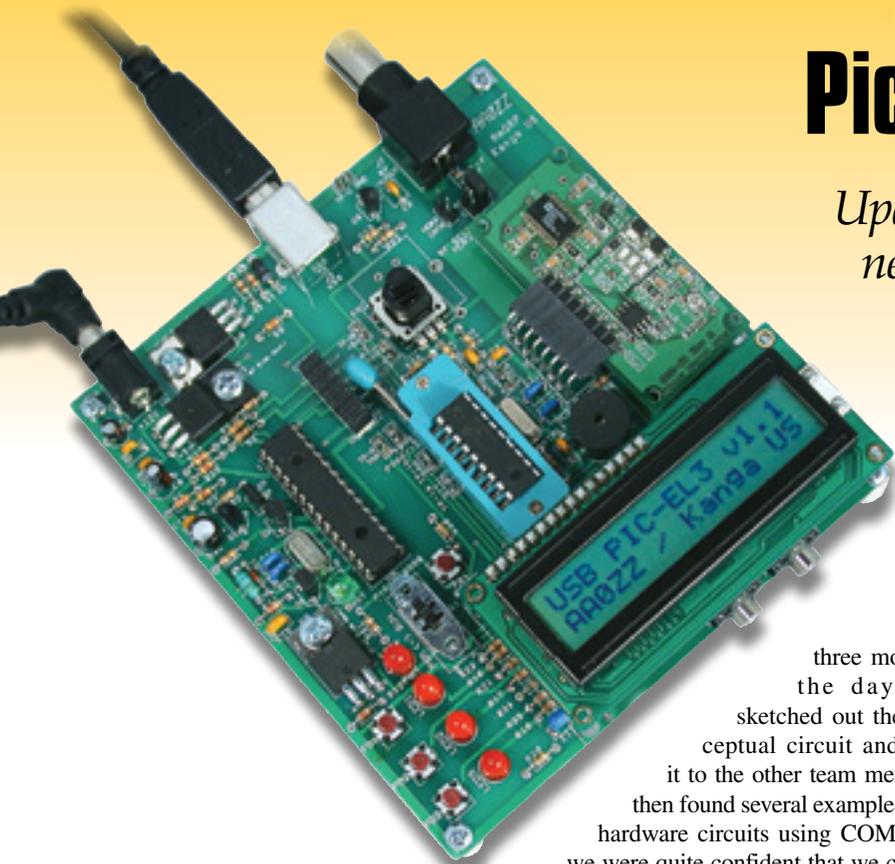
W hat do you mean it doesn't have a USB interface? I don't even *have* a COM port on my new computer!" Such was the cry from the very beginning from many people who wanted to learn to program PIC microcontrollers and then read about the COM port based PIC-EL board and the associated online PIC programming course that we developed.

The PIC-EL Revisited

In the May and June 2007 issues of *QST* I presented a two part article describing the PIC-EL. This project was developed to help hams get practical hands-on experience using PIC microcontrollers for some fun and useful ham applications.[1] As John McDonough, WB8RCR, began writing his lessons for the *Elmer-160* course, several of us who are associated with the AmQRP club began designing this companion board to allow the students to experiment with the material as they navigate through the lessons.[2] I decided to call it the PIC-EL ("pickle"), since it's for PIC microcontrollers and it goes along with the *Elmer-160* lessons.

Right from the beginning we knew that a USB interface would be a desirable interface for the tool. However, there were several problems. First of all, we had a very tight, self-imposed time schedule. We really wanted to get the board ready to go about

three months from the day I hand-sketched out the first conceptual circuit and e-mailed it to the other team members. We then found several examples of simple hardware circuits using COM ports and we were quite confident that we could make similar circuitry work in our application. To develop the circuitry for a USB interface, whether it was one of the integrated circuits designed for this type of task or if the circuitry was designed from scratch, it was obvious that a USB solution would be much more complicated.

Second, we didn't have a simple PC application to drive the programmer with a USB interface that we could use to send the low level (HEX) code into the PIC. We had several nice, free, PC applications that could drive the COM port circuitry we were looking at, however.

We carefully evaluated these reasons and concluded that the risks with pushing for a USB interface were too great. We decided to go with the simple COM port for the first version, with the intent of continuing to look for a USB solution for the future.

In my PIC-EL YAHOO group, users of the PIC-EL board discuss their projects and ask questions about PIC programming issues as well.[3] One of the subjects that frequently comes up is how to get the PIC-EL to work with a USB interface. While we tried using various COM-to-USB converters (they don't work, since we drive the COM port pins directly) I quietly kept looking for a good solution.

I investigated many different methods but each had severe drawbacks, including the need for me to write new programming software to support it. Then, in late 2007, I bought a Microchip PICKit2, thinking it might be a way to quickly attach to the PIC-EL with

a USB connection.[4] The PICKit2 would be connected such that it would bypass the PIC-EL's hardware programmer and connect directly into the PIC-EL's configuration header. My initial attempts to make this work failed, but several months later I discovered that Microchip had published a set of updates for the PICKit2 hardware. After I installed them, it worked perfectly on the PIC-EL II. I then discovered a number of "clones" on the Internet and, upon examining them, realized that a lot of the hardware in the Microchip PICKit2 is nice but not really needed for a PIC-EL environment. I then made and prototyped a stripped down version of the PICKit2 using ideas from other designs on the Internet but putting my own spin on it. After some extended debugging sessions the PIC-EL III was born.

PIC-EL III Hardware Description

The schematic of the new portion of the PIC-EL III board is shown in Figure 1. The "right side," the various hardware components that can be driven by the target PIC, has been changed very slightly since the PIC-EL II, but the "left side," the programmer portion, has been completely replaced and is shown in the figure. The complete schematic and parts list is available on the QST binaries Web site.[5]

PIC-EL III Computer (USB) Interface and Programmer

The programmer section has been replaced with circuitry to provide a USB interface. The circuitry is a simplified version of the PICKit2 by Microchip Technology (**www.microchip.com**) and has a Microchip PIC 18F2550 in its center. The code for the 18F2550 is produced and distributed (free of charge) by Microchip.

The PIC-EL III hardware programmer uses MOSFETs to drive the programming lines. It does not draw 5 V power from the USB con-
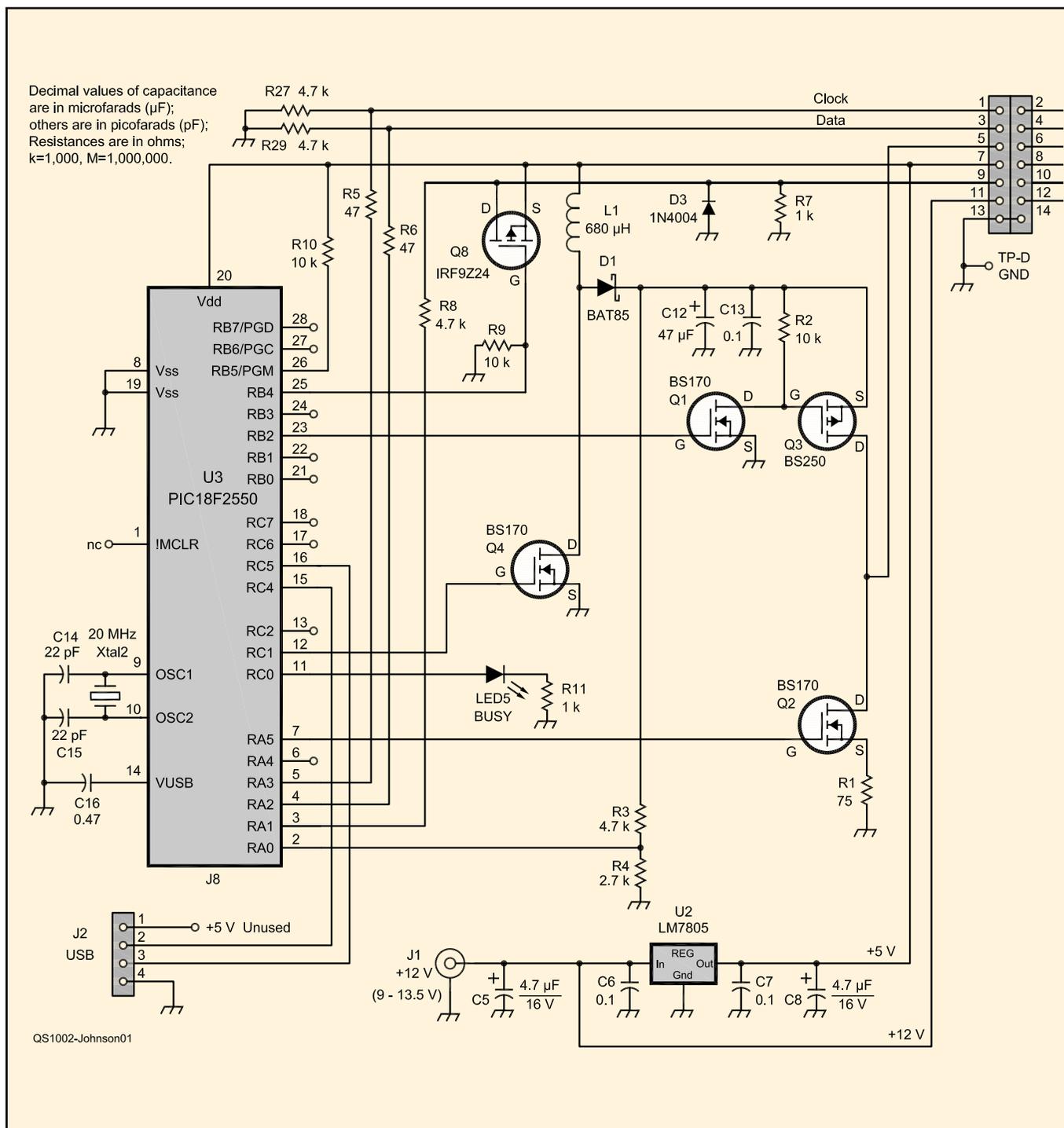
Figure 1 — Schematic diagram of the USB interface portion of the PIC-EL. Although the display subassembly is functionally equivalent to the version in May 2007 *QST*, there are a few differences in detail.

nection but instead runs on 5 V power from the PIC-EL's 12 V to 5 V regulator. The programmer hardware has charge pump circuitry to internally generate the +12 V programming voltage (V_{PP}) to be applied to the MCLR pin. Note that the PIC-EL expects 12 V power being supplied at all times and it does not use the 5 V power supplied by the USB. The PIC-EL's 12 V to 5 V voltage converter supplies all the 5 V power for the PIC-EL board.

Note that when the PIC-EL is in PROGRAM mode, the programmer hardware generates V_{DD} (+5 V) and V_{PP} (+12 V) with the appropriate timing to program the target PIC. Depending on the type of PIC being programmed, the programmer sometimes raises V_{PP} before V_{DD} and sometimes raises V_{DD} before V_{PP}. It's tricky but very important. (This removes a limitation found in the previous PIC-EL versions as well as all other *Tait*

type programmers that use a COM interface. There simply aren't enough pins in a COM port to do this.)

Project/Demonstration Portion

The project/demonstration portion of the PIC-EL board was described in great detail in the 2007 *QST* article so I won't repeat it here. A summary is on the QST binaries Web page, however.

Programming a PIC with the PIC-EL III

Code to be loaded into the PIC in the PIC-EL can be developed in many different ways. If you follow the lessons in the *Elmer-160* course you will be able to understand the PIC architecture and command set well enough to write your applications in low level *machine language* code. Hey, it's not that hard. There are many examples of PIC code on the Internet, and you can easily collect pieces that are useful in your application. I also have several working examples on my Web page and in the FILES section of the PIC-EL YAHOO group of code that works on the PIC-EL.[6,7] There are a simple CW keyer, a frequency counter and a signal generator using the DDS-30 or DDS-60. The code for these sample applications is written in a form that makes it easy to understand (many comments) and can be used as a springboard for your own application. All are readable with a text editor. Go ahead and take a look.

Writing the Code into the PIC

One of the great benefits in having PICKit2 look-alike hardware is that it can be driven with the neat, stand-alone *Windows* application that Microchip freely distributes. Alternatively, it can be done with Microchip's freely distributed *MPLAB IDE* or their command-line version, *PK2CMD*. Note that Microchip also has versions of *PK2CMD* that run under the *Linux* and *Macintosh* operating systems as well.[8,9] With the PIC powered up and the USB cable connected, the PICKit2 application can be started. It will *connect* with the 18F2550 PIC in the PIC-EL and immediately report that it found a PICKit2. Figure 2 is a screenshot showing the PICKit2 application *connected* to the PIC-EL and ready to program a PIC. Programming a PIC is this easy:

■ Power up the PIC-EL

■ Connect the USB connector to the PC to the PIC-EL

■ Start the *PICKit2* application

■ Flip the PROGRAM / RUN switch in the PIC-EL to the PROGRAM position

■ Insert the PIC to be programmed in the PIC-EL socket

■ Select the PIC TYPE in the pull-down box in the *PICKit2* application

■ Import the code (.HEX format) from the location on your PC where it is stored

■ Press the WRITE tab

■ Wait for a SUCCESS message. The verify operation is automatic.

Running the PIC Program on the PIC-EL III

Now you are ready to try out the PIC program that you just loaded into the PIC. It's just a matter of flipping the PROGRAM / RUN switch to RUN position and the PIC program will start up. Now you can see the results of your labor. Fun, isn't it?

How quick is it? I can literally change the source code of a program, assemble it, write it into the PIC, flip the switch and try it out in about 20 seconds! That's why developing code with a PIC-EL is so much fun.

Options

Microchip's *PICKit2* Application runs on the *Windows 98 SE* platform and beyond. I mentioned Microchip's stand-alone *PICKit2* application because it's perhaps the easiest way to go. There are a few other possibilities, however. Microchip's *MPLAB IDE* can also be used as well as Microchip's command-line interface program called *PK2CMD*. Microchip has versions of *PK2CMD* for *Windows* as well as Linux and *Mac OS X*.

Questions and Support

For up-to-date details and documentation regarding this project, please see my Web page, **www.cbjohns.com/aa0zz**. For additional support questions, see the PIC-EL YAHOO group or e-mail the author.

Conclusion

It's very satisfying to be able to develop a PIC program that performs a task exactly the way *you* want it to. Using the PIC-EL to develop and test code is a very convenient and enjoyable way to do this. With the low prices for PIC microcontrollers these days, it's really easy to think of lots of ways to use them. In simple configurations, you don't even need a crystal, so it's easy to throw a micro into a simple circuit to accomplish a task that you have in mind. Yes, it takes a bit of effort, but the end result is well worth it. Once you get going, you will be amazed how many more applications you will dream up.

Notes
[1]www.amqrp.org/elmer160/index.html.
[2]G. Heron, N2APB; J. Everhart, N2CX; J. McDonough, WB8RCR; E. Morris, N8ERO; J. Kortge, K8IQY, and the author.
[3]www.groups.yahoo.com/group/PIC-EL.
[4]Microchip Technology Inc (www.microchip.com).
[5]www.arrl.org/files/qst-binaries.
[6]www.cbjohn.com/aa0zz.
[7]See Note 3.
[8]www.linux.com.
[9]www.apple.com/mac/.

ARRL member and Extra Class licensee Craig Johnson, AAØZZ, has earned BSEE and MBA degrees. He worked for Unisys for 35 years on the design and development of large computers and now works for Alliant Techsystems, a Defense Department contractor, developing microprocessor based products for the military. Craig holds seven US patents based on his work in computer hardware and software.

Craig got his first ham license in 1964 at the age of 14. He credits ham radio with sparking his interest in electronics and as a major factor in pointing him toward a career in electrical engineering. During and after his college years, however, he let his license lapse for several years and concentrated on computers.

For several years, Craig led a team of Volunteer Examiners (VE) and helped hundreds of people in the St Paul area get or upgrade their licenses. He still serves as a VE on occasion. He is an active member of the Minnesota QRP Society. Craig enjoys low power operating (QRP), DXing and contesting. He is happiest, however, when he is tinkering, building or experimenting with his new designs, circuits and software. His current interests are centered around projects that use microcontrollers, Direct Digital Synthesis and the new digital modes.

*You can reach Craig at 4745 Kent St, Shoreview, MN 55126 or at* **aa0zz@arrl.net**.
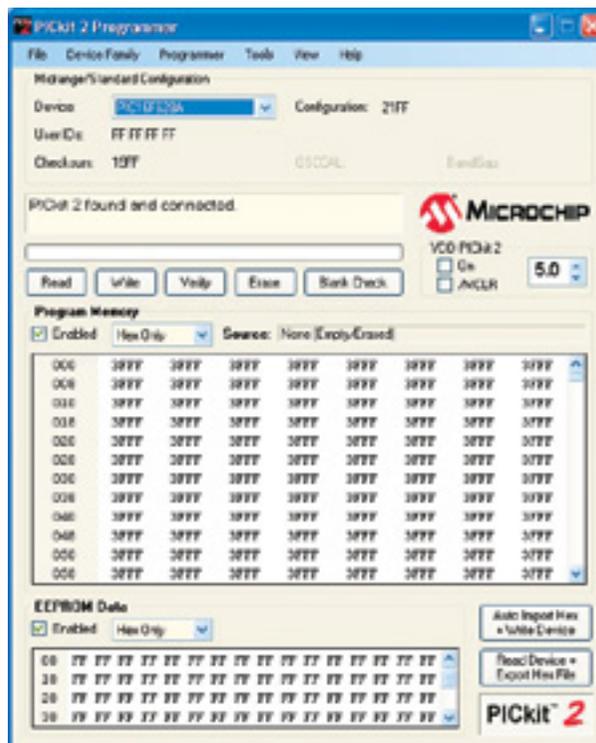
Figure 2 — Screenshot of Microchip *PICKit2* software screen.