HPC EXPLORER BOARD FAQS AND TROUBLESHOOTING GUIDE

HPC EXPLORER BOARD FAQS AND TROUBLESHOOTING GUIDE

Chapter 1. FAQs and Troubleshooting

1.1 INTRODUCTION

This document includes a collection of common issues and questions regarding the PICDEM™ HPC Explorer demonstration board.

1.2 FREQUENTLY ASKED QUESTIONS (FAQS)

FAQ-1 The demonstration code is sending garbage to my terminal program. What are the recommended settings?

The demonstration program requires that your terminal be set to 57600 baud, N, 8, 1.

FAQ-2 Where can I find information about the boot loader that is programmed on the HPC Explorer demo board?

Download Application Note AN851 from the Microchip web site. The pre-assembled HPC Explorer boot loader hex code, demonstration hex code and complete source code can be found in the “HPC Explorer V1_01.zip” file located on the HPC Explorer web page.

FAQ-3 Can I use a terminal program such as “HyperTerm” or “TeraTerm” to download my application code to the HPC Explorer board?

Unfortunately, no. You download your application code or reload the default demonstration code using the PIC18F/PIC16F Quick Programmer Windows Application GUI program. See **FAQ-4**. This Windows application is designed to communicate specifically with the boot loader that is factory programmed on the HPC Explorer demonstration board. See **FAQ-6** for the basic steps to perform this download.

FAQ-4 Where can I find information about PIC18F/PIC16F Quick Programmer Windows application?

Download Application Note AN851 and the accompanying AN851 source code zip file “00851.zip” from the Microchip web site. The zip file contains both the PIC18F/PIC16F Quick Programmer executable and its Visual Basic source code. Please note that this Visual Basic application is provided “as-is” and is not supported by Microchip. You are free to modify it for your own use.

FAQ-5 How do I start the boot loader?

The boot loader is started by performing the following steps:

1. While holding the MCLR button depressed, press and hold the RB0 button.
2. Release the MCLR button.
3. Release the RB0 button.

This method forces a system reset and invokes the boot loader to run instead of the user's application or demonstration program.

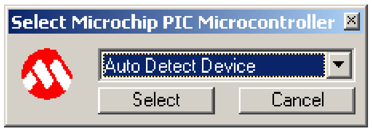
FAQ-6 How can I reload the demo program that came with the HPC Explorer Board?

The demo code is preserved in a .hex file and should be available from the Microchip web site on the HPC Explorer web page. You need to download the "HPC Explorer V1_01.zip" file and extract the file "PICDEM HPC 8722 V1_01_no_boot.hex."

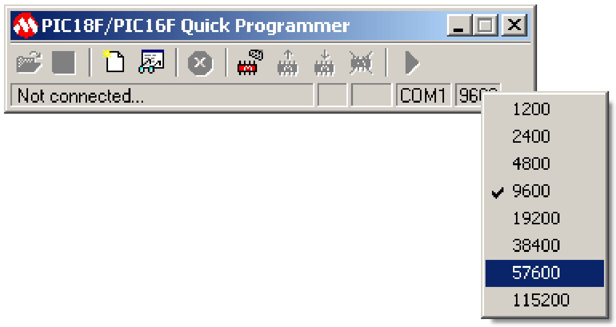
Steps to program demonstration code or user's application code:

1. Ensure that the boot loader is running. See **FAQ-5**.

2. Next, start the PIC18F/PIC16F Quick Programmer Windows application and click the "Select" button.

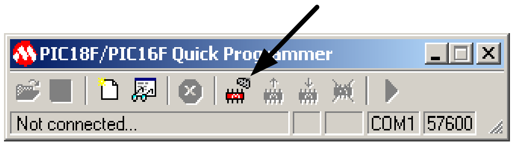


3. The boot loader may not connect the first time and you may see a "Not connected" message shown below. If the baud rate setting is 9600 or lower, or you are connected to a different COM serial port, be sure that you have selected the correct COM serial port and select 57600 baud as the communications baud rate. Right mouse click on the status bar to change these parameters as needed. Also, be sure a communications cable is connected to the HPC Explorer demonstration board. See **FAQ-7**.

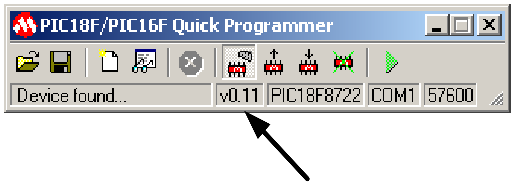


Not connected...

4. After verifying the connections and settings described above, click on the “connect” icon shown in the toolbar.

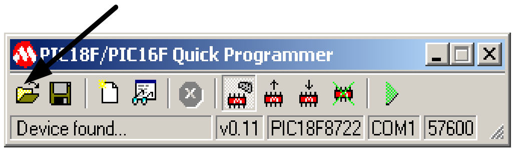


If the boot loader is running and communicating, you will see the PIC18F8722 and the boot loader version v0.11 in the status bar of the PIC18F/PIC16F Quick Programmer Windows application.

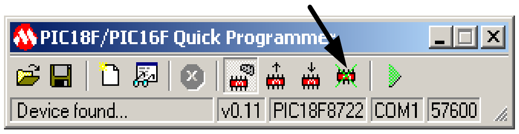


VERY IMPORTANT TIP: If you do not see the version number, the boot loader is not running or not communicating with the PIC18F/PIC16F Quick Programmer application. Note: the version is v0.11 at the time this document was prepared and may be different in the future. If you fail to see a version, see FAQ-5 and FAQ-9.

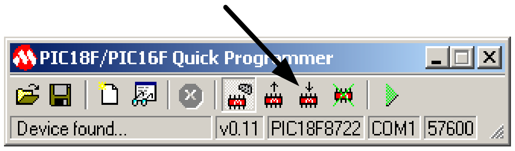
5. Select the demo or application hex file by clicking on the “File Open” icon in the toolbar as shown.



6. Before downloading the application code, make sure you erase the previous program memory contents by clicking on the “Erase” icon in the toolbar as shown.

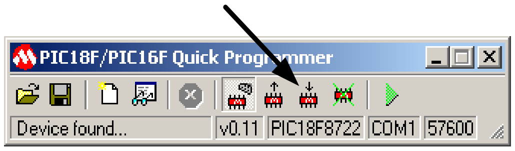


7. Program the hex code into the PIC18F8722 by clicking on the “Write” icon in the toolbar as shown.



8. To start the demo or application code, press and release the MCLR button. If you want to re-enable the boot loader, see **FAQ-5**.

   (Optional) To enable the demonstration code to run, click the green "Run" icon.



   After the green "Run" icon has been pressed, the boot loader will enable the demonstration or application code to operate.

   TIP: The optional method for starting the demonstration or application code is only required if using the "Boot chkEEPROM" boot entry method described in FAQ-13. The default boot entry method for the HPC Explorer mode is "Boot Button" and only requires a reset, (press and release MCLR), in order for the demonstration or application code to begin operating.

FAQ-7 How do I reprogram the boot loader and demonstration code that was originally programmed on my HPC Explorer board?

To re-program the boot loader, you will need an MPLAB® ICD 2. You will also need the file "PICDEM HPC 8722 V1_01.hex" that comes zipped in "HPC Explorer V1_01.zip." See **FAQ-2** regarding where to find this zip file. This pre-assembled file contains the boot loader and demonstration hex code.

1. Connect the MPLAB ICD 2 to the HPC Explorer demonstration board and launch MPLAB IDE.
2. From the MPLAB IDE menu, select File>Import. When prompted for the .hex file, select the "PICDEM HPC 8722 V1_01.hex" file.
3. Program this code into the HPC Explorer board. When complete, disconnect the MPLAB ICD 2 and you will see the demonstration code running and all LEDs are illuminated.
4. Follow the procedure in **FAQ-5** to start the boot loader.

FAQ-8 I have modified the demo code that comes with the HPC Explorer board for my own use. However, when rebuilding the "8722Demo.mcw" workspace, the following warning messages are generated:

```
C:\main.c:97:Warning [2058] call of function without prototype
C:\main.c:111:Warning [2058] call of function without prototype
```

There is a missing #include directive in the main.c file. Simply add the following #include directive just below the existing #include directives in main.c:

```
#include   "rtc.h"
```

FAQ-9 The boot loader is not responding or doesn't appear to be running. What can I do to fix this problem?

There are several possible causes. These are the three most common problems:

Problem 1: An incorrect COM port or low baud rate setting in the PIC18F/PIC16F Quick Programmer application. Right mouse click on the status bar windows of the application in order to select the appropriate setting. Select baud rate = 57600. You can permanently configure these settings by editing the following parameters in the P1618QP.ini file using a standard text editor, such as Notepad.

```
[PIC18FBOOT]
...
portindex=1       <---- Modify to change default COM port; 1=COM1, 2 = COM2, etc.
bitrateindex=7    <---- Modify to change default BAUD rate; 1=1200, 2 = 2400, etc.
```

Suggestion: Set bitrateindex = 7 makes default BAUD rate = 57600 for HPC Explorer Board

Problem 2: Not having an updated P1618QP.ini file containing the PIC18F8722 device information required by the boot loader. You can get the updated version from the Microchip web site on the HPC Explorer's web page. Or you can edit your existing P1618QP.ini file by adding the device ID and device information as shown below.

```
[DEVICELIST]
161="PIC18F8722"
...

[PIC18F8722]
writeblock=8
readblock=1
eraseblock=64
devicetype=1
maxpacketsize=128
bytesperaddr=1
pmrangelow="000800"
pmrangehigh="01FFFF"
eerangelow="000000"
eerangehigh="0003FF"
usrrangelow="200000"
usrrangehigh="20000F"
cfgrangelow="300000"
cfgrangehigh="30000F"
300001="CONFIG1H"
300002="CONFIG2L"
300003="CONFIG2H"
300004="CONFIG3L"
300005="CONFIG3H"
300006="CONFIG4L"
300008="CONFIG5L"
300009="CONFIG5H"
30000A="CONFIG6L"
30000B="CONFIG6H"
30000C="CONFIG7L"
30000D="CONFIG7H"
```

Problem 3: The boot loader has been erased from boot section in program memory. In this case, the only way to make the HPC Explorer demonstration board functional again is to reprogram the boot loader into the PIC18F8722 using an MPLAB ICD 2 In-circuit programmer. If you don't have one, try to borrow one or contact Microchip technical support requesting a replacement HPC Explorer demonstration board. For instructions to reprogram the boot loader, see **FAQ-7**.

FAQ-10 When rebuilding the “8722Demo.mcw” workspace, the following warning message is generated:

```
Warning[230] C:\CONFIG.ASM 29 : __CONFIG has been deprecated for PIC18 devices.
Warning[230] C:\CONFIG.ASM 30 : __CONFIG has been deprecated for PIC18 devices.
Warning[230] C:\CONFIG.ASM 31 : __CONFIG has been deprecated for PIC18 devices.
Warning[230] C:\CONFIG.ASM 32 : __CONFIG has been deprecated for PIC18 devices.
Warning[230] C:\CONFIG.ASM 34 : __CONFIG has been deprecated for PIC18 devices.
Warning[230] C:\CONFIG.ASM 35 : __CONFIG has been deprecated for PIC18 devices.
Warning[230] C:\CONFIG.ASM 36 : __CONFIG has been deprecated for PIC18 devices.
```

The version of MPASM assembler included with MPLAB 7.20 uses a new directive method for setting the configuration bits in source code. Since this workspace was originally created using an older version of MPLAB, it will be necessary to comment out the older __CONFIG directives and use the new CONFIG directive in the “config.asm” file as shown below.

```
;(DEPRECATED) __CONFIG  _CONFIG1H, _OSC_HSPLL_1H         <------ Old method
   CONFIG OSC = HSPLL                                    <------ New method

;(DEPRECATED) __CONFIG  _CONFIG2L, _BOREN_SBORDIS_2L & _BORV_43_2L &
_PWRT_ON_2L
   CONFIG BOREN = OFF
   CONFIG PWRT = ON
;(DEPRECATED) __CONFIG  _CONFIG2H, _WDT_OFF_2H
   CONFIG WDT = OFF
;(DEPRECATED)__CONFIG  _CONFIG3L, _MODE_MC_3L
   CONFIG MODE = MC
;(DEPRECATED)__CONFIG  _CONFIG4L, _LVP_OFF_4L & _DEBUG_OFF_4L &
_XINST_OFF_4L
   CONFIG LVP = OFF
   CONFIG DEBUG = OFF
   CONFIG XINST = OFF
```

An alternative method is to use the MPLAB C18 version of these directives by including them in a similar file named config.h as shown below. Note, you will have to remove the config.asm from the workspace and add a #include “config.h” directive to the main.c source file.

```
//CONFIG1L [300001H]
#pragma config OSC = HSPLL

//CONFIG2L [300002L]
#pragma config PWRT = ON
#pragma config BOREN = OFF

//CONFIG2H [300002H]
#pragma config WDT = OFF

//CONFIG3L [300003L]

//CONFIG3H [300003H]

//CONFIG4L [300004L]
#pragma config LVP = OFF
#pragma config DEBUG = OFF
#pragma config XINST = ON
```

For a complete list of the new PIC18F8722 configuration directives, check the “MPLAB C18 Complier Addendum” located in the MPLAB C18 web page on the Microchip web site.

FAQ-11 I want to design my own MPLAB C18 application and download it to the HPC Explorer board using the boot loader. Does the boot loader know where to place the resulting hex code in program memory during programming?

Unfortunately, no. Since the boot loader is located in the first 2K bytes (1K words) of protected program memory, you need to ensure that your code is properly configured to be located above the boot loader in program memory at location 0x0800. This can be done by modifying the linker script file. Example source code is provided in "8722Demo_no_boot.mcw" found in the "HPC Explorer V1_01.zip" file. See **FAQ-2**.

First, make a copy of the MPLAB C18 "18F8722.lkr" file found in **c:\mcc18\lkr** directory and save it in your local working directory. Add the copy of the linker script file to your project tree in MPLAB IDE project window. Edit the linker script as demonstrated below. The linker will use this configuration to link the compiled source code into the user program memory region above the protected boot loader. Note in this linker script example the MPLAB C18 start-up file c018i.o has been commented out preventing the linker from linking this object file to the project.

```
//FILES c018i.o         <-- Note this line is to be commented out or removed
FILES clib.lib
FILES p18f8722.lib

CODEPAGE    NAME=vectors     START=0x0      END=0x29        PROTECTED
CODEPAGE    NAME=boot        START=0x2A     END=0x7FF       PROTECTED
CODEPAGE    NAME=rvectors    START=0x800    END=0x829       PROTECTED
CODEPAGE    NAME=page        START=0x82A    END=0x1FFFF
```

Second, make a copy of the MPLAB C18 "c018i.c" source file found in **c:\mcc18\source\traditional\startup** and save it in your local working directory. Add the copy of the linker script to your project. Since the MPLAB C18 application code you are creating must be relocated to program memory address 0x0800, it is necessary to edit the code section "_entry_scn" definition in c018i.c file as shown below.

```
#pragma code _entry_scn=0x000800   <---- Note this must be changed to 0x0800 if using code
                                         protected boot block. Otherwise, use 0x200.
void_entry (void)
{
_asm goto _startup _endasm

}
```

FAQ-12 Is it possible to develop an application for the HPC Explorer board using the “extended” instruction set available in the PIC18F8722?

Absolutely. However, if you plan to use the boot loader that comes programmed on the HPC Explorer board, it is not coded to take advantage of the new PIC18F8722 “extended” instruction set. You will have to re-assemble the boot loader with the “Extended CPU” configuration bit enabled. This will also require you to re-program the boot loader in the PIC18F8722 using an MPLAB ICD 2.

Re-assembling the boot loader:

The boot loader was written in MPASM assembler and will require you to enable the “Extended Mode” option in MPLAB IDE for the MPASM assembler. Select from the MPLAB IDE menu, Project>Build Options>Project and select the MPASM Assembler TAB in the dialog box. Select the check box for “Extended Mode.” You will also need to set the configuration bit for the PIC18F8722 that enables the “extended” instruction set from the MPLAB IDE menu Configure>Configuration Bits or you can add the following configuration directive to the MPASM boot loader source code to automatically enable the extended instruction set.

```
CONFIG XINST = ON
```

You will have to re-program the new boot loader into the PIC18F8722. This requires an MPLAB ICD 2.

If you are not planning to use the boot loader and you are developing your application code using MPASM assembler, your application code can use the same configuration directive above to enable the extended instruction set. Since you won’t be using the boot loader to program your application code into the PIC18F8722, you will need an MPLAB ICD 2.

If developing your application code using MPLAB C18, you can use the C18 configuration directive shown here.

```
#pragma config XINST = ON
```

When using MPLAB C18, you will need to select the “Extended Mode” option in MPLAB IDE for the compiler to utilize the extended instruction set. Select from the MPLAB IDE menu, **Project>Build Options>Project** and select the MPLAB C18 TAB in the dialog box. Select the check box for “Extended Mode.” Finally, make sure you add the MPLAB C18 “18F8722e.lkr” linker script file to your project tree. This linker script file can be found in directory **c:\mcc18\lkr** and is required to support this mode.

FAQ-13 The boot loader source code appears to support more than one boot entry method at power-on reset. How are they different and when you would use one versus the other?

Yes, there are two power-on reset “boot entry” methods supported by this boot loader, depending on the options enabled when the boot loader is assembled. One method is “Boot Button”, the other is “chkEEPROM”.” The HPC Explorer board implements the “Boot Button” method.

The first method is the “Boot Button.” This method examines the state of button RB0 at power-on reset and if not pressed, RB0 = 1, a branch to the demo or user’s application code is executed.

Example MPLAB C18

```
if (!PORTBbits.RB0)
   {
         _asm
                clrf STKPTR, 0          // Reset the hardware stack pointer
                goto BootVector         // BootVector located at 0x0004
         _endasm
   }
```

If button RB0 is pressed, RB0 = 0, the boot loader is executed and monitors the RS232 port for a user application download.

PROS – The boot loader can be invoked by simply holding down RB0 at power-on reset or while causing a reset as described in **FAQ-5.** This method does not require the user’s code to perform any call to the boot loader or modify the EEPROM memory.

CONS – The target design requires an input switch or other input signal that can be monitored by the boot loader during power-on reset.

The boot loader can be re-assembled for either boot entry method. A simple #define in the “bootload.asm” file determines which method the boot loader will use. The following is an example for the “Boot Button” method.

Boot loader “Boot Button” entry conditional assembly – bootload.asm

```
;Uncomment the definition to use last EEPROM address to invoke Bootloader

;#define ChkEEPROMForBoot          <----- Be sure to comment out this #define
#define BOOT_BUTTON                <----- Define the “Boot Button” method here
...

Setup:
#IFDEF     BOOT_BUTTON             <----- The following two lines are assembled
   btfss   PORTB, 0
   bra     StartBypass
#ENDIF

#IFDEF ChkEEPROMForBoot            <----- The following three lines are not assembled
   clrf    EECON1
   setf    EEADR                          ; Point to last location
#IFDEF     EEADRH
   setf    EEADRH
#ENDIF
   bsf     EECON1, RD                     ; Read the control code
   incfsz  EEDATA, W
#ENDIF
   bra     RVReset                        ; If not 0xFF then normal reset
```

The second method is the “Boot chkEEPROM.” This method uses the last location in the PIC18F8722’s data EEPROM memory to store a non-volatile flag, where value = 0xFF indicates execute the boot loader and monitor the RS232 port for a download or, if value = 0x00, branch to the user’s application code.

PROS – Requires no user interface, such as a button press.

CONS – If your application code needs to pass control to the boot loader, you must provide a mechanism to reset EEPROM memory location or call the boot loader directly.

For more detailed information regarding this method, see **FAQ-14**.

FAQ-14 I want to use the boot loader on the HPC Explorer board to upgrade my application firmware. How does my application pass control to the boot loader?

There are two methods for an application to pass control to the boot loader. One method modifies the non-volatile boot flag stored in data EEPROM; the other method branches directly to the boot loader.

In the first method, suppose the boot loader is using the “chkEEPROM” boot entry method and your application has received a command through the RS232 port to prepare for a firmware update. Your application can setup the address in EEADR, EEADRH = 0x3FF, set the data in EEDATA = 0xFF and perform a call to the “StartWrite” vector located at 0x0002. Your application code must check the EECON1.WR bit. While this bit = 1, the PIC18F8722 is busy writing to EEPROM and you must wait. When this bit = 0, the write operation is complete and the “reset” command may be executed. See the following MPASM and MPLAB C18 examples.

Example MPLAB C18

```
void ResetEE(void)
{
   EEADRH = 0x03;
   EEADR = 0xFF;              // Load EEADRH,L = 0x3FF
   EEDATA = 0xFF;             // Load EEDATA = 0xFF

   EECON1 = 0x00;             // Clear the EECON register
   EECON1bits.WREN = 1;       // Enable the write operation

   _asm
   call    0x0002,0           // Call the boot loaders EEPROM “StartWrite”
                                 function
   _endasm

   While(EECON1bits.WR);      // Wait until EEPROM write operation is
                                 complete

   Reset();                   // Force a processor reset
}
```

Example MPASM

```
ResetEE:
   setf    EEADR               ; Point to location 0x3FF
   setf    EEADRH
   setf    EEDATA              ; Data "0xFF" will be written to this location

   clrf    EECON1              ; Clear the EECON register
   bsf     EECON1,WREN         ; Enable the write operation

   clrf    STKPTR              ; Reset the hardware stack pointer
   call    0x0002              ; Call the boot loader EEPROM write function

   btfsc   EECON1,WR           ; Wait until EEPROM write operation is complete
   bra     $-2
   reset                       ; Force a processor reset
```

Fortunately, the boot loader provides a function named “StartWrite” that performs the actual EEPROM unlock - write operation for you.

Bootload.asm – StartWrite Function

```
; Unlock and start the write or erase sequence.
; Vector for this function is located at 0x0002 in bootload.asm
StartWrite:

   movlw   0x55                   ; Unlock
   movwf   EECON2
   movlw   0xAA
   movwf   EECON2
   bsf     EECON1, WR             ; Start the write
   nop

   return                         ; Return to caller
```

In the second method, your application can pass control to the boot loader by branching to the boot loader’s “StartBypass vector at location 0x0004. Shown below are the branch vectors located at PIC18F8722’s power-on reset vector.

```
   ORG     0x0000
   bra     Setup
   bra     StartWrite
   bra     StartBypass     <-----  vector located at 0x0004
```

The following is an example of MPLAB C18 in-line assembly code preparing to branch directly to the boot loader “StartBypass” vector located at location 0x0004.

CAUTION! It is recommended that you clear the STKPTR register prior to branching to this vector.

Example MPLAB C18

```
_asm
clrf STKPTR, 0
goto 0x0004
_endasm
```

FAQ-15 I'm using the "chkEEPROM" boot loader entry method and I want to be sure that the last location in the PIC18F8722's EEPROM memory is set to 0xFF when programming the boot loader hex code with my MPLAB ICD 2.

Add the MPASM "de" directive to the boot loader source code to embed 0xFF directly into the last location of EEPROM memory. This will ensure the correct value 0xFF is present when the boot loader is executed the first time after being programmed into PIC18F8722.

```
; Force the last location in EEPROM = 0xFF causing the boot loader to execute
; by default on first power up after being programmed.

       ORG     0xF003FE              <------ Add this ORG statement
       DE      0xFF, 0xFF            <------ Add this DE directive and these two bytes
```

FAQ-16 There are several workspace and project files in the "HPC Explorer V1_01.zip" download file. Which one do I use?

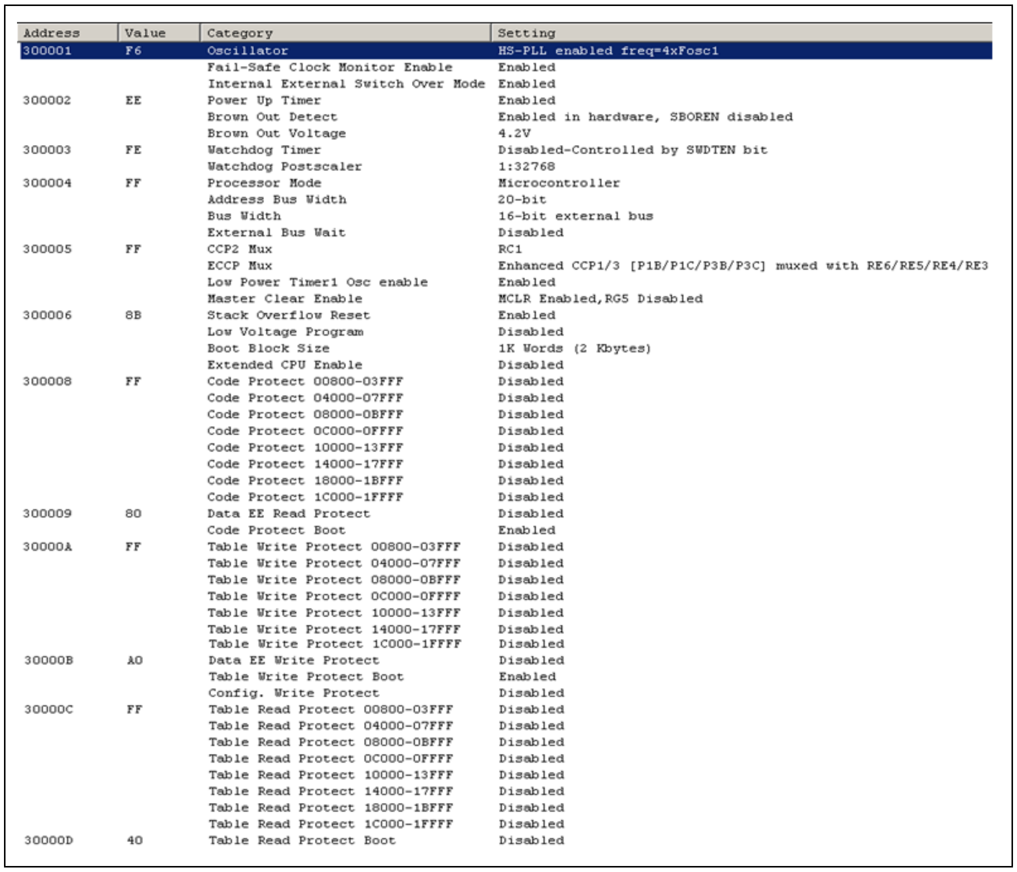
There are three different workspace files available in the zip file.

- 8722Demo.mcw – This workspace contains both the boot loader and demo source files. Use this to rebuild the complete HPC Explorer code image. MPLAB C18 is required. The resulting hex code must be programmed into the PIC18F8722 using an MPLAB ICD 2.
- Bootload.mcw – This workspace contains only the boot loader source code. Use this to re-assemble the boot loader using MPASM. The resulting hex code must be programmed into the PIC18F8722 using an MPLAB ICD 2.
- 8722Demo_no_boot.mcw – This workspace contains only the relocated demo C18 source code. Use this to re-compile the demo code for download using the boot loader. MPLAB C18 is required.

TIP: The "8722Demo_no_boot.mcw" workspace provides a good example of a "relocated" application for use with a boot loader. Copy and modify this existing source code to accelerate your own application code development.

FAQ-17 What are the default configuration bit settings for the HPC Explorer demo board?

Here is a screen shot of the configuration bit settings for the HPC Explorer board.



Address	Value	Category	Setting
300001	F6	Oscillator	HS-PLL enabled freq=4xFosc1
		Fail-Safe Clock Monitor Enable	Enabled
		Internal External Switch Over Mode	Enabled
300002	EE	Power Up Timer	Enabled
		Brown Out Detect	Enabled in hardware, SBOREN disabled
		Brown Out Voltage	4.2V
300003	FE	Watchdog Timer	Disabled-Controlled by SWDTEN bit
		Watchdog Postscaler	1:32768
300004	FF	Processor Mode	Microcontroller
		Address Bus Width	20-bit
		Bus Width	16-bit external bus
		External Bus Wait	Disabled
300005	FF	CCP2 Mux	RC1
		ECCP Mux	Enhanced CCP1/3 [P1B/P1C/P3B/P3C] muxed with RE6/RE5/RE4/RE3
		Low Power Timer1 Osc enable	Enabled
		Master Clear Enable	MCLR Enabled,RG5 Disabled
300006	8B	Stack Overflow Reset	Enabled
		Low Voltage Program	Disabled
		Boot Block Size	1K Words (2 Kbytes)
		Extended CPU Enable	Disabled
300008	FF	Code Protect 00800-03FFF	Disabled
		Code Protect 04000-07FFF	Disabled
		Code Protect 08000-0BFFF	Disabled
		Code Protect 0C000-0FFFF	Disabled
		Code Protect 10000-13FFF	Disabled
		Code Protect 14000-17FFF	Disabled
		Code Protect 18000-1BFFF	Disabled
		Code Protect 1C000-1FFFF	Disabled
300009	80	Data EE Read Protect	Disabled
		Code Protect Boot	Enabled
30000A	FF	Table Write Protect 00800-03FFF	Disabled
		Table Write Protect 04000-07FFF	Disabled
		Table Write Protect 08000-0BFFF	Disabled
		Table Write Protect 0C000-0FFFF	Disabled
		Table Write Protect 10000-13FFF	Disabled
		Table Write Protect 14000-17FFF	Disabled
		Table Write Protect 1C000-1FFFF	Disabled
30000B	A0	Data EE Write Protect	Disabled
		Table Write Protect Boot	Enabled
		Config. Write Protect	Disabled
30000C	FF	Table Read Protect 00800-03FFF	Disabled
		Table Read Protect 04000-07FFF	Disabled
		Table Read Protect 08000-0BFFF	Disabled
		Table Read Protect 0C000-0FFFF	Disabled
		Table Read Protect 10000-13FFF	Disabled
		Table Read Protect 14000-17FFF	Disabled
		Table Read Protect 18000-1BFFF	Disabled
		Table Read Protect 1C000-1FFFF	Disabled
30000D	40	Table Read Protect Boot	Disabled

FAQ-18 I have installed a PIC18F87J10 on the small break-off module that comes attached to the HPC Explorer demonstration board. What are the recommended values for the un-populated resistors and capacitors?

The resistor values will depend on the VDD operating voltage. A 0.1 µF bypass capacitor, SMT 0603, can be used in locations C101, C102, C103, C104 and C105. See scenario #1 below for the value recommended for C106.

The PIC18F87J10 device has a lower maximum 3.6V VDD specification as compared to the PIC18F8722. To accommodate the lower operating voltage requirement, there are two resistors on the module, R101 and R102 that can configure the LM317 adjustable voltage regulator for the desired VDD operating voltage. For a more detailed description on selecting the appropriate resistor values, see the “Notes on 3V application for the HPC Explorer Board” document located on the HPC Explorer web page.

In addition to VDD, the PIC18F87J10 requires a separate VDD_core voltage provided by either an external nominal 2.5v (1.8V min / 2.75V max) or the internal 2.5V VDD_core regulator.

Scenario #1

Suppose you want VDD = 3.6 VDC. Using the table below, we determine that R102 = 1.62K ohms. Since we are providing 3.6 VDC for VDD, we must enable the internal 2.5V VDD_core regulator by installing a shunt (0 ohm) resistor in R103 and leave R104 un-populated. When using the internal VDD_core regulator, you must provide an external filter capacitor with low ESR characteristics in the range of 1.0 µF to 10.0 µF in location C106. The PCB layout will accommodate a 1.0 µF, SMT 0603, capacitor.

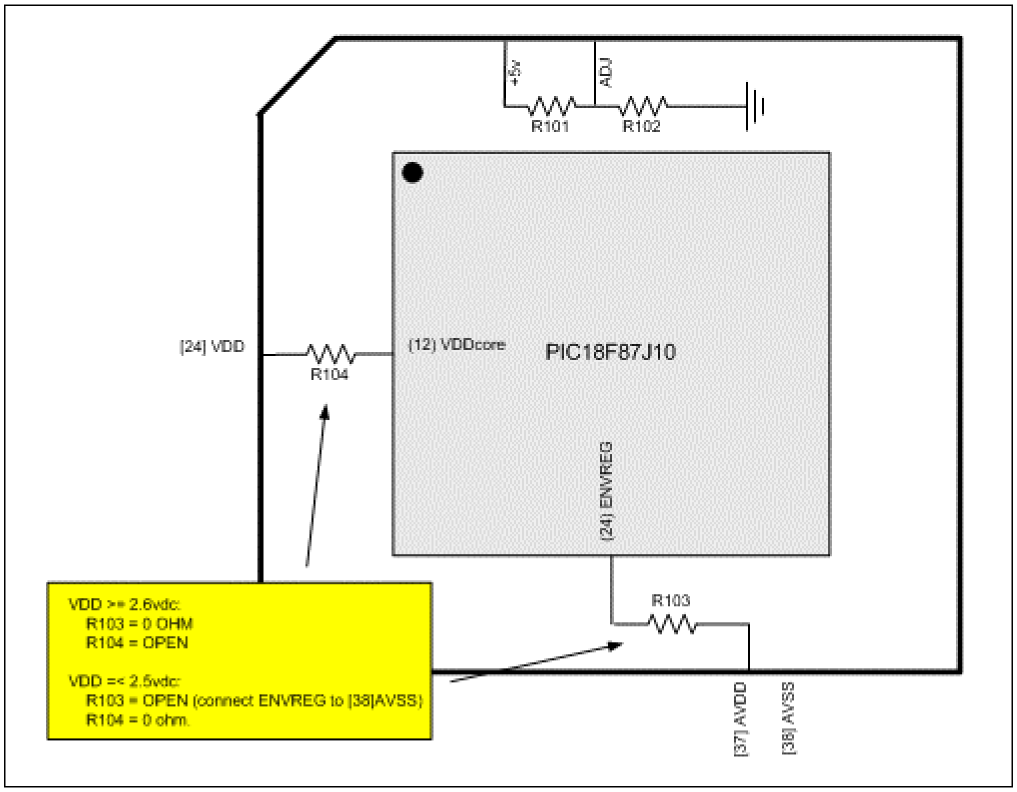
Scenario #2

Suppose you want VDD = 2.5 VDC and choose a value for R102 = 487 ohm. In this case, the PIC18F87J10 VDD is the same as the VDD_core and we can disable the internal 2.5 VDD_core regulator by not populating R103 and adding a small jumper wire from module pin #36 (RF0) to module pin #38 (AVSS). Connect VDD to VDD_core, by populating R104 with a shunt (0 ohm) resistor. Because of a limitation noted below, the break-off module was not designed for use below 3.0 VDC. However, the PIC18F87J10 will operate normally.

LIMITATION: The HPC Explorer’s RS232 transceiver may not operate below 3.0 VDC

VDD	R101	R102	R103	R104
3.6V	Open	1.62K ohm	0 ohm	Open
3.3V	Open	1.18K ohm	0 ohm	Open
3.0V	Open	866 ohm	0 ohm	Open
2.5V	Open	487 ohm	Open	0 ohm

Note 1: Values shown assume R25 = 1.0K, R26 = 300 ohm. Resistor sizes are SMT 0603.

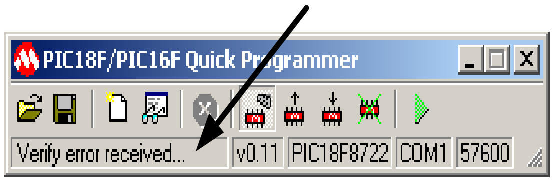


FAQ-19 What is the manufacturer and part number of the connectors on the bottom of the small break-off module?

These are 1 x 21 1.27mm high profile connectors. You will need (4) 1 x 21 and (1) 1 x 3. The manufacturer is OUPIIN and the manufacturer’s part number is 2246-1x21GS.

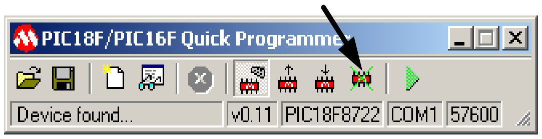
FAQ-20 Why do I receive a verify error when attempting to download my application to the HPC Explorer boot loader?

The PIC18F/PIC16F Quick Programmer application has detected an error while verifying the contents of program memory during the programming operation. In most cases this error is caused by not erasing the previous program memory contents before downloading a new application.



Verify error received... | v0.11 | PIC18F8722 | COM1 | 57600

Before downloading the application code, make sure you erase the previous program memory contents by clicking on the "Erase" icon in the toolbar as shown.



Device found... | v0.11 | PIC18F8722 | COM1 | 57600

FAQ-21 When using the MPLAB ICD 2 to debug the 8722Demo project, I receive warning messages about code protection bits that must be disabled. What am I doing wrong?

The 8722Demo project contains a file named "config.asm" that defines the state of the configuration bits for the PIC18F8722 on the HPC Explorer Demo Board. Two of these configuration bits are boot block write protect, CONFIG6H<WRTB> and boot block code protect, CONFIG5H<CPB>. Both are enabled to protect the boot loader during normal operation. When debugging with the MPLAB ICD 2, it is normal to receive a warning message regarding these two bits because code protection is not allowed during debugging and MPLAB IDE will temporarily override and disable these bits.

To prevent the warning messages from occurring, you can modify the config.asm file to temporarily disable these configuration bits as shown in the example below:

```
;Either comment out these two lines
;CONFIG CPB = ON
;CONFIG WRTB = ON

;Or change the setting = OFF
CONFIG CPB = OFF
CONFIG WRTB = OFF
```

NOTES:

MICROCHIP

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

San Jose
Mountain View, CA
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

India - New Delhi
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Gumi
Tel: 82-54-473-4301
Fax: 82-54-473-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Penang
Tel: 60-4-646-8870
Fax: 60-4-646-5086

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

10/31/05