Appendix

A4 QWIKBUG MONITOR PROGRAM

Burt Sims

A4.1 INTRODUCTION

QwikBug is a resident monitor program for the Microchip PIC18F452 microcontroller with its flash program memory and its built-in "background debug mode." QwikBug supports the downloading of a user program from the PC to the QwikFlash board. The CPU can be reset and the program then run from reset. For debugging, a breakpoint address can be entered and the program again run from reset to the breakpoint. The value of a variable or SFR can be modified before continuing. Selected variables and SFRs can be set up as watch variables and displayed in response to running again to the breakpoint or single-stepping from the breakpoint.

The installation of QwikBug into a PIC18F452 chip currently requires the one-time use of Microchip's ICD2 in-circuit debugger for programming the QwikBug code into the PIC18F452 chip, to enable the background debug mode and to set up the vectoring that causes the chip to respond to a power-on reset by ignoring a user program's reset vector at address 0x0000 and instead vector automatically into the QwikBug monitor code. This programming operation using the ICD2 module is discussed in Section A4.3.

As this book was being completed, David Flowers (creator of the QwikAddress utility described in Appendix A5) had begun working on a QwikProgram utility that could make use of a low-cost, widely available "hobby" programmer called the P16PRO PIC Programmer to program QwikBug into a PIC18F452. His QwikProgram utility is described in Section A4.4.

A4.2 RESOURCES USED

QwikBug resides in the high addresses of the program memory. Since the PIC18F452 has a huge program memory of 16,384 16-bit words, QwikBug's residence in upper memory will not cramp the space needed for most user programs. QwikBug shares the program stack with the user program, but since the PIC18F452 supports a large stack of 31 levels, this should never cause a user program to run out of stack space. QwikBug reserves the last 256 bytes of RAM (i.e., Bank 5) for its own use, leaving the remaining 1280 bytes of RAM for use by the user program. When a user program is downloaded to the PIC18F452, the code intended to set the configuration bytes (e.g., to select the oscillator type) is ignored. The configuration bytes will have already been programmed along with the programming of QwikBug into the chip. QwikBug takes over the on-chip UART, and thereby pins RC7 and RC6, to support serial communication with a PC at 19,200 baud. It also takes away pins RB7 and RB6 from the user. Microchip's ICD2 in-circuit debugger uses these two pins for its own special communication protocol with the chip. Because QwikBug uses the background debug mode built into the chip to support ICD2, these pins are lost to the user. The one use made of these pins by QwikBug occurs when executing a user program. With the J1 jumper in place on the QwikFlash board, a falling edge occurring as a character is received into the PIC18F452's UART is steered to the RB6 pin (see the schematic for the board, Figure A1-2). This causes the switch from user program execution to QwikBug monitor execution.

Whereas the MPLAB interface to the QwikFlash board via the ICD2 module supports the setting of watch variables with an internally displayed list, Tera Term Pro's interface to the QwikFlash board provides no such support. However, David Flowers' QwikAddress utility, discussed in Appendix A5, provides this support in a separate window. It also supports the setting of breakpoints by quickly helping to identify the program address of any subroutine and from there to any point in a subroutine. This utility is also a help to MPLAB users who have source code written with the structured assembly constructs of Chapter 6, since MPLAB deals with the .apr file that generated the assembler output files, whereas a user is more familiar with the original .asm file. QwikAddress deals with the .asm file and the .lst file to produce its concise lists of addresses.

Neither the user program's use of the reset vector address 0x0000 nor its interrupt vector addresses are affected by QwikBug. The template programs in this book show a "nop" instruction followed by a "goto Mainline" instruction at address 0x0000. These were inserted before it was fully understood how the background debug mode in this new chip would work. The earlier-generation PIC16F877 microcontroller's background debug mode did require a "nop" at address 0x0000. We postulated that we might have to copy the user program's reset vector code to a high memory address when downloading the user program and then initiate execution from there to run the user program from reset. Doing this would require a "goto Mainline" instruction rather than a "bra Mainline" instruction. The designers of the background debug mode for the PIC18F452 microcontroller allow the user program's reset vector to be downloaded and installed directly at address 0x0000 and executed from there, so none of our precautions turned out to be necessary.

A4.3 INSTALLATION OF QWIKBUG USING ICD2 MODULE

Download the QwikBug monitor from www.picbook.com. A new PIC18F452 chip will have its pins flared out, to support their use with automatic insertion equipment. After placing the chip on a conductive surface like a sheet of aluminum foil placed on a flat surface and touching the conductive surface with both hands to discharge any static electicity, rock one side of pins against the surface to bend them in, all at one time. Do the same on the other side of pins so that the two rows of pins are bent until they are parallel to each other, rather than flared out. With the power plug disconnected from the QwikFlash board, the PIC18F452 chip can be inserted into the 40-pin IC socket, with the chip's notch oriented toward the top of the board. Connect the QwikFlash board's power supply and turn on its power.

Connect Microchip's ICD2 in-circuit debugger to the PC with either its serial cable or its USB cable. If the serial-cable configuration is used, the ICD2 module must be powered by its own 9 V unregulated supply, included with the DV164007 packaging of the ICD2 module. Although the barrel connector for the QwikFlash board's 9 V unregulated supply looks almost identical to the supply that comes with the ICD2 module (and needed for the serial connection but not for the alternative USB connection to the PC), they actually differ in the inside diameter of the "barrel." Electrically they could be interchanged, but mechanically they cannot.

Connect the 9″ cable with modular plugs on each end between the ICD2 module and the QwikFlash board's CON1 connector. Open the MPLAB program installed on the PC. Click on the **Options** pulldown menu and then click on **Development Mode. . . .** Select PIC18F452 from among the choices in the **Processor** window. Select **MPLAB ICD Debugger** and click **OK**. This will open a window labeled MPLAB ICD 2. Click on the **Advanced . . .** button to open the **Configuration Bits** options, the **Programming Options**, and the **Communications** options. In the MPLAB ICD 2 window, you should see a series of diagnostic messages: "Reading MPLAB ICD 2 module product ID; MPLAB ICD 2 Detected; Reading MPLAB ICD 2 firmware version." If you do not get this message, then check that the **Communications** section is correct for your serial or USB cable connection. If you are using the serial cable and communication fails at 57,600 baud, select 19,200 baud and click on the **Reconnect** button to try again.

Having established the connection between ICD2 and the PIC18F452, click on MPLAB IDE's **Window** pulldown menu and click on **Program Memory**. Click on

File → Import → Import to Memory

and go to the directory holding the QwikBug.hex file downloaded from www.picbook.com and select it. Click **OK**. This will download two programs into the **Program Memory** window. One is the code for QwikBug, residing at high memory addresses. The other is a little program called ModifyBDMvector.

Return to the **MPLAB ICD 2 Advanced** window. If the configuration bits have not already been set by the downloading of QwikBug, then select the following:

Oscillator	HS
Watchdog Timer	Disable
Power Up Timer	Enable
Stack Reset	Enable
Osc. Switch Enable	Enable
Brown Out	Enable
Brown Out Voltage	4.5V
Low Voltage Program Disable	
CCP2 Mux	RC1

In the **Programming Options** section, enter

Start Address	0x0000
End Address	0x7bff

This "End Address" will be above the end of the QwikBug code and below the section of memory reserved for ICD2 when **Enable Debug Mode** is selected.

Select **Program Memory** and **Enable Debug Mode**. Click on **Program** at the bottom of the **MPLAB ICD 2 Advanced** window. Programming will take a minute or so. Upon completion the **MPLAB ICD 2 Version . . .** status window should show a series of messages, ending up with ". . . Programming Debug Executive; . . . Programming Configuration Bits; . . . Program Succeeded." (As a further error check, make sure that the "Error" LED on the ICD2 module has not turned on.) This step has enabled the background debug mode used by ICD2 and used by QwikBug when the ICD2 module is no longer attached to the QwikFlash board.

With background debug mode enabled, the PIC18F452 will subsequently come out of power-on reset into the ICD2's "Debug Executive." The next step is to change this so as to come out of power-on reset into QwikBug. In the **MPLAB ICD 2 Version . . .** window, press the **Reconnect** button. In the **MPLAB IDE** window, click on

Debug → Run → Reset

Make sure that the **MPLAB ICD 2 Version . . .** window's last message is "Resetting Target." If, instead, it indicates that this operation failed (or if the ICD2 module's Error LED has turned on), then click on the **Reconnect** button and try again. Then

Debug → Run → Run

The right LED on the QwikFlash board should turn on immediately as a sign that it has completed its job successfully. Immediately remove the modular cable from the QwikFlash board while the program is still running (as indicated by the yellow strip across the bottom of the **MPLAB IDE** window). Then turn off the power to the QwikFlash board. Exit from **MPLAB**.

By letting ICD2 run what it thinks is a user program that it has loaded into program memory, it will actually run a little "ModifyBDMvector" program that will do nothing more than rewrite the power-on vectoring into the QwikBug monitor program rather than into the ICD2's Debug Executive program.

Finally, remember to install the J1 jumper so that control of the board from the PC is not lost once the PIC18F452 chip begins execution of a user program. A press of any PC key returns control to the QwikBug monitor program.

A4.4 INSTALLATION OF QWIKBUG USING THE P16PRO PIC PROGRAMMER

Before ordering this programmer, check www.picbook.com for David Flowers' successfully completed QwikProgram utility. The free utility will program PIC18F452 parts with any program, including QwikBug, using the P16PRO PIC Programmer.

The P16PRO PIC Programmer is a "hobby" programmer that attaches to a PC through the parallel port and makes use of a wall transformer ("wall wart") to supply a high enough dc or ac voltage to generate both the +5 V and the +13 V supplies that it needs to program a part. It is widely available both as an assembled, working board and also in kit form (saving only $4). One source is Amazon Electronics (www.electronics123.com). Their toll-free number is 1-888-549-3749 (USA and Canada). The assembled board is their part CPA96, costing $17.95. The recommended 14 VAC wall transformer is their part BB041, costing $5.95. While some PCs have enough clearance to plug the board directly into the printer port, most users will probably require a DB-25 "straight-through" extension cable, available here as part BB040, costing $4.95. They ship from Ohio by priority mail. All told, the cost is about $30–35, depending on the need for the extension cable.

This programmer can also be used to program many other types of PIC microcontroller parts. A free programming utility is available through Amazon Electronics. A registered version is also available that removes the file-size limitation of the free utility.

To use this programmer, a PIC18F452 is inserted into the programmer and then the programmer is connected to the PC's parallel (i.e., printer) port, either directly or through a cable, but without power applied to the board. The QwikProgram utility is then launched. It begins by prompting for the desired hex file to be programmed into the part. If the QwikBug.hex file has been downloaded from www.picbook.com to the desktop, then it can be selected. It next requests that power be applied to the P16PRO PIC Programmer board followed by the press of any key. The utility then begins programming the part, providing feedback:

```
User program ................................................
Configuration bytes programmed
QwikBug vector programmed
```

The QwikBug hex file specifies the programming of the DEBUG bit in the CONFIG4L byte (see Figure 20-1). When QwikProgram detects this, it also programs a

```
goto QwikBug
```

instruction into address 0x200028, the BDM (background debug mode) vector address used at power-on reset, breakpoints, and single stepping. It is this last step that is difficult to achieve with standard PIC microcontroller programmers. Finally, the user is prompted to remove power from the programmer, disconnect the programmer from the PC, remove the PIC18F452 from the programmer, and insert it into the socket on the QwikFlash board.

A4.5 QWIKBUG VERIFICATION

Having exited from MPLAB, the PC's serial port is free for use by Tera Term Pro (or another terminal emulator). Set up Tera Term Pro with the serial port setup of Figure 5-8, described in Section 5.8. Connect the QwikFlash board to the PC via a serial cable and begin the execution of QwikBug by turning on the power to the QwikFlash board. You should see QwikBug's start-up "help" message. If a user program is already installed (e.g., the ModifyBDMvector program of Section A4.3), press any key on the PC within four seconds to enter QwikBug. Obtain the QwikFlash Performance Verification file, QFPV.hex, available at www.picbook.com. Using the F3 key, download the QFPV.hex file to the board. Then reset the chip (F4 key) and run the QFPV program (F7 key). This will establish that you can load and run programs satisfactorily. If the download is not completed successfully, increase Tera Term Pro's line delay to "20 ms/line," save this new setup, and try again. It is our experience that Tera Term Pro was developed several years ago and does not handle the line delay accurately for a present-generation fast computer. A setting of "20 ms/line" on this computer produced a line delay that varied somewhat from line to line but that was always more than the required 10 ms/line. There is nothing mysterious about this process that cannot be clarified by connecting a digital scope to the PIC18F452's RX input. By setting the scope to capture a single trace and starting the download, the gaps in the waveform corresponding to pauses between the PC's sending of each line of the .hex file can be examined to verify that these gaps are longer than 10 milliseconds. If not, then increase the line-delay parameter of the terminal emulator to meet this requirement.

A4.6 AUTOSTART FEATURE

If a user program is already resident in the PIC18F452 when power is applied to the QwikFlash board, then QwikBug will pause for about four seconds, waiting to see if a user wants to intervene by pressing any key on the PC. If so, then the QwikBug "help" screen will be written to the PC's screen followed by the QwikBug prompt:

```
QB>
```

If no key is pressed, then execution of the resident user code will begin.

A4.7 COMMAND KEYS

All of QwikBug's commands begin with the "QB>" prompt. The function keys, F1 to F9, can initiate each of the commands. The F1 key provides a brief help screen of the commands. Each command can also be initiated from the uppercase letter listed in the response to the the F1 key. For example, the "Modify" command can be initiated in response to the "QB>" prompt by either the F9 key or the "m" key.

For a command with a complex syntax, QwikBug displays a brief synopsis of the syntax. For example, in response to the F5 or "b" command to deal with a breakpoint, QwikBug displays

```
<aaaa> to set breakpoint, 'R' to remove breakpoint, 'D' to display breakpoint
```

followed by an

```
F5>
```

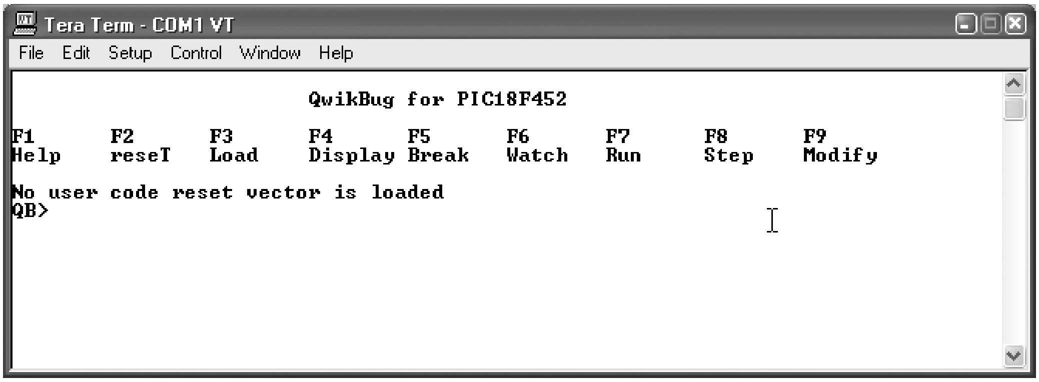
prompt, looking for either the entry of a breakpoint address, or the press of the "r" key to remove the breakpoint, or the press of the "d" key to display a previously entered breakpoint address (or a message indicating that no breakpoint address is set). In any case, in response to the command, a message indicates the result of the command followed by a new "QB>" prompt.

A4.8 Help COMMAND (F1 or H)

The F1 function key or the "h" key displays the QwikBug user menu. At any time during the running of the monitor, F1 can be pressed to display the list of function keys along with the operation of each key. Figure A4–1 shows the help message and the power-on start-up message that is displayed when no user code resides in program memory.

A4.9 reseT COMMAND (F2 or T)

Pressing the F2 or "t" key while at the "QB>" prompt executes the PIC18F452's reset instruction. This instruction affects all registers and flags in the same manner as would occur if the master-clear pin on the chip were toggled low and then high again by pressing and then releasing the QwikFlash board's RESET button (See "Other resets" in Figure 20-9.)



```
                        QwikBug for PIC18F452

F1      F2      F3      F4      F5      F6      F7      F8      F9
Help    reseT   Load    Display Break   Watch   Run     Step    Modify

No user code reset vector is loaded
QB>
```

Figure A4-1 Help message at power-on with no user code loaded

A4.10 Load COMMAND (F3 or L)

This command supports two actions:

The erasing of user program memory
The loading of a new user program into the erased memory

To erase the entire user program memory, just press the F3 key (or the "l" key) and then turn off power to the board. To download the "hex" file for a user program, press the F3 key. QwikBug will respond with

```
QB› User program memory erased - Send HEX file now
```

Tera Term Pro supports the sequence

```
File → Send file... → ‹select drive, folder, hex file›
```

to select the file. Alternatively, and more easily, if a desktop window is opened to the folder holding the hex file (e.g., C:\Work), then simply click on the hex file to select it and drag it into the **Tera Term Pro** window. In either case, QwikBug will respond by displaying a series of periods while the hex file is downloaded. Upon completion, it returns with

```
Download successful
```

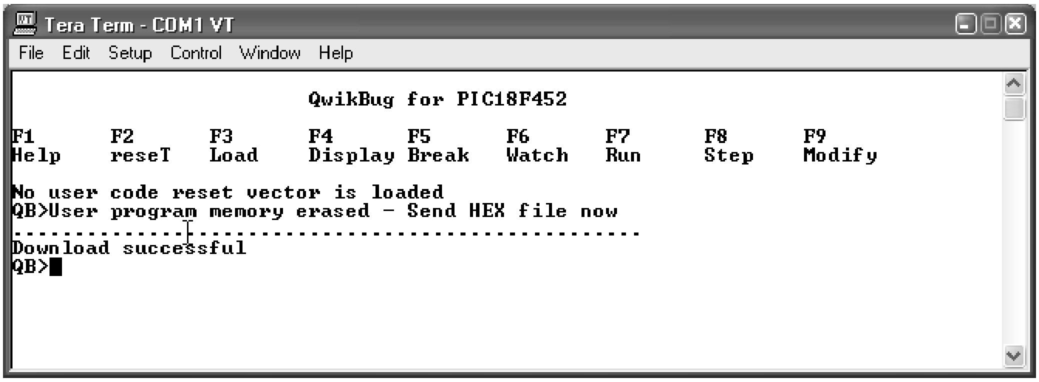
followed by a new "QB>" prompt, as shown in Figure A4-2.

If the download is unsuccessful, try again. If still unsuccessful, check that the Tera Term Pro's setup is correct with

```
Setup → Serial port... →
```

Compare the setup with that of Figure 5-8, with the possible exception of the line delay mentioned in Section A4.5. That is, because of a Tera Term Pro quirk, the line-delay parameter may need to be entered as a larger value than "10 ms/line" in order to obtain an actual pause after each line of the hex file of at least 10 milliseconds.

If the downloaded file does not execute in response to the Run command, check that the source file includes the reset vector at address 0x0000. That is, the user program should begin at address 0x0000, perhaps with a "goto Mainline" instruction, a "bra Mainline" instruction, and perhaps preceded by a "nop"



```
                         QwikBug for PIC18F452

F1      F2      F3      F4      F5      F6      F7      F8      F9
Help    reseT   Load    Display Break   Watch   Run     Step    Modify

No user code reset vector is loaded
QB>User program memory erased - Send HEX file now
...................................................
Download successful
QB>
```

Figure A4-2 Response to load command

instruction. For a user program that employs interrupts, the normal interrupt vectors must be present at 0x0008 and/or 0x0018.

A4.11 Display COMMAND (F4 or D)

This command is an adjunct to the Step command or the Run (to breakpoint) command. The first time that a step is taken, the watch variable labels (i.e., PC, W, etc.) are listed, followed on the next line by the state of each watch variable. Each subsequent step results only in the display of the state of each watch variable. After many steps, the labels will have scrolled out of view. Pressing the F4 key or the "d" key, rewrites the watch variable labels.

The same behavior occurs when a breakpoint has been set and successive presses of the Run command return to QwikBug with a display of the watch variables. When the watch variable labels have scrolled out of view, execute the Display command to rewrite the labels.

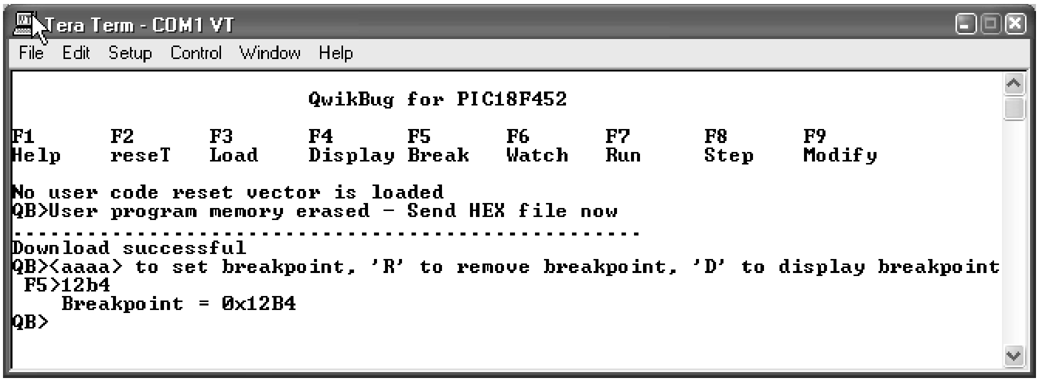
A4.12 Break COMMAND (F5 or B)

See Section A4.7, where this command was described as an example of how commands operate. The QwikAddress utility, described in Appendix A5, can be used to get the hex address of a desired location for a breakpoint by double-clicking on a subroutine name in one window. This opens a second window showing the source file with the subroutine name highlighted. Move to the desired line in the subroutine and double-click on it to obtain its address. The execution of the Break command is illustrated in Figure A4-3.

A4.13 Watch COMMAND (F6 or W)

The following registers and bits are automatically displayed in response to each Step command (F8) or to the stop at a breakpoint in response to the Run command (F7):

PC W Z C N FSR0 FSR1 FSR2



```
                        QwikBug for PIC18F452

F1      F2      F3      F4      F5      F6      F7      F8      F9
Help    reseT   Load    Display Break   Watch   Run     Step    Modify

No user code reset vector is loaded
QB>User program memory erased - Send HEX file now
.................................................
Download successful
QB><aaaa> to set breakpoint, 'R' to remove breakpoint, 'D' to display breakpoint
 F5>12b4
    Breakpoint = 0x12B4
QB>
```

Figure A4-3 Response to break command

Additional RAM variables or SFRs can be added to this list. Because of its many useful variations, this is the most complex of the commands. QwikBug expects a hex address to be entered for a RAM variable or a Special Function Register. The QwikAddress utility described in Appendix A5 supports this process by listing within one window the addresses of all RAM variables and within an alternative window all of the SFRs.

If nothing further is entered, then the variable will be displayed as a hex byte. If that variable is already set up as a watch variable, then it will be removed. For example, if the variable with address 0x012 has not already been set up as a watch variable, then the entry

```
012
```

will set it up, to be displayed as a 1-byte number, expressed in hexadecimal code.

If the hex address is followed by a space and then a "b", the variable will be displayed in binary format. For example, the entry of

```
012 b
```

will set up the contents of address 0x012 as a watch variable, to be displayed as an 8-bit binary number.

If the hex address is followed by a space and then an "a", then the variable will be treated as an ASCII value. For example, the entry of

```
012 a
```

will set up the contents of address 0x012 as a watch variable, to be displayed as a single displayable ASCII character. If the variable does not represent a displayable ASCII character, then the variable will be represented by "*".

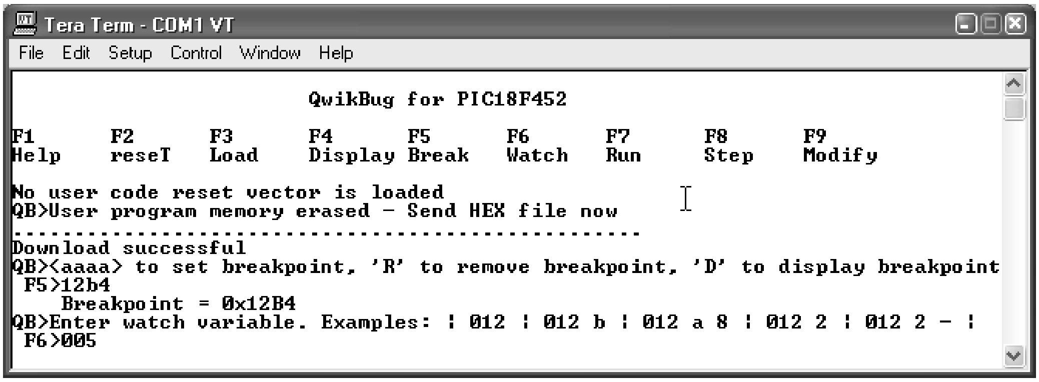
Adding a space followed by a digit *N* (ranging from 2 to 8) after either the preceding hex or ASCII entry will result in the display of *N* bytes, beginning at the entered address. For example, the entry of

```
012 a 8
```

will set up the contents of eight addresses beginning at 0x012 to be displayed as an ASCII string.

The entry of

```
012 2
```



```
                        QwikBug for PIC18F452

F1      F2      F3      F4      F5      F6      F7      F8      F9
Help    reseT   Load    Display Break   Watch   Run     Step    Modify

No user code reset vector is loaded
QB>User program memory erased - Send HEX file now
...................................................
Download successful
QB><aaaa> to set breakpoint, 'R' to remove breakpoint, 'D' to display breakpoint
 F5>12b4
    Breakpoint = 0x12B4
QB>Enter watch variable. Examples: ¦ 012 ¦ 012 b ¦ 012 a 8 ¦ 012 2 ¦ 012 2 - ¦
 F6>005
```

Figure A4-4 Response to watch command

will set up the contents of addresses 0x012 and 0x013 to be displayed as a 2-byte hex number, with 0x012 being treated as the most-significant byte.

Following this with a space and a "-" will result in the bytes being displayed in reverse order. Thus, the entry of

```
012 2 -
```

will set up the contents of addresses 0x012 and 0x013 to be displayed as a 2-byte hex number, with 0x013 being treated as the most-significant byte.

An example of a watch variable entry is shown in Figure A4-4.

A4.14 Run COMMAND (F7 or R)

Execution of this command switches the CPU from the execution of the QwikBug monitor to the execution of the user program. If the user program has just been downloaded or if the Reset command has just been executed, then execution begins at address 0x0000. Otherwise, execution picks up where it left off when a breakpoint or a step or a press of a PC key switched the CPU from the execution of the user program to the execution of the QwikBug monitor.

A4.15 Step COMMAND (F8 or S)

This command switches the CPU from the execution of the QwikBug monitor to the execution of one instruction of the user program before bouncing back into the QwikBug monitor. Successive presses of either the F8 key or the "s" key will single-step through the user program, instruction by instruction.

A4.16 Modify COMMAND (F9 or M)

This command brings up the message:

```
QB>Display/Modify contents of RAM or register address:
<aaa> to Display, <aaa xx> to Modify
```

Both the address and the contents are expected to be expressed as hexadecimal numbers.

A4.17 MODIFYING QWIKBUG

Several minor changes can be made to the QwikBug source code (available at www.picbook.com) to alter the operation of the monitor. These changes can be made by altering the following constants found at the beginning of the source code file: BAUD_CONSTANT, STARTDELAY, and WATCHDEPTH. By altering the BAUD_CONSTANT (set to 32 for a baud rate of 19,200 and a crystal frequency of 10 MHz), the user can change the baud rate. Refer to Figure 18-4a. The STARTDELAY constant determines the number of seconds before QwikBug automatically starts. STARTDELAY is presently set to 4 for a four-second delay. WATCHDEPTH determines the maximum number of user-added watch variables that can be added to the watch list. It is presently set to ten watch variables.

A4.18 ACKNOWLEDGMENT

The features of QwikBug were defined by the following students during the spring of 2001. Since this was prior to the introduction of the "first silicon" PIC18F452 parts, the code they wrote executed on the PIC16F877, with Backbround Debug features only discussed but not implemented.

Thomas Backus
Rafael Ballagas
Brandon Cromer
James Eubanks
Darren Gerhardt
Mayuresh Gogate
Lawrence McDonald
Barreus Sims
Christopher Stephens

During the summer of 2001, Mayuresh and Larry translated the code to PIC18F452 code, getting it running on a PIC18C452, an EPROM program memory predecessor of the PIC18F452 that did not support Background Debug Mode features. In was not until August 2001 that we received several first silicon PIC18F452 parts. During the fall of 2001, I developed enough further modifications to support QwikBug as a "load and go" tool (i.e., the implementation of the F3 and F7 functions). Then during the spring of 2002, Rawin Rojvanit, with the help of Drew Maule, got the remaining features of QwikBug working, including a creative use of the Background Debug Mode's "FREEZE" feature to keep the PIC18F452's counters from running while in BDM and yet to overcome the FREEZE feature's stopping of the UART. The UART is needed in BDM to accept commands and return results to the PC. Stopping the counters in BDM is needed to support single-stepping through mainline code in the presence of enabled timer interrupts.

This development would not have been possible without the help of Al Lovrich, Greg Robinson, and Craig Miller of Microchip Technology. They supported our understanding of a subset of the BDM features built into the hardware of the chip to support their ICD2 module and needed by QwikBug.