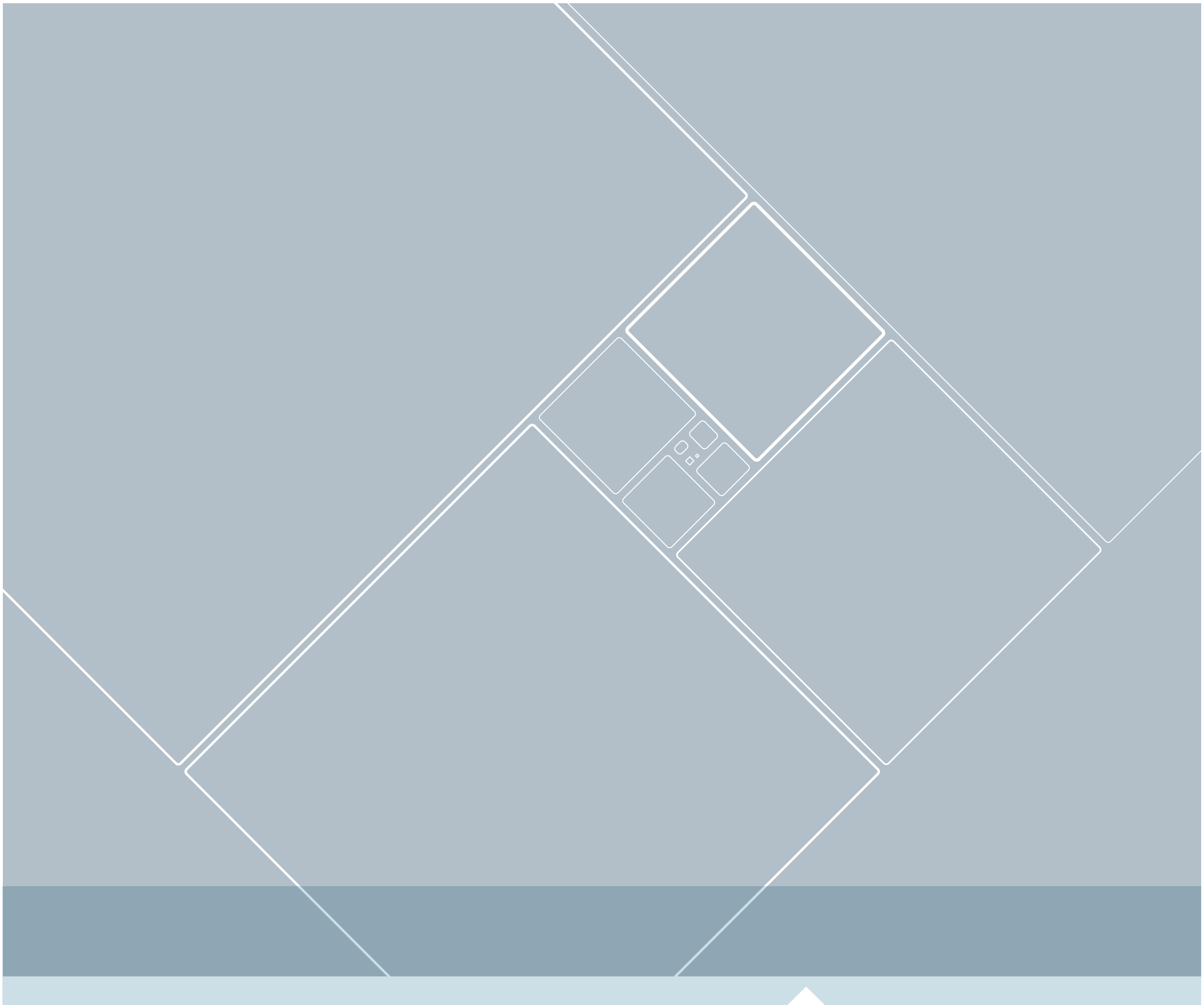


Freescale Solutions

Freescale MQX Software Solutions

Enabling Embedded Systems—Accelerating Success

freescale™
semiconductor

Table of Contents

The following lab tutorials are applicable to both the M52259EVB and M52259DEMOKIT

LAB 1

HVAC Controller, Freescale MQX RTOS for MCF52259

Covers both M52259EVB and M52259DEMOKIT

This lab will guide you through the general use of Freescale MQX operating system under the CodeWarrior™ environment to familiarize you with the general compile and download process as well as provide an understanding of the application.

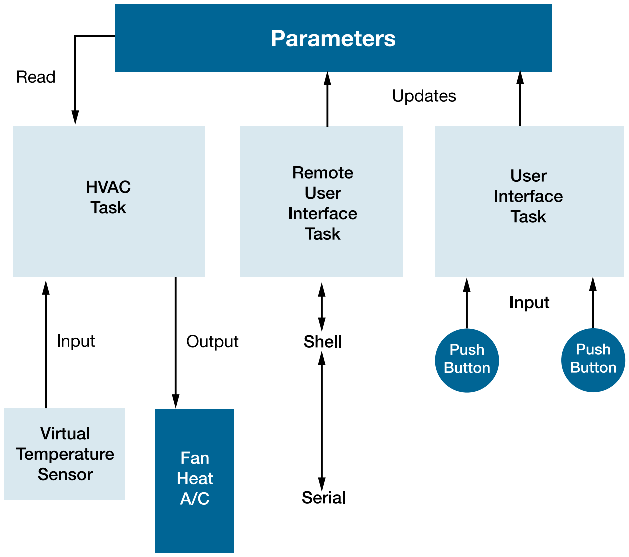
Demonstrates

- MQX project in CodeWarrior IDE
- Project build, download and run in CodeWarrior IDE
- MQX Shell and serial line console
- MQX GPIO driver (push button and LEDs)

Step by Step Instructions

1. Make the following connections from the MCF52259 board to the computer. See Figure 1 to the right, or Figure 2 below.
   a. USB debugger connection to a USB port on PC
   b. Serial Port 0 (UART 0) to a Serial Port on PC
   c. Attach power cable and turn on the board's power switch (M52259EVB only)
2. The first time you connect the USB debugger cable to your PC, Windows will install a driver for the debugger. Follow the prompts to automatically detect and install the driver.
3. Install the 7.1.1 (or later) update patch for CodeWarrior for ColdFire® V7.1 if it has not already been installed. Then, open CodeWarrior for ColdFire V7.1.
4. If you did not install the MQX project in the default **C:\Program Files\Freescale MQX 3.0\directory** during installation, you must first recompile the MQX libraries to reflect the new path name. See the release notes for more information before continuing on with the lab.

MQX Application Software—HVAC Controller



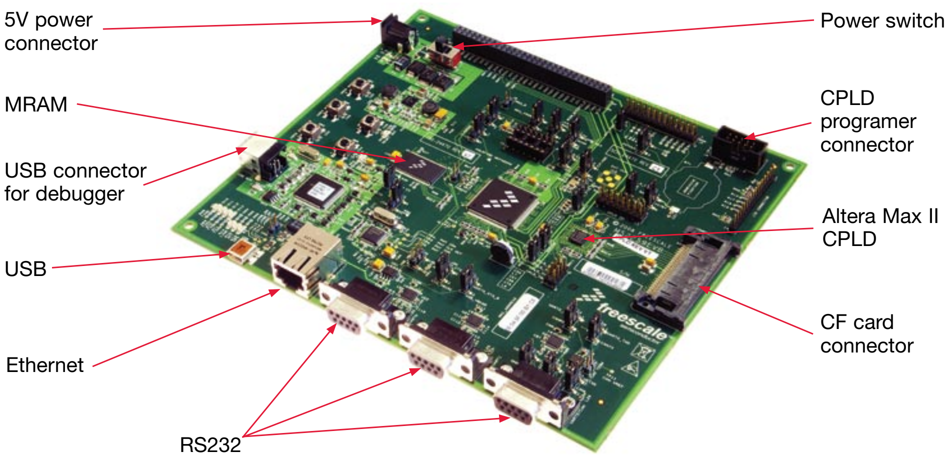


Figure 1: M52259EVB board layout

5. Open the Lab Project by selecting the **File > Open** menu item:
   - For M52259EVB board, open file: **c:\Program Files\Freescale MQX 3.0\demo\hvac\codewarrior\hvac_m52259evb.mcp**
   - For M52259DEMO board, open file: **c:\Program Files\Freescale MQX 3.0\demo\hvac\codewarrior\hvac_m52259demo.mcp**

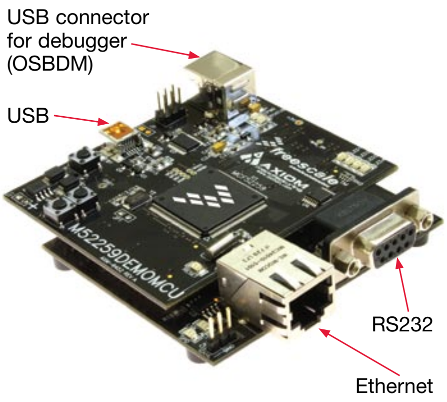
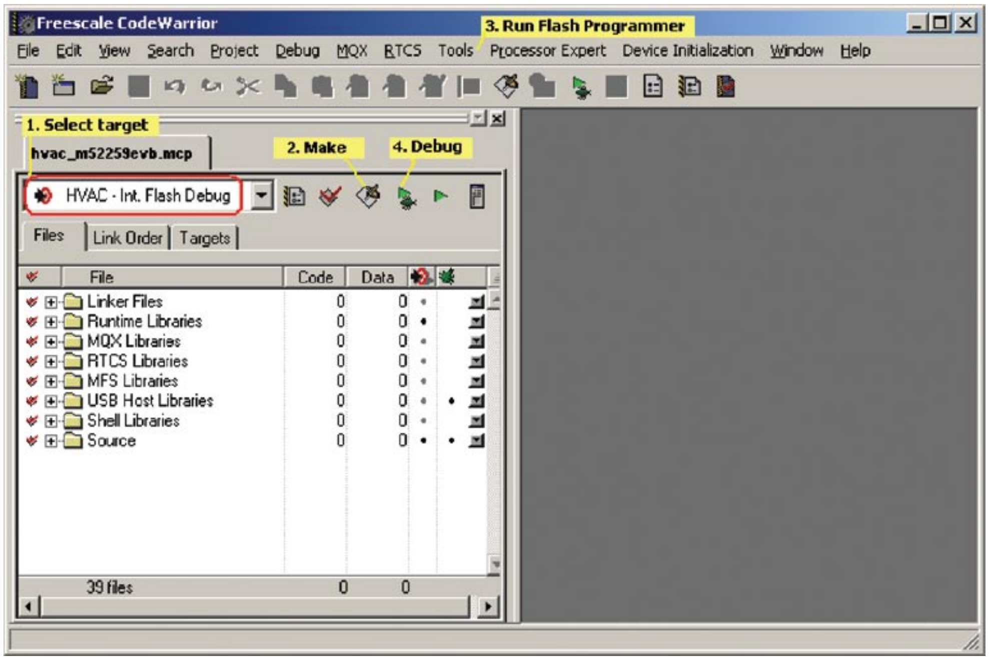


Figure 2: M52259DEMO board layout

6. In the project pane, select “HVAC—Int. Flash Debug” build target. See Figure 3 to the right.
7. Compile the project by pressing the F7 key or by clicking the **Make** icon on the project pane toolbar.
8. Open the “Flash Programmer” tool in the *Tools* menu. Make sure the “Use custom Settings” checkbox is not selected (i.e. the configuration will be taken from the project). See Figure 4 below.
9. Click the “Load Settings” button and select *MCF52259_INTFLASH.xml.*
10. Select the *Erase/Blank Check* page, select “All Sectors” and click **Erase**. See Figure 5 below.
11. Select the *Program/Verify* page and click **Program**. Upon successful programming, the status will display “Completed Successfully.”
12. Now start debugger and execute the application by clicking the **Debug** icon. This icon is identified in **Figure 3** on right.


Freescale CodeWarrior
3. Run Flash Programmer
File Edit View Search Project Debug MQX RTCS Tools Processor Expert Device Initialization Window Help
1. Select target
2. Make
4. Debug
Files
Link Order
Targets

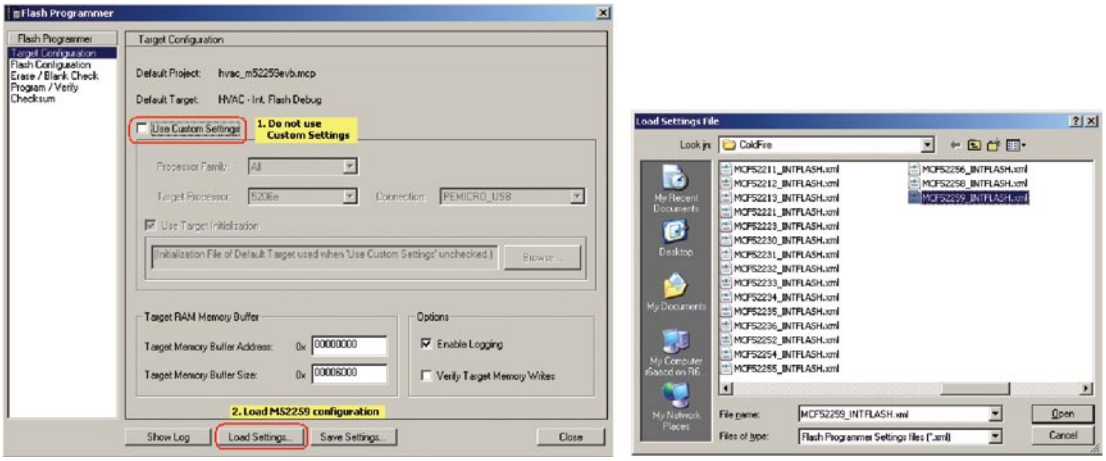
hvac_m52259evb.mcp

HVAC - Int. Flash Debug

File	Code	Data
Linker Files	0	0
Runtime Libraries	0	0
MQX Libraries	0	0
RTCS Libraries	0	0
MFS Libraries	0	0
USB Host Libraries	0	0
Shell Libraries	0	0
Source	0	0
39 files	0	0

Figure 3: Project Loaded in the CodeWarrior IDE

NOTE: With M52259EVB board, you can use MRAM memory to store the executable code instead of the flash. Select “HVAC—Ext. MRAM Debug” and skip the Flash Programmer steps 8–11. The code will be downloaded to MRAM automatically by the debugger.



Target Configuration

Default Project: hvac_m52259evb.mcp

Default Target: HVAC - Int. Flash Debug

□ Use Custom Settings

Processor Family: All

Target Processor: 5206e Connection: PEMICRO_USB

☑ Use Target Initialization

(Initialization File of Default Target used when 'Use Custom Settings' unchecked.) Browse...

Target RAM Memory Buffer

Target Memory Buffer Address: 0x 00000000

Target Memory Buffer Size: 0x 00006000

Options

☑ Enable Logging

□ Verify Target Memory Writes

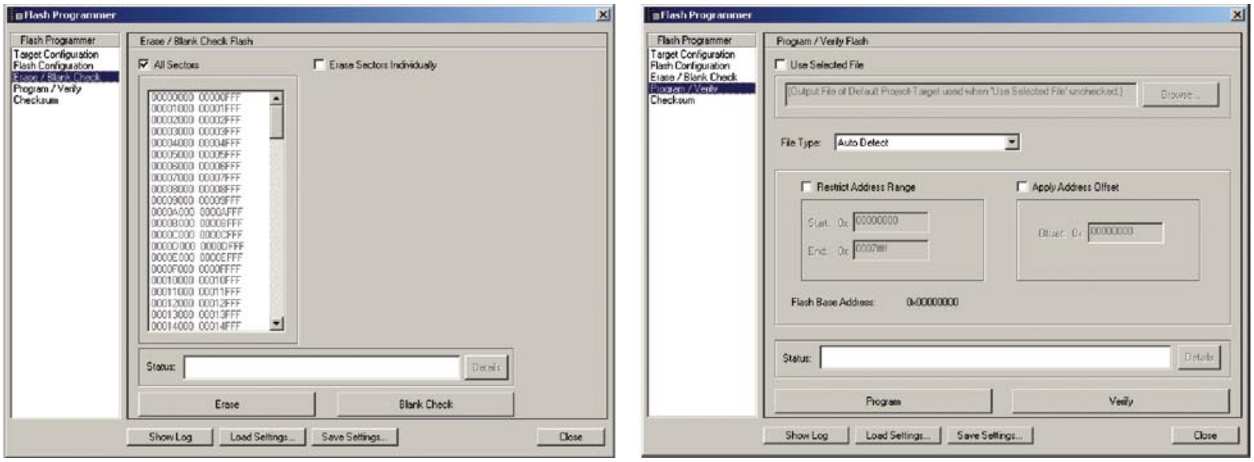
Look in: ColdFire

MCF52211_INTFLASH.xml
MCF52212_INTFLASH.xml
MCF52213_INTFLASH.xml
MCF52221_INTFLASH.xml
MCF52223_INTFLASH.xml
MCF52230_INTFLASH.xml
MCF52231_INTFLASH.xml
MCF52232_INTFLASH.xml
MCF52233_INTFLASH.xml
MCF52234_INTFLASH.xml
MCF52235_INTFLASH.xml
MCF52236_INTFLASH.xml
MCF52252_INTFLASH.xml
MCF52254_INTFLASH.xml
MCF52255_INTFLASH.xml
MCF52256_INTFLASH.xml
MCF52258_INTFLASH.xml
MCF52259_INTFLASH.xml

File name: MCF52259_INTFLASH.xml

Files of type: Flash Programmer Settings files (*.xml)

Figure 4: Opening Flash Programmer Tool



Erase / Blank Check Flash

☑ All Sectors □ Erase Sectors Individually

```
00000000  00000FFF
00001000  00001FFF
00002000  00002FFF
00003000  00003FFF
00004000  00004FFF
00005000  00005FFF
00006000  00006FFF
00007000  00007FFF
00008000  00008FFF
00009000  00009FFF
0000A000  0000AFFF
0000B000  0000BFFF
0000C000  0000CFFF
0000D000  0000DFFF
0000E000  0000EFFF
0000F000  0000FFFF
00010000  00010FFF
00011000  00011FFF
00012000  00012FFF
00013000  00013FFF
00014000  00014FFF
```

Status:

Program / Verify Flash

□ Use Selected File

(Output File of Default Project-Target used when 'Use Selected File' unchecked.)

File Type: Auto Detect

□ Restrict Address Range

Start: 0x 00000000

End: 0x 0007ffff

□ Apply Address Offset

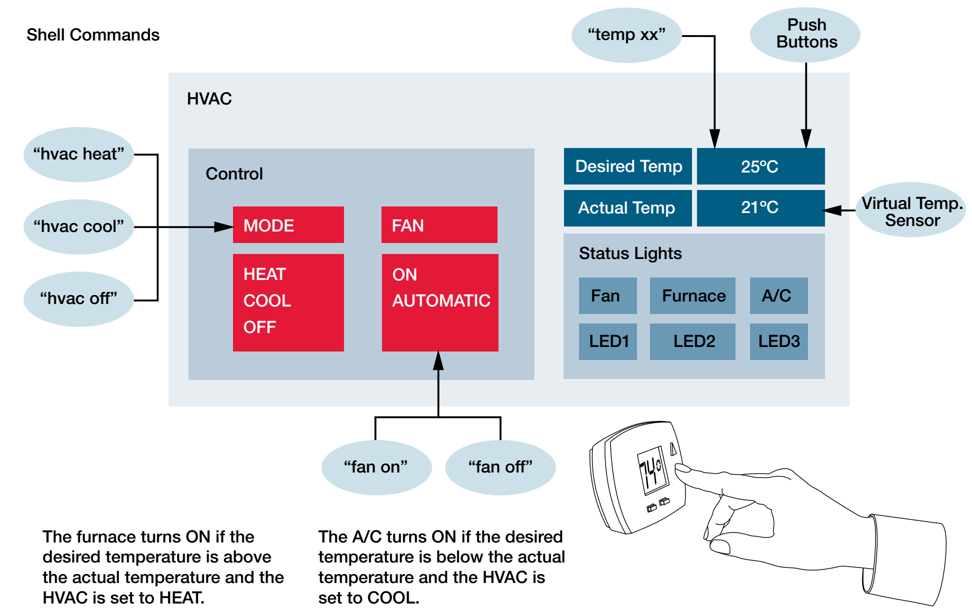
Offset: 0x 00000000

Flash Base Address: 0x00000000

Status:

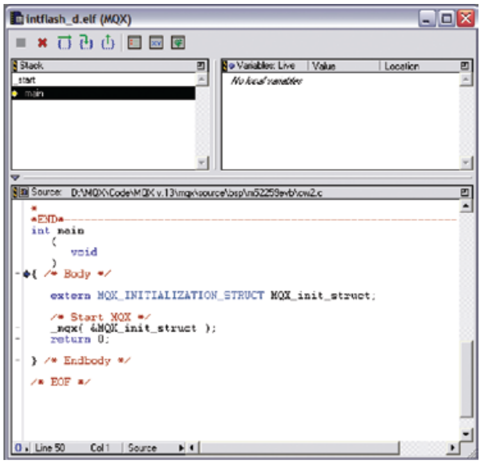
Figure 5: Downloading Code with Flash Programmer Tool

MQX HVAC Control Panel Thermostat



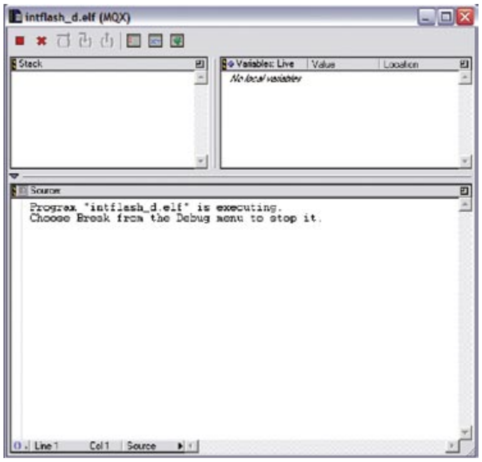
The furnace turns ON if the desired temperature is above the actual temperature and the HVAC is set to HEAT.

The A/C turns ON if the desired temperature is below the actual temperature and the HVAC is set to COOL.



```
 *
 *END*------------------------------------------------------------
 int main
    (
       void
    )
{ /* Body */

   extern MQX_INITIALIZATION_STRUCT MQX_init_struct;

   /* Start MQX */
   _mqx( &MQX_init_struct );
   return 0;

} /* Endbody */

/* EOF */
```

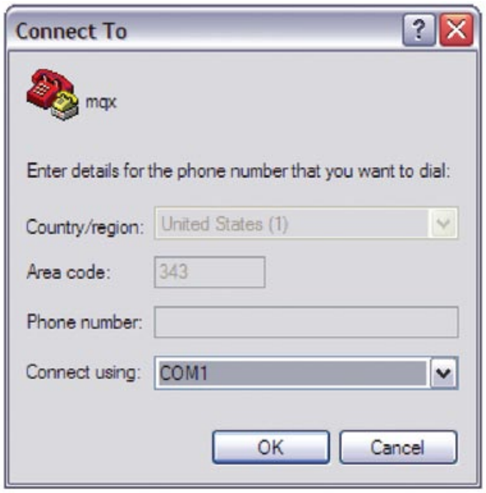
Figure 6: MQX Debug entry point



```
Program "intflash_d.elf" is executing.
Choose Break from the Debug menu to stop it.
```

Figure 7: Running MQX code

13. Once the debug session starts you will see the MQX entry-point function in the code window as seen in Figure 6. Click the **Debug** icon again to run the code, which will look like Figure 7.
14. Start HyperTerminal on the PC *(Start menu->Programs->Accessories->Communications)*. Make a connection to the serial port that is connected to the board (usually will be COM1)



mqx

Enter details for the phone number that you want to dial:

Country/region: United States (1)

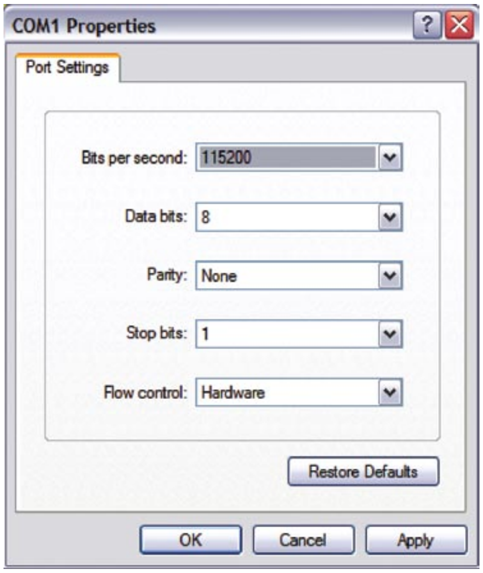
Area code: 343

Phone number:

Connect using: COM1

Figure 8: HyperTerminal Console Window

15. Set it for 115200 baud, no parity, 8 bits and click OK.



Bits per second: 115200

Data bits: 8

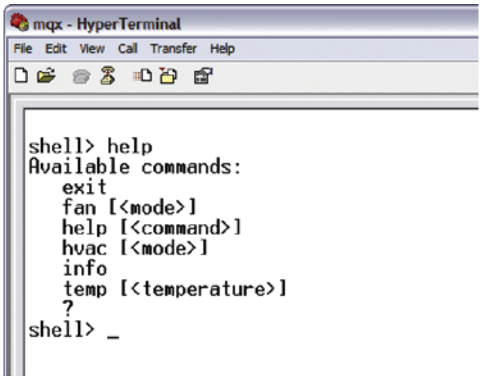
Parity: None

Stop bits: 1

Flow control: Hardware

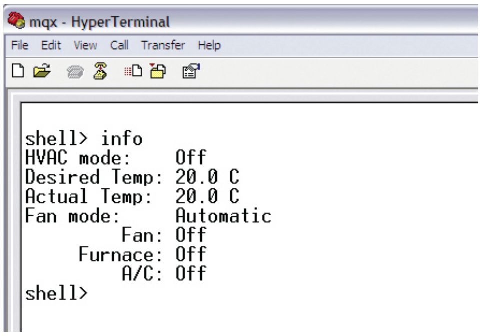
Figure 9: HyperTerminal Console Window

16. Press **Enter** in the terminal window on the PC. You should see the **shell>** prompt. Status updates will appear every 15 seconds in the terminal window.



```
shell> help
Available commands:
   exit
   fan [<mode>]
   help [<command>]
   hvac [<mode>]
   info
   temp [<temperature>]
   ?
shell> _
```

Figure 10: Help command in HyperTerminal Console Window



```
shell> info
HVAC mode:     Off
Desired Temp:  20.0 C
Actual Temp:   20.0 C
Fan mode:      Automatic
         Fan:  Off
     Furnace:  Off
         A/C:  Off
shell>
```

Figure 11: Info command in Hyperterminal Console Window

17. Type **help** to see the list of available commands. Use the **info** command to display the current settings (See Figures 10 and 11).
18. A simple HVAC controller has been implemented as the example application to demonstrate the features of the software. The HVAC system operates per the diagram on page 5.
19. The LEDs represent the current state of the HVAC:

    a. When using the EVB,
    D10 on EVB represents **Fan ON/OFF**
    D11 on EVB represents **Furnace ON/OFF**
    D12 on EVB represents **A/C ON/OFF**

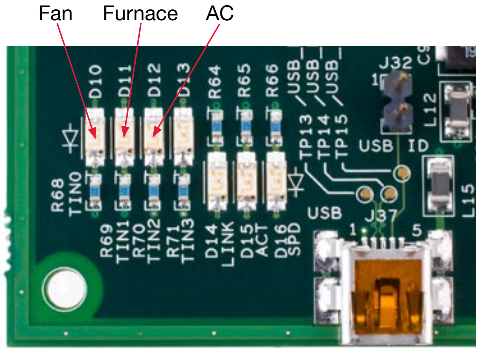


Figure 12: EVB LEDs

b. When using the DEMOKIT,
LED 1 represents **Fan ON/OFF**
LED 2 represents **Furnace ON/OFF**
LED 3 represents **A/C ON/OFF**

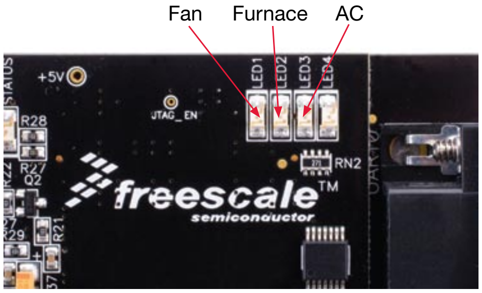


Figure 13: DEMOKIT LEDs

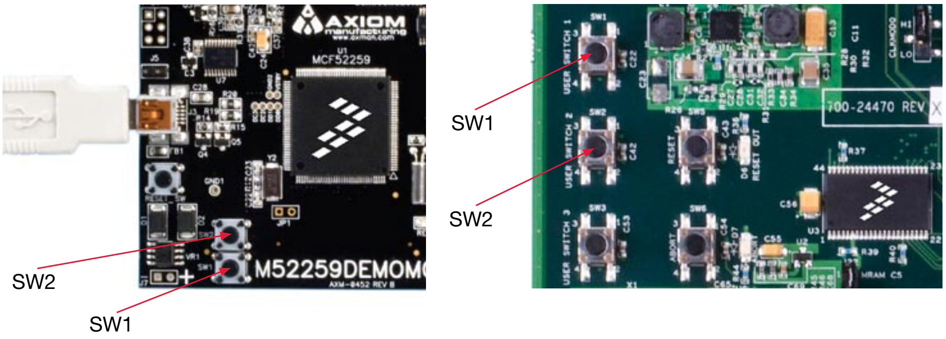


Figure 14: Push buttons on the DEMO and EVB boards

20. Use the **fan** command (**fan on** and **fan off**) to control the fan (LED 1). Use **info** at any time to confirm the status. The **fan on** command forces the fan to be on, while **fan off** puts the fan into automatic mode so that the fan is only on when the HVAC needs to change temperature.
21. Since the default settings for the demo have the “desired temp” and “actual temp” at 20ºC, both the furnace or the AC do not need to be turned on.
22. Turn the fan off to put it in automatic mode, and put the HVAC into heat mode using the **hvac** command (**hvac heat**). This turns on the furnace, and with the fan in automatic mode, the fan will only turn on when the desired temperature is higher than the actual temperature.
23. Push buttons SW1 and SW2 on the board are used to change the desired temperature. SW1 increases the desired temp and SW2 lowers it. Press SW1 once and then type **info** in the shell.
24. Press SW1 again, type **info** and see that the furnace has turned on and the fan has automatically turned on. The furnace and fan LEDs should be on to indicate this status.
25. Use SW2 to lower the desired temp such that the furnace and the fan turn off. Use the **temp** command to change the desired temperature to 24ºC (**temp 24**). The fan and the furnace should turn back on.
26. Experiment on your own with changing temperature and fan settings.
27. Terminate the debug session by pressing Shift-F5 or by selecting **Debug>Kill** in CodeWarrior.

LAB 2

USB functionality, Freescale MQX USB and MFS

Covers both M52259EVB and M52259DEMOKIT

This lab will guide you through the use of the USB Stack, the MFS File System, and the logging of data to the memory stick.

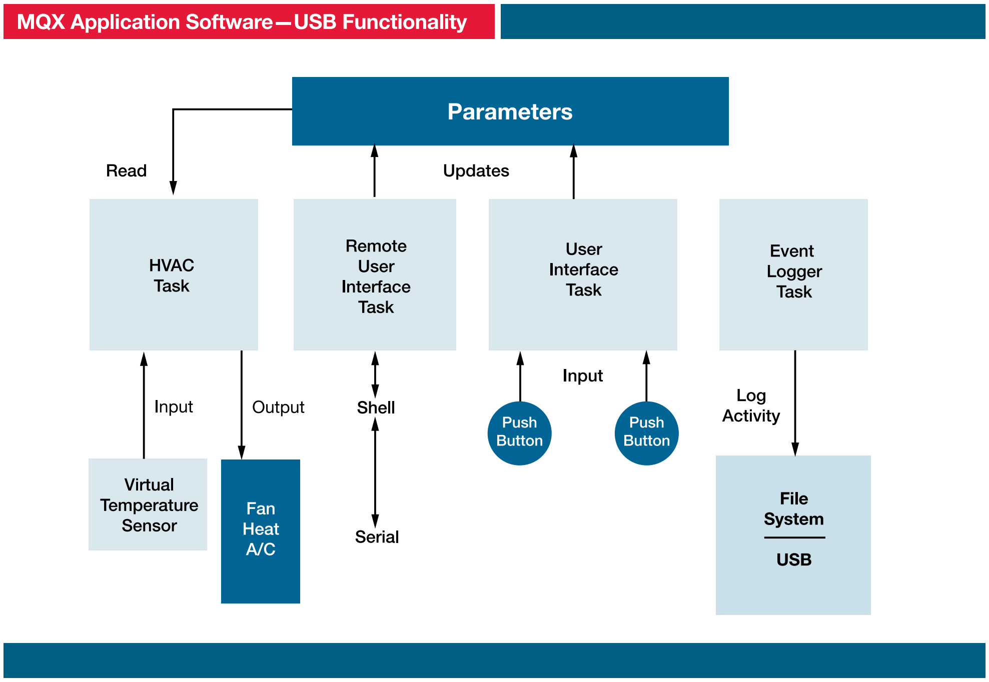
Demonstrates

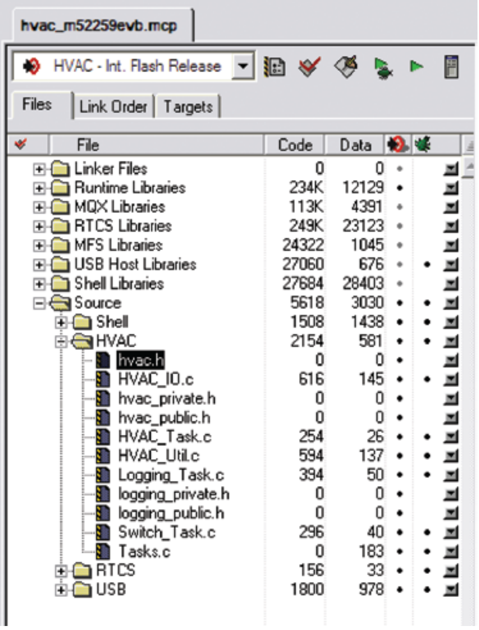
- USB host functionality with mass storage class support
- USB drive read and write access by using the MFS file system component

Step by Step Instructions

1. Stop the application in CodeWarrior if it is currently running (**Debug>Kill**)
2. Open **hvac.h** in CodeWarrior

MQX Application Software—USB Functionality





File	Code	Data		
Linker Files	0	0	•	
Runtime Libraries	234K	12129	•	
MQX Libraries	113K	4391	•	
RTCS Libraries	249K	23123	•	
MFS Libraries	24322	1045	•	
USB Host Libraries	27060	676	•	•
Shell Libraries	27684	28403	•	
Source	5618	3030	•	•
Shell	1508	1438	•	•
HVAC	2154	581	•	•
hvac.h	0	0	•	
HVAC_IO.c	616	145	•	•
hvac_private.h	0	0	•	
hvac_public.h	0	0	•	
HVAC_Task.c	254	26	•	•
HVAC_Util.c	594	137	•	•
Logging_Task.c	394	50	•	•
logging_private.h	0	0	•	
logging_public.h	0	0	•	
Switch_Task.c	296	40	•	•
Tasks.c	0	183	•	•
RTCS	156	33	•	•
USB	1800	978	•	•

Figure 1: MQX Source Tree

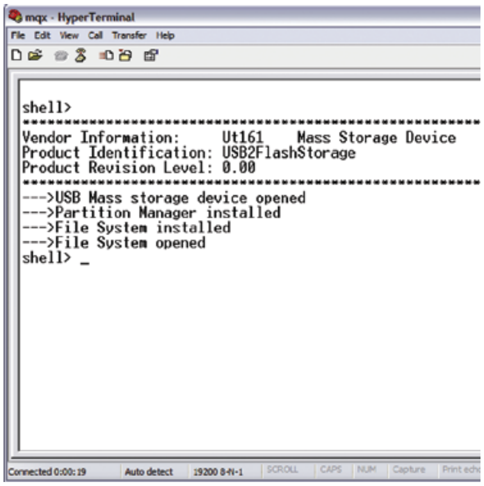
3. Enable the USB support by changing:

   #define DEMOCFG_ENABLE_USB_FILESYSTEM 0

   to:

   #define DEMOCFG_ENABLE_USB_FILESYSTEM 1

4. Recompile the code, download, and run the application as was done in steps 6 to 13 of Lab 1.
5. Insert the USB memory stick into the USB connector on the board. You will need the mini USB-B adaptor plug. See Figure 2 on the next page.
6. On the HyperTerminal console you should see text that indicates that the USB memory stick was found, the type of memory stick that was found, the USB mass storage device was opened, the partition manager was installed, and that the file system was installed and opened. It may take several seconds to fully install the memory stick. It will look like Figure 3.



```
shell>
**********************************************************
Vendor Information:     Ut161    Mass Storage Device
Product Identification: USB2FlashStorage
Product Revision Level: 0.00
**********************************************************
--->USB Mass storage device opened
--->Partition Manager installed
--->File System installed
--->File System opened
shell> _
```

Figure 3: Inserting a USB mass storage device

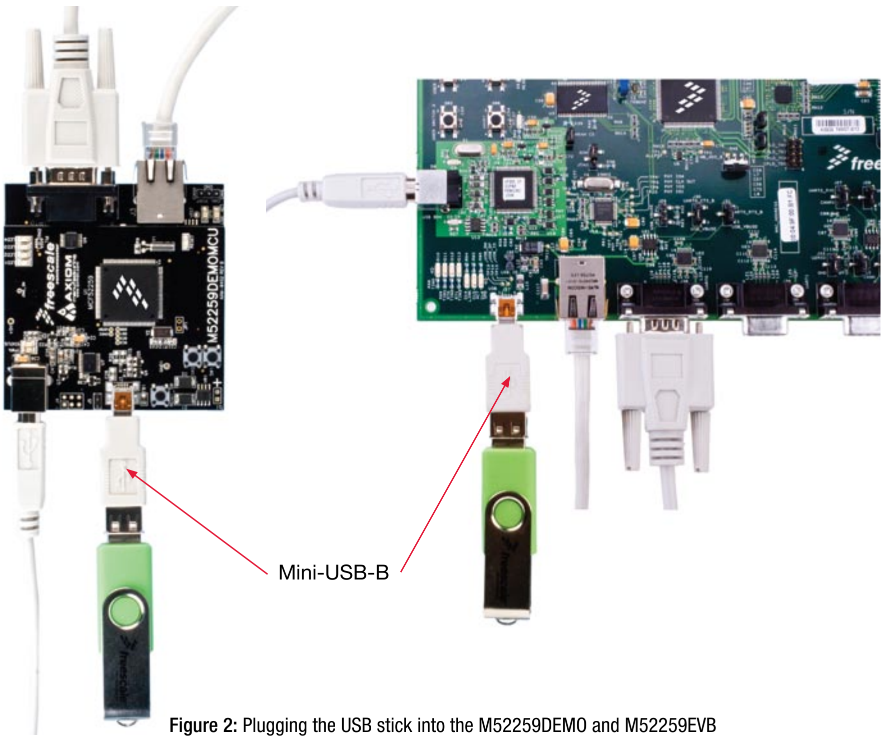
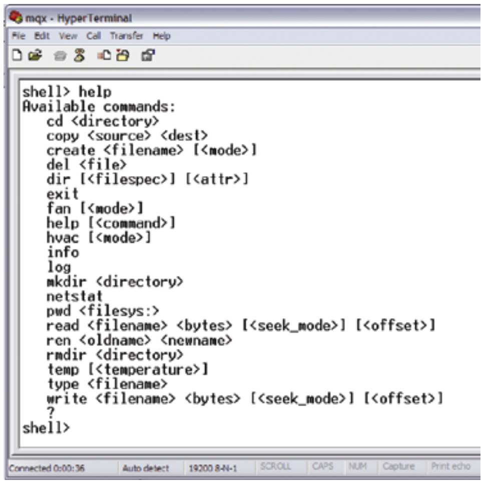
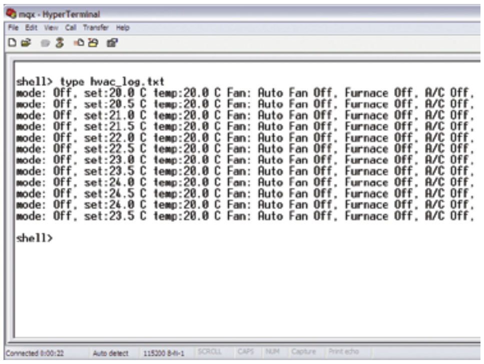


Figure 2: Plugging the USB stick into the M52259DEMO and M52259EVB


mqx - HyperTerminal
File Edit View Call Transfer Help
Connected 0:00:36 Auto detect 19200 8-N-1 SCROLL CAPS NUM Capture Print echo

```
shell> help
Available commands:
   cd <directory>
   copy <source> <dest>
   create <filename> [<mode>]
   del <file>
   dir [<filespec>] [<attr>]
   exit
   fan [<mode>]
   help [<command>]
   hvac [<mode>]
   info
   log
   mkdir <directory>
   netstat
   pwd <filesys:>
   read <filename> <bytes> [<seek_mode>] [<offset>]
   ren <oldname> <newname>
   rmdir <directory>
   temp [<temperature>]
   type <filename>
   write <filename> <bytes> [<seek_mode>] [<offset>]
   ?
shell>
```

Figure 4: Shell commands with USB functionality enabled



```
shell> type hvac_log.txt
mode: Off, set:20.0 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:20.5 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:21.0 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:21.5 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:22.0 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:22.5 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:23.0 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:23.5 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:24.0 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:24.5 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:24.0 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:23.5 C temp:20.0 C Fan: Auto Fan Off, Furnace Off, A/C Off,

shell>
```

Figure 5: HVAC log file contents

7. Press return on the PC to get the **shell**> prompt.
8. Type **help** to see the list of commands. You should see additional commands available as seen in Figure 4. You can use the commands to exercise the memory stick attached.
9. Type **dir** to see the memory stick directory listing.
10. The logging information is now being sent to the memory stick in file **hvac_log.txt** instead of being printed to the serial port. You should see this file in the directory listing. Note the file size of **hvac_log.txt**.
11. Press SW1 and SW2 a few times each to change the desired temperature.
12. Type **dir** again in the shell and note the file size of **hvac_log.txt**. It should be bigger than before indicating that the changed parameters are being logged to the memory stick.
13. Use the **type** command to display the contents of the log file located on the memory stick (**type hvac_log.txt**). It will look like Figure 5.
14. Delete the log file using the **del** command (**del hvac_log.txt**). Once again view the directory listing to verify that the file has been removed.
15. Within 15 seconds (or after SW1/SW2 key press) you should see that the log file has been recreated. Use the **type** command on the log file (**type hvac_log.txt**) to see the latest entry.

LAB 3

Telnet and FTP Operation, Freescale MQX RTCS

Covers both M52259EVB and M52259DEMOKIT

This lab will introduce you to the use of Telnet and FTP to access a product remotely via Ethernet and TCP/IP protocols.

Demonstrates

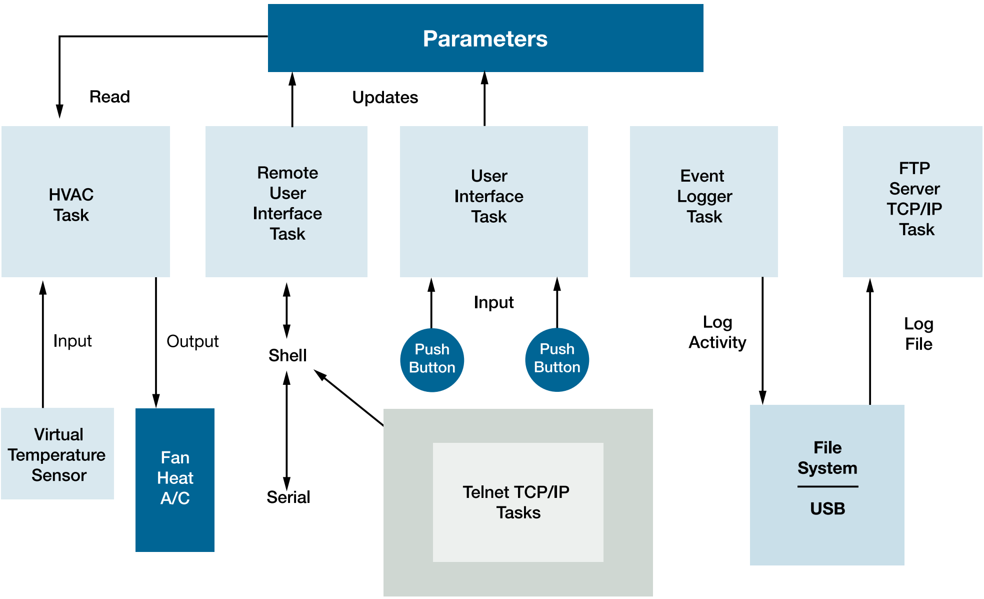
- MQX RTCS TCP/IP network stack
- Telnet server functionality
- FTP server functionality

Step by Step Instructions

1. Stop the application if it is currently running (**Debug>Kill**).
2. Connect a crossover Ethernet cable between the board and an Ethernet port on your computer
3. The default IP address of the board is 169.254.3.3. Typically, when you connect your computer directly to the board, the computer will default to an auto IP address on the same subnet as the board (169.254.x.x), therefore requiring no setup. Note: The PC may take a few minutes to default to the auto IP address and make the connection.

   However, if you have trouble connecting, you may configure the IP address of the computer manually. Select Start > Settings > Network Connections > Local Area Connection. Note your original TCP/IP settings, and then set your IP address to 169.254.3.4 and your subnet mask to 255.255.0.0.

MQX Application Software—Telnet/FTP Operation



4. Open the **HVAC.h** file in the CodeWarrior window. Double-click the file item located in the "Source/HVAC" group in the CodeWarrior project tree.
5. Locate the line of code starting with **#define ENET_IPADDR** and specify your target IP address by using the IPADDR macro. Use the IP address 169.254.3.3 if it has not already been set:

   ```
   #define ENET_IPADDR IPADDR(169,254,3,3)
   ```
6. Do the same with the IP address mask value ENET_IPMASK:

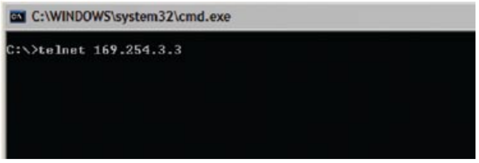
   ```
   #define ENET_IPMASK IPADDR(255,255,0,0)
   ```
7. In the same **HVAC.h** file, enable RTCS and Telnet servers by changing this:

   ```
   #define DEMOCFG_ENABLE_RTCS  0
   #define DEMOCFG_ENABLE_FTP_SERVER 0
   #define DEMOCFG_ENABLE_TELNET_SERVER  0
   ```

   to this:

   ```
   #define DEMOCFG_ENABLE_RTCS  1
   #define DEMOCFG_ENABLE_FTP_SERVER   0
   #define DEMOCFG_ENABLE_TELNET_SERVER  1
   ```
8. Recompile the code, download, and run the application as was done in steps 6 to 13 of Lab 1. Then plug in the USB stick as done in Lab 2.

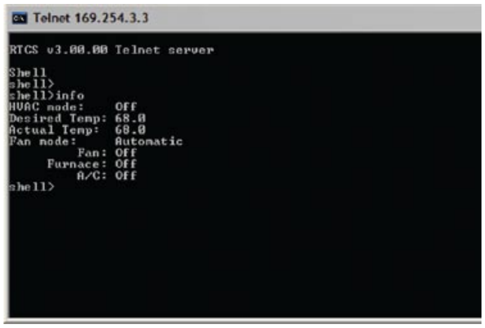
9. Open a Command Prompt on the PC *(Start > Programs > Accessories > Command Prompt)*



```
C:\>telnet 169.254.3.3
```

Figure 1: Connect to telnet

10. At the prompt invoke a telnet session to the board by typing **telnet 169.254.3.3** you will be connected to the MQX shell via telnet.



```
RTCS v3.00.00 Telnet server

Shell
shell>
shell>info
HVAC mode:     OFF
Desired Temp: 68.0
Actual Temp:  68.0
Fan mode:     Automatic
         Fan: Off
     Furnace: Off
         A/C: Off
shell>
```

Figure 2: Using shell commands over telnet

11. The telnet shell has its own command set, which is similar to the terminal shell commands as in the previous labs. Try the **help** and **info** commands.

12. Try the **log** command to print content of the **hvac_log.txt** file to the telnet console.

13. In the same **HVAC.h** file, disable Telnet and enable the FTP server by changing this:

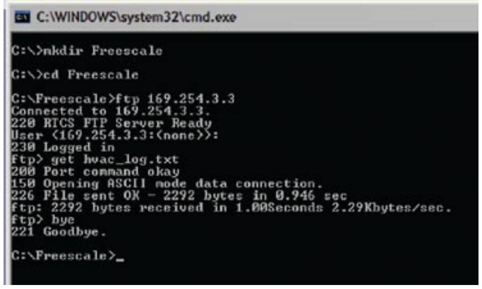
```
#define DEMOCFG_ENABLE_RTCS 1
#define DEMOCFG_ENABLE_FTP_SERVER 0
#define DEMOCFG_ENABLE_TELNET_SERVER 1
```

to this:

```
#define DEMOCFG_ENABLE_RTCS 1
#define DEMOCFG_ENABLE_FTP_SERVER 1
#define DEMOCFG_ENABLE_TELNET_SERVER 0
```

14. Recompile the code, download, and run the application as was done in steps 6 to 13 of Lab 1.

15. Open a second command prompt *(Start > All Programs > Accessories > Command Prompt)* and change the working directory for the prompt to the c:\Freescale folder (type **“c:”, “cd \”, “mkdir Freescale”** and **“cd Freescale”** commands). See Figure 3 below for an example.



```
C:\>mkdir Freescale

C:\>cd Freescale

C:\Freescale>ftp 169.254.3.3
Connected to 169.254.3.3.
220 RTCS FTP Server Ready
User (169.254.3.3:(none)):
230 Logged in
ftp> get hvac_log.txt
200 Port command okay
150 Opening ASCII mode data connection.
226 File sent OK - 2292 bytes in 0.946 sec
ftp: 2292 bytes received in 1.00Seconds 2.29Kbytes/sec.
ftp> bye
221 Goodbye.

C:\Freescale>_
```

Figure 3: Communicating with FTP

16. Start the **ftp** session by typing the following in the Windows command prompt:

```
>ftp 169.254.3.3
```

For the user name, press Enter to leave it blank.

```
User (169.254.3.3(none)):
```

17. Get the USB memory stick directory listing in the ftp session by the **ls** command.

18. Use **get hvac_log.txt** command in the ftp session to retrieve the log file. The file will be copied to the local working directory **c:\Freescale**. Open the local file with notepad to verify that the file was copied properly.

LAB 4

Web-Enabled HVAC System, Freescale MQX RTCS

Covers both M52259EVB and M52259DEMOKIT

This lab extends the HVAC application described in Labs 1–3 and adds Web server functionality. The dynamic Web pages serve as a graphical user's interface to the HVAC application.

Demonstrates

- MQX RTCS TCP/IP network stack
- HTTP server functionality

Step by Step Instructions

1. Make the following connections from the MCF52259 board to the computer. See **Figure 1** on right.
   a. USB debugger connection to a USB port on PC
   b. Serial Port 0 (UART 0) to a Serial Port on PC
   c. A crossover Ethernet cable between the board and an Ethernet port on your computer
   d. Attach power cable and turn on the board's power switch (M52259EVB only)
2. The first time you connect the USB debugger cable to your PC, Windows will install a driver for the debugger. Follow the prompts to automatically detect and install the driver.

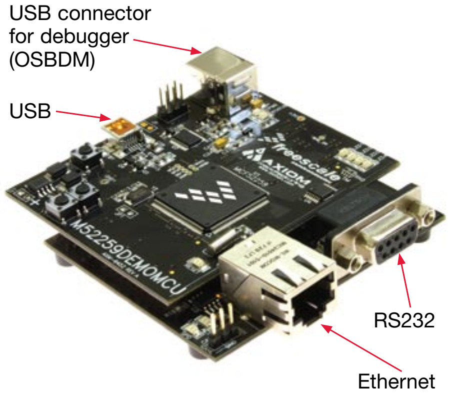
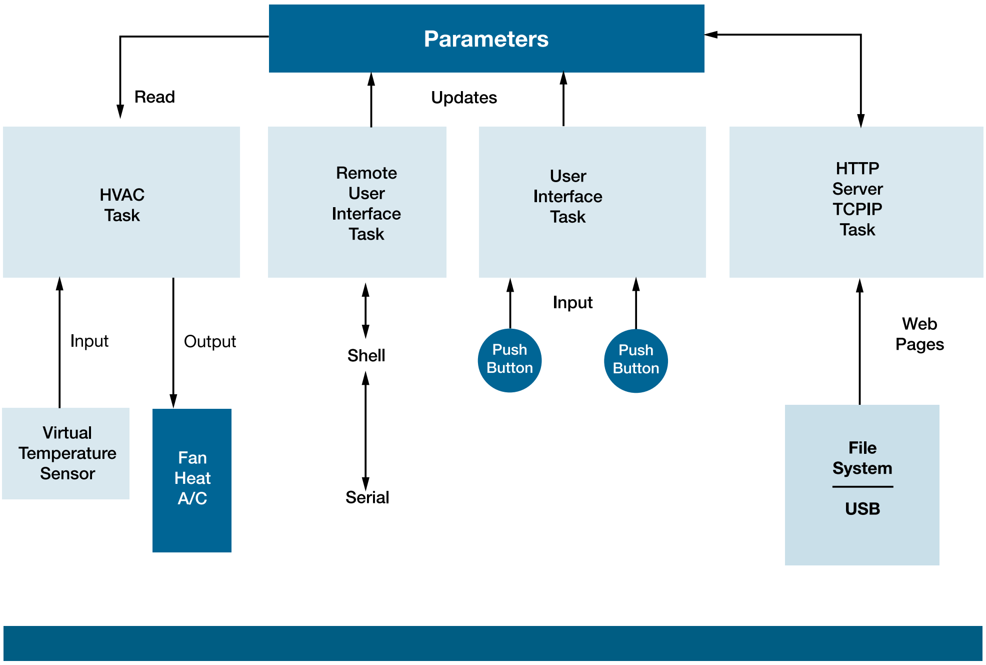


Figure 2: M52259DEMO board layout

MQX Application Software—Web Server Operation



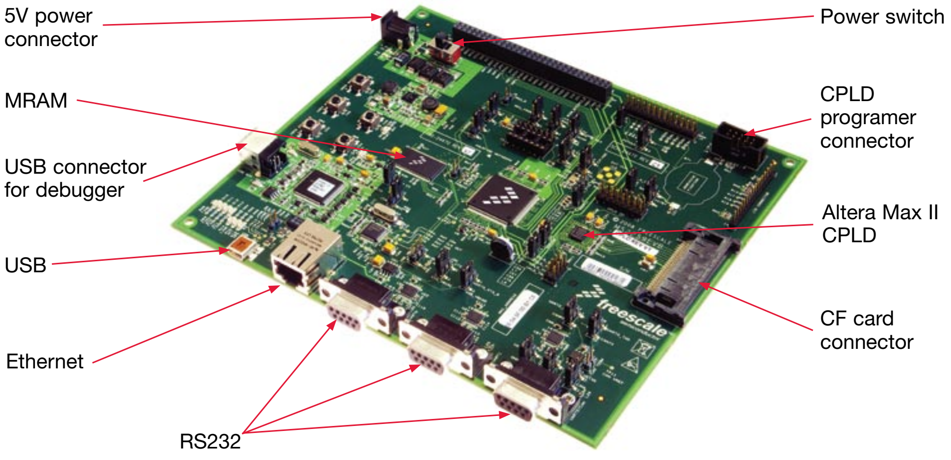
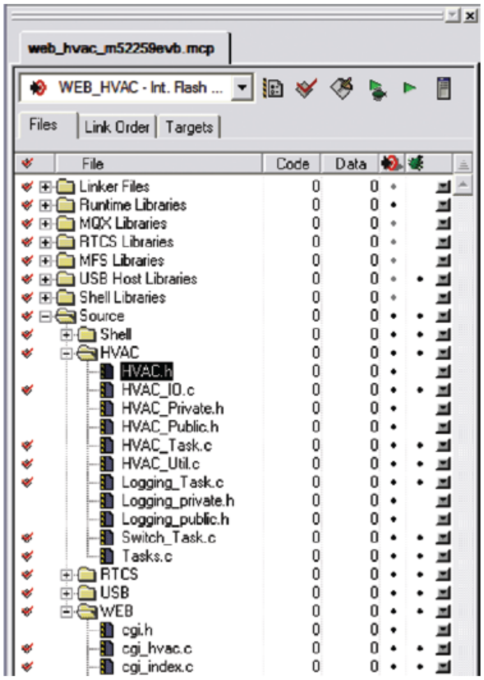


Figure 1: M52259EVB board layout

3. Open the Lab Project by selecting the **File > Open** menu item:
   - For M52259EVB board, open file: **c:\Program Files\Freescale MQX 3.0\demo\web_hvac\codewarrior\web_hvac_m52259evb.mcp**
   - For M52259DEMO board, open file: **c:\Program Files\Freescale MQX 3.0\demo\web_hvac\codewarrior\web_hvac_m52259demo.mcp**
4. The default IP address of the board is 169.254.3.3. Typically, when you connect your computer directly to the board, the computer will default to an auto IP address on the same subnet as the board (169.254.x.x), therefore requiring no setup. Note: The PC may take a few minutes to default to the auto IP address and make the connection.

However, if you have trouble connecting, you may configure the IP address of the computer manually. Select Start > Settings > Network Connections > Local Area Connection. Note your original TCP/IP settings, and then set your IP address to 169.254.3.4 and your subnet mask to 255.255.0.0.

5. Open the **HVAC.h** file in the CodeWarrior window. Double-click the file item located in the "Source/HVAC" group in the CodeWarrior project tree.


web_hvac_m52259evb.mcp

WEB_HVAC - Int. Flash ...

File	Code	Data
Linker Files	0	0
Runtime Libraries	0	0
MQX Libraries	0	0
RTCS Libraries	0	0
MFS Libraries	0	0
USB Host Libraries	0	0
Shell Libraries	0	0
Source	0	0
Shell	0	0
HVAC	0	0
HVAC.h	0	0
HVAC_IO.c	0	0
HVAC_Private.h	0	0
HVAC_Public.h	0	0
HVAC_Task.c	0	0
HVAC_Util.c	0	0
Logging_Task.c	0	0
Logging_private.h	0	0
Logging_public.h	0	0
Switch_Task.c	0	0
Tasks.c	0	0
RTCS	0	0
USB	0	0
WEB	0	0
cgi.h	0	0
cgi_hvac.c	0	0
cgi_index.c	0	0

Figure 3: MQX Source Tree

6. Locate the line of code starting with **#define ENET_IPADDR** and specify your target IP address by using the IPADDR macro. Set the target address to 169.254.3.3, and the line will be:

#define ENET_IPADDR IPADDR(169,254,3,3)

7. Do the same with the IP address mask value ENET_IPMASK:

#define ENET_IPMASK IPADDR(255,255,0,0)

8. Compile, download, and run the application and open a HyperTerminal window as was done in steps 6 to 13 of Lab 1.

9. Start your Internet browser and navigate to the target device address. In this case, 169.254.3.3.

10. You should see the Web-server welcome page in the browser window, as seen in Figure 4.

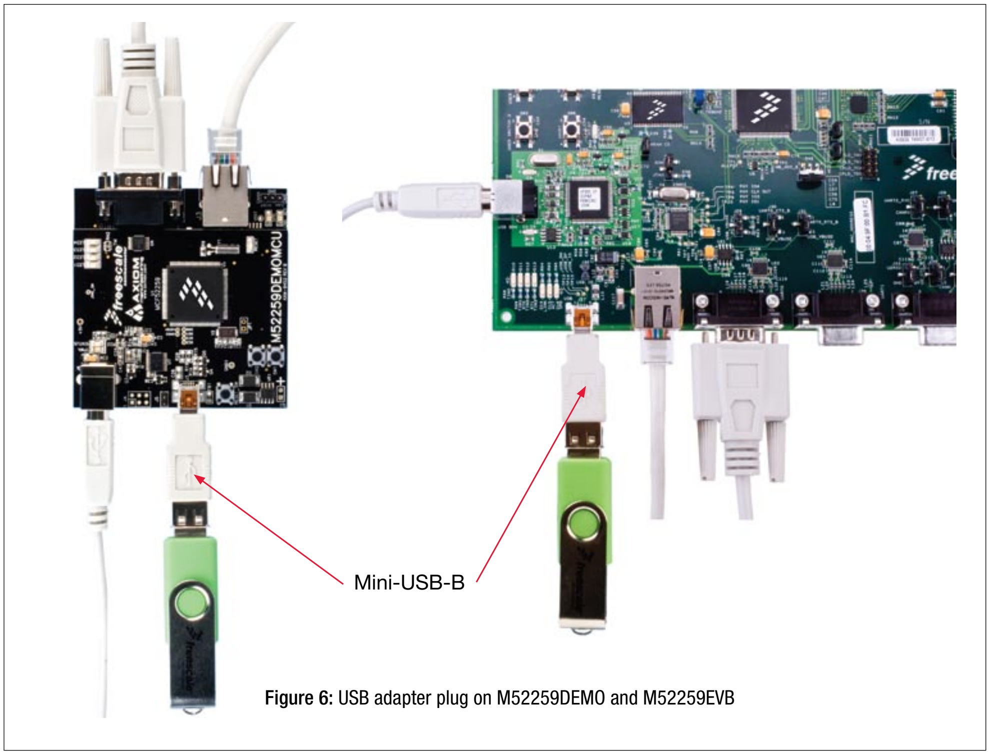
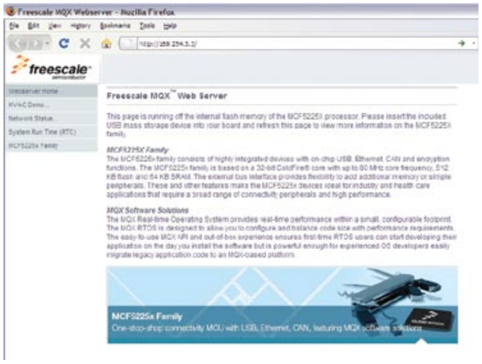


Figure 6: USB adapter plug on M52259DEMO and M52259EVB


Freescale MQX Webserver - Mozilla Firefox

http://169.254.3.3/

Freescale MQX™ Web Server

This page is running off the internal flash memory of the MCF5225X processor. Please insert the included USB mass storage device into your board and refresh this page to view more information on the MCF5225X family.

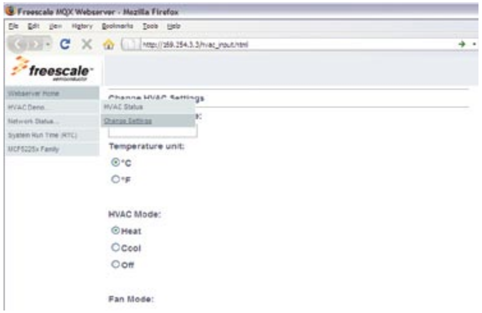
MCF5225X Family
The MCF5225x family consists of highly integrated devices with on-chip USB, Ethernet, CAN and encryption functions. The MCF5225x family is based on a 32-bit ColdFire® core with up to 80 MHz core frequency, 512 KB flash and 64 KB SRAM. The external bus interface provides flexibility to add additional memory or simple peripherals. These and other features make the MCF5225x devices ideal for industry and health care applications that require a broad range of connectivity peripherals and high performance.

MQX Software Solutions
The MQX Real-time Operating System provides real-time performance within a small, configurable footprint. The MQX RTOS is designed to allow you to configure and balance code size with performance requirements. The easy-to-use MQX API and out-of-box experience ensures first-time RTOS users can start developing their application on the day you install the software but is powerful enough for experienced OS developers easily migrate legacy application code to an MQX-based platform.

File Edit View History Bookmarks Tools Help
freescale
Webserver Home
HVAC Demo...
Network Status...
System Run Time (RTC)
MCF5225x Family
MCF5225x Family
One-stop-shop connectivity MCU with USB, Ethernet, CAN, featuring MQX software solutions

Figure 4: HVAC Web demo page running from internal memory

11. Navigate to the HVAC control page by moving your mouse over the HVAC Demo link on the left hand side, and selecting the "Change Settings" link. You can change the HVAC settings via that Web page. Figure 5 demonstrates how that web page will look.


Freescale MQX Webserver - Mozilla Firefox

http://169.254.3.3/hvac_input.html

Change HVAC Settings

Temperature unit:
◉ °C
○ °F

HVAC Mode:
◉ Heat
○ Cool
○ Off

Fan Mode:

Figure 5: Change HVAC setting via Website.

12. Similarly as in Labs 1-3, you can control the HVAC application by SW1 and SW2 push buttons. You should see the desired temperature changed on HVAC status Web page.

13. Return to the Web server's home page, by clicking the "Web server Home" link.

14. Insert the USB Memory Stick that is included in your kit into the USB connector on the EVB. You will need the mini USB-B adaptor plug. See Figure 6.

15. Reload the Web server home page by pressing F5 in the browser window. You should now see the "Browse USB Mass Storage Device" link on the left hand side as shown here.

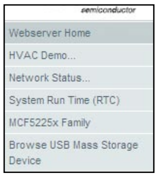
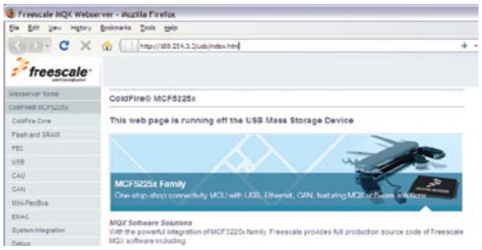


Figure 7.

16. Click the "Browse USB Mass Storage Device" link. You should see the web pages shown in Figure 8, that are located on the USB memory stick.


Freescale MQX Webserver - Mozilla Firefox

http://169.254.3.3/usb/index.htm

ColdFire® MCF5225x

This web page is running off the USB Mass Storage Device

MQX Software Solutions
With the powerful integration of MCF5225x family, Freescale provides full production source code of Freescale MQX software including

Figure 8: Browsing Web pages stored on USB mass storage device

LAB
5

Finding an Error Using Task Aware Debugging (TAD) in CodeWarrior

Covers both M52259EVB and M52259DEMOKIT

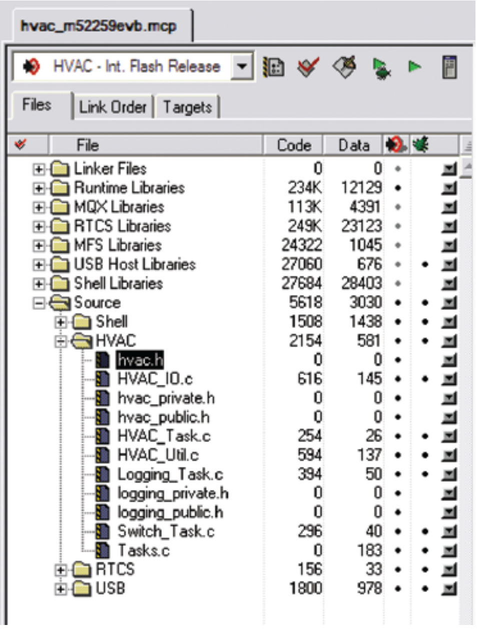
This lab will show you the power of using task-aware debugging to troubleshoot your application. An error has purposely been introduced into this project, and this lab will show you how to find and solve that error.

Demonstrates

- Sending messages between tasks (logging task)
- CodeWarrior task-aware debugging windows

Step by Step Instructions

1. Stop the application if it is currently running (**Debug>Kill**)
2. Open the Lab Project by selecting the **File > Open** menu item:
   - For M52259EVB board, open file: **c:\Program Files\Freescale MQX 3.0\demo\ hvac_error \codewarrior\hvac_error_m52259evb.mcp**
   - For M52259DEMO board, open file: **c:\Program Files\Freescale MQX 3.0\demo\ hvac_error\codewarrior\hvac_error_m52259demo.mcp**
3. Enable the Auto Logging feature. This is done by opening the **hvac.h** file



hvac_m52259evb.mcp

HVAC - Int. Flash Release

File	Code	Data		
Linker Files	0	0	●	
Runtime Libraries	234K	12129	●	
MQX Libraries	113K	4391	●	
RTCS Libraries	249K	23123	●	
MFS Libraries	24322	1045	●	
USB Host Libraries	27060	676	●	●
Shell Libraries	27684	28403	●	
Source	5618	3030	●	●
Shell	1508	1438	●	●
HVAC	2154	581	●	●
hvac.h	0	0	●	
HVAC_IO.c	616	145	●	●
hvac_private.h	0	0	●	
hvac_public.h	0	0	●	
HVAC_Task.c	254	26	●	●
HVAC_Util.c	594	137	●	●
Logging_Task.c	394	50	●	●
logging_private.h	0	0	●	
logging_public.h	0	0	●	
Switch_Task.c	296	40	●	●
Tasks.c	0	183	●	●
RTCS	156	33	●	●
USB	1800	978	●	●

Figure 1: MQX Source Tree

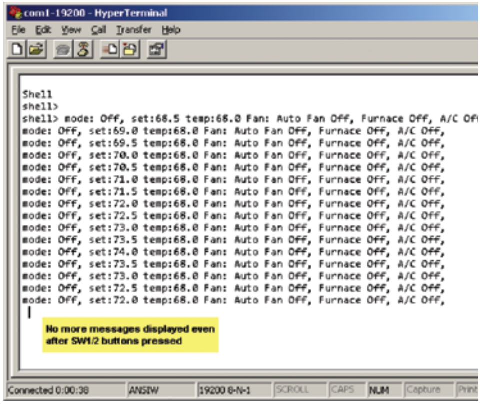
4. Then change the auto logging define:

   #define DEMOCFG_ENABLE_AUTO_LOGGING 0

to this:

   #define DEMOCFG_ENABLE_AUTO_LOGGING 1

5. Notice that the impacted files have a red check mark beside them to indicate that they need to be re-compiled, or are 'touched'. Compile, download, and run the application as was done in steps 6 to 13 of Lab 1.
6. Go to the shell console in HyperTerminal. Note that a string of logging information will be printed out every 15 seconds. It will also be printed out when there is an update to any of the parameters such as the desired temperature. You can test this by pressing SW1.
7. Press SW1 until the desired temperature gets to 24°C and then use SW2 to bring it back down to 20°C. You should notice that the logging will eventually stop and that no more updates are printed.



com1-19200 - HyperTerminal

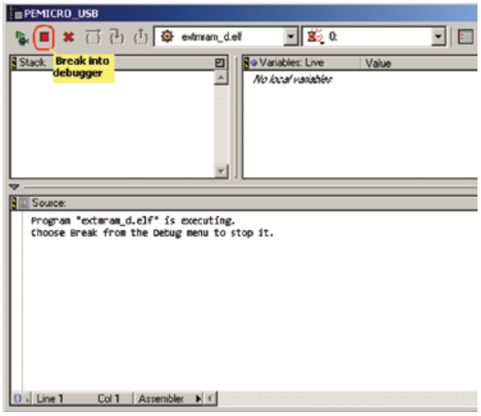
```
Shell
shell>
shell> mode: Off, set:68.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off
mode: Off, set:69.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:69.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:70.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:70.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:71.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:71.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:72.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:72.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:73.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:73.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:74.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:73.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:73.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:72.5 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
mode: Off, set:72.0 temp:68.0 Fan: Auto Fan Off, Furnace Off, A/C Off,
```

No more messages displayed even after SW1/2 buttons pressed

Connected 0:00:38 | ANSIW | 19200 8-N-1 | SCROLL | CAPS | NUM | Capture | Print

Figure 2: Up to 24, on the way down it stops logging

8. Your job now is to use the task aware debugging (TAD) feature to check for errors to determine why this error is happening. To see the TAD data, pause the application by clicking on the **Break** icon (which is a red square) or by selecting **Break** in the **Debug** menu.



PEMICRO_USB

extmram_d.elf

0:

Break into debugger

No local variables

```
Program "extmram_d.elf" is executing.
Choose Break from the Debug menu to stop it.
```

Figure 3: Break application execution

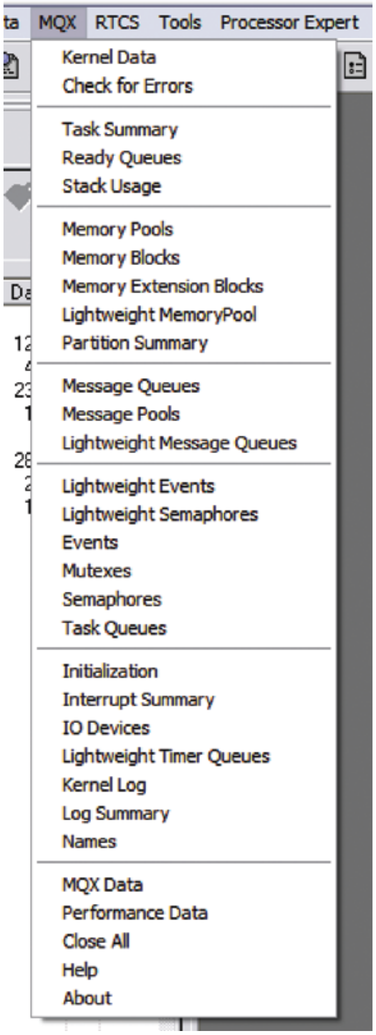


Figure 4: MQX Task-Aware Debugging menu

9. Then click on the **MQX** pull down menu. Read down the list of available information windows and select the one(s) that you might think would be a good indication as to what happened.
10. One of the TAD windows which will quickly help you to get an application status overview is Task Summary or Check for Errors.

Solution

The HVAC task is using MQX messages to send data to the Log task (see HVAC_LogCurrentState() and the Log() functions) in **HVAC_Util.c** and **Logging_Task.c**. The HVAC task, as the message sender, assumes each message is "consumed" by the Log task and removed from the message pool after the text is printed to the console.

From the HVAC task error code found in the Task Summary TAD window seen in Figure 5, it is apparent that a message could not be sent because the message pool is full. You can verify this assumption by showing the Message Pools TAD window (Figure 6) and double-clicking on the one and only message pool entry to bring up the window shown in Figure 7.

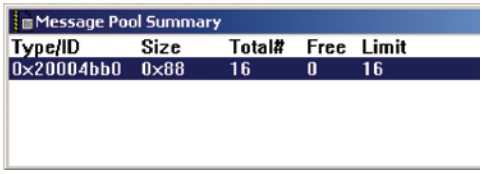

```
Message Pool Summary
Type/ID     Size   Total#  Free  Limit
0x20004bb0  0x88   16      0     16
```

Figure 6: Message Pool Summary TAD window

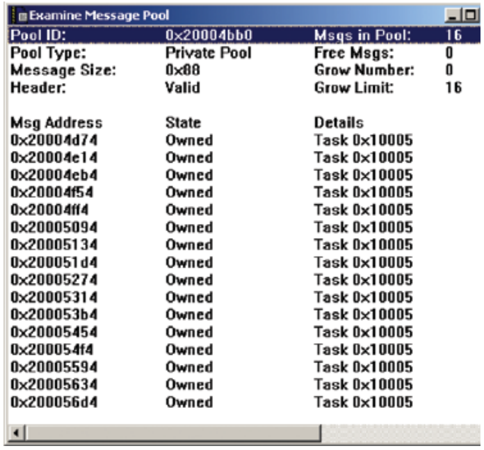

```
Examine Message Pool
Pool ID:        0x20004bb0       Msgs in Pool:   16
Pool Type:      Private Pool     Free Msgs:      0
Message Size:   0x88             Grow Number:    0
Header:         Valid            Grow Limit:     16

Msg Address     State            Details
0x20004d74      Owned            Task 0x10005
0x20004e14      Owned            Task 0x10005
0x20004eb4      Owned            Task 0x10005
0x20004f54      Owned            Task 0x10005
0x20004ff4      Owned            Task 0x10005
0x20005094      Owned            Task 0x10005
0x20005134      Owned            Task 0x10005
0x200051d4      Owned            Task 0x10005
0x20005274      Owned            Task 0x10005
0x20005314      Owned            Task 0x10005
0x200053b4      Owned            Task 0x10005
0x20005454      Owned            Task 0x10005
0x200054f4      Owned            Task 0x10005
0x20005594      Owned            Task 0x10005
0x20005634      Owned            Task 0x10005
0x200056d4      Owned            Task 0x10005
```

Figure 7: Message Pool Examined
(after double-clicking the pool entry)

The message pool is exhausted. The problem is on the receiving side, as it is always the message receiver's responsibility to reuse the message object or free it when no more needed.

Looking at the Logging_task() function, located in **Logging_Task.c**, you can see the message is received by

msg_ptr = _msgq_receive(log_qid, 0);

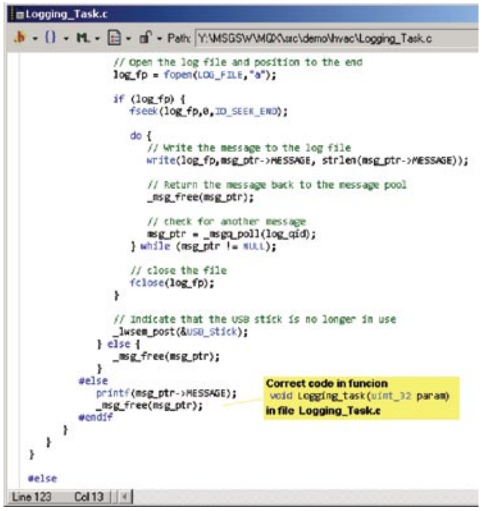
and further the data of the message (the log text) is printed

printf(msg_ptr->MESSAGE);

What is missing is deletion of the message after the log text is printed:

_msg_free(msg_ptr);

Add this line, recompile and run the application. The message memory will now be released after the message is printed out.



```
            // Open the log file and position to the end
            log_fp = fopen(LOG_FILE,"a");

            if (log_fp) {
               fseek(log_fp,0,IO_SEEK_END);

               do {
                  // Write the message to the log file
                  write(log_fp,msg_ptr->MESSAGE, strlen(msg_ptr->MESSAGE));

                  // Return the message back to the message pool
                  _msg_free(msg_ptr);

                  // check for another message
                  msg_ptr = _msgq_poll(log_qid);
               } while (msg_ptr != NULL);

               // close the file
               fclose(log_fp);
            }

            // Indicate that the USB stick is no longer in use
            _lwsem_post(&USB_Stick);
         } else {
            _msg_free(msg_ptr);
         }
      #else
         printf(msg_ptr->MESSAGE);
         _msg_free(msg_ptr);
      #endif
    }
  }
}

#else
```

Correct code in funcion
void Logging_task(uint_32 param)
in file Logging_Task.c

Figure 8: Correct code

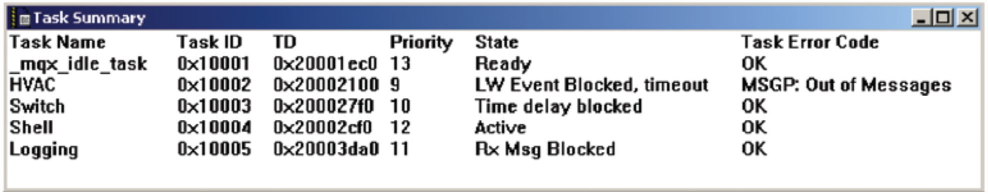

```
Task Summary
Task Name      Task ID   TD          Priority  State                      Task Error Code
_mqx_idle_task 0x10001   0x20001ec0  13        Ready                      OK
HVAC           0x10002   0x20002100  9         LW Event Blocked, timeout  MSGP: Out of Messages
Switch         0x10003   0x200027f0  10        Time delay blocked         OK
Shell          0x10004   0x20002cf0  12        Active                     OK
Logging        0x10005   0x20003da0  11        Rx Msg Blocked             OK
```

Figure 5: Task Summary TAD window

LAB 6

Ethernet to Serial Bridge, Freescale MQX RTCS

Covers both M52259EVB and M52259DEMOKIT

This lab demonstrates how to create a bridge between a TCP/IP (telnet) connection and a serial line.

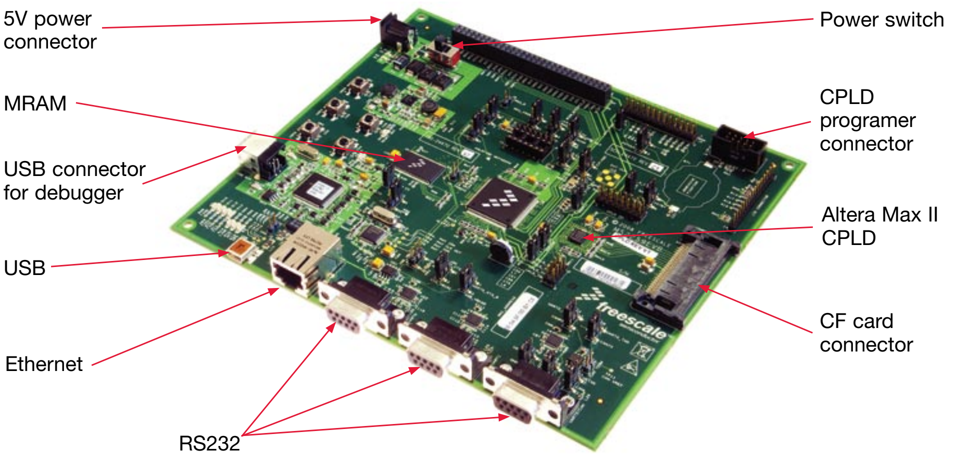


Figure 1: M52259EVB board layout

Demonstrates

- MQX RTCS TCP/IP network stack
- Custom telnet server implementation
- Re-directing STDIN and STDOUT output within an MQX task

Step by Step Instructions

1. Make the following connections from the MCF52259 board to the computer. See Figure 1 and Figure 2.
   a. USB debugger connection to a USB port on PC
   b. Serial Port 0 (UART 0) to a Serial Port on PC
   c. A crossover Ethernet cable between the board and an Ethernet port on your computer
   d. Attach power cable and turn on the board's power switch (M52259EVB only)
2. The first time you connect the USB debugger cable to your PC, Windows will install a driver for the debugger. Follow the prompts to automatically detect and install the driver.

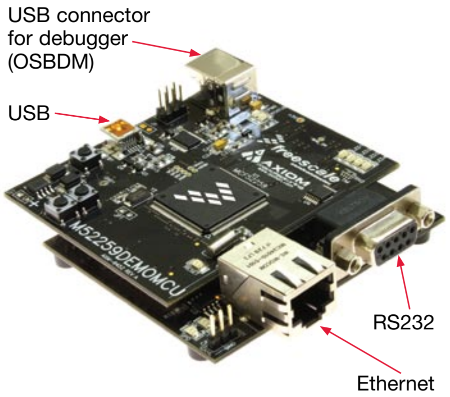
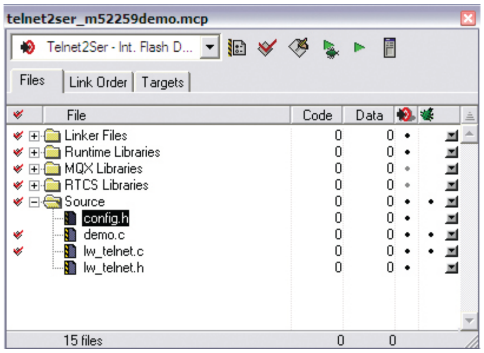


Figure 2: M52259DEMO board layout

3. Open the Lab Project by selecting the **File > Open** menu item:
   - For M52259EVB board, open file: **c:\Program Files\Freescale MQX 3.0\demo\telnet_to_serial\codewarrior\telnet2ser_m52259evb.mcp**
   - For M52259DEMO board, open file: **c:\Program Files\Freescale MQX 3.0\demo\telnet_to_serial\codewarrior\telnet2ser_m52259demo.mcp**
4. The default IP address of the board is 169.254.3.3. Typically, when you connect your computer directly to the board, the computer will default to an auto IP address on the same subnet as the board (169.254.x.x), therefore requiring no setup. Note: The PC may take a few minutes to default to the auto IP address and make the connection.

   However, if you have trouble connecting, you may configure the IP address of the computer manually. Select Start > Settings > Network Connections > Local Area Connection. Note your original TCP/IP settings, and then set your IP address to 169.254.3.4 and your subnet mask to 255.255.0.0.



File	Code	Data
Linker Files	0	0
Runtime Libraries	0	0
MQX Libraries	0	0
RTCS Libraries	0	0
Source	0	0
config.h	0	0
demo.c	0	0
lw_telnet.c	0	0
lw_telnet.h	0	0
15 files	0	0

Figure 3: MQX Source Tree

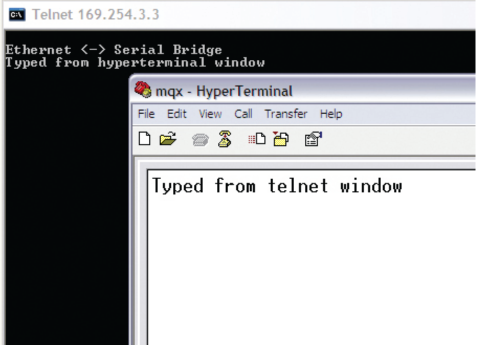
5. Open the **config.h** file in the CodeWarrior window as shown in Figure 3. Double click the file item located in the “Source” group in the CodeWarrior project tree.
6. Locate the line of code starting with **#define ENET_IPADDR** and specify your target IP address by using the IPADDR macro. Set the target address to 169.254.3.3, and the line will be:

#define ENET_IPADDR IPADDR(169,254,3,3)

7. Do the same with the IP address mask value ENET_IPMASK:

#define ENET_IPMASK IPADDR(255,255,0,0)

8. Compile, download, and run the application and open a HyperTerminal window as was done in steps 6 to 13 of Lab 1.
9. Open a Command Prompt on the PC *(Start > All Programs > Accessories > Command Prompt)*. At the prompt invoke a telnet session to the board by typing **telnet 169.254.3.3** You will be connected to the MQX shell via telnet.
10. Now the serial console and the telnet sessions should be “bridged.” Type some characters into the telnet session, and you should see the characters appearing on the console terminal window. See Figure 4 for how it will appear.



```
Ethernet <-> Serial Bridge
Typed from hyperterminal window
```

```
Typed from telnet window
```

Figure 4: Ethernet to Serial Bridge

11. Then try typing into the console terminal window, and you should see the characters appearing in the telnet session.

Learn More: For more information about MQX and Freescale Solutions, please visit **www.freescale.com/mqx**.