Getting Started with Freescale MQX™ RTOS

PRODUCT:	Freescale MQX™ RTOS
PRODUCT VERSION:	3.8.0
DESCRIPTION:	Getting Started with the Freescale MQX™ RTOS, version 3.8.0
RELEASE DATE:	December 15th, 2011

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan**:**
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Rev. 08
12/2011

Table of Contents

1 Read Me First

This document describes steps required to configure the CodeWarrior development tool and use it to build, run and debug applications of the Freescale MQX™ RTOS operating system. This document also provides board-specific information related to the MQX RTOS.

2 Building the MQX Libraries

2.1 Compile-time Configuration

Major compile-time configuration options are centralized in a single user configuration file located in

```
<install_dir>/config/<board>/user_config.h
```

This user configuration file is included internally by private configuration files in MQX PSP and BSP projects.

To share configuration settings between different boards, the user_config.h file may include other header files with common settings. The header files may only be located in the same <*board*> directory or in the "common" directory:

```
<install_dir>/config/common
```

All MQX configuration files are also *indirectly* used by other core components like RTCS, MFS, etc. "Indirectly" means that the MQX PSP and BSP must be build first, which causes the configuration file being copied into the output (lib) directory. The other components then include the configuration file from the `/lib` output directory.

Caution: Until the PSP or BSP libraries are rebuilt, configuration changes made in the `user_config.h` file are not used by any other MQX component. On the other hand, after the PSP and BSP libraries are re-compiled with a new configuration, it is important to recompile the other libraries so the compiled code is consistent with the configuration file. See the next section for more details.

2.2 Build Process

After any change to the compile-time user configuration file or MQX kernel source files, the MQX libraries need to be re-built. The build process is similar with all core components:

- The output directory is `<install_dir>/lib/<board>.<compiler>/<component>` (where `<compiler>` is a name of build tool).

  For example when CodeWarrior tool is used to build MQX PSP and BSP libraries for M52259EVB board, the libraries are built into the `/lib/m52259evb.cw/bsp` and `/lib/m52259evb.cw/psp` directories.

  Note: The BSP and PSP library output path was changed in MQX 3.7.0. In earlier versions both BSP and PSP libraries were built into a common folder: `/lib/m52259evb.cw/mqx`. See the "Freescale MQX^{TM} Porting Guide" - *FSL_MQX_Porting_Guide.pdf* document for instructions how to port old MQX projects.
- All public header files needed by application are automatically copied from internal include folders to the same output directory as the library itself.
- During PSP or BSP build process, also the `user_config.h` file and other header files from the `config/<board>` and `config/common` directories are copied into the `lib/<board>.<compiler>` output directory.
- Other components like RTCS, MFS, etc. use the copied configuration files only.

- Similarly as the PSP and BSP, the other libraries build into the output directories inside `/lib/<board>.<compiler>`

To summarize the points above, there are simple **rules to obey** when re-building the MQX libraries.

- After any change to the `/config/common/user_config.h` file, all MQX libraries should be re-built.
- The PSP and BSP libraries must be build first, before the MFS, RTCS and other libraries.

Important: No changes should be made to header files in the output build directory (`/lib`). The files get overwritten any time the libraries are built.

2.3 Build Targets

Each build project in Freescale MQX™ RTOS contains multiple compiler/linker configurations (so called build „targets“).

Two different types of build targets exist for different compiler optimization settings:

- **Debug** – the compiler optimizations are turned off or set to low. The compiled code is easy to debug but may be less effective and much larger than the Release build. All output libraries (or executables) have **_d** postfix in the file name (e.g. `rtcs_d.a`).
- **Release** – the compiler optimizations are set to maximum. The compiled code is very hard to debug and should be used for final applications only. There is no postfix in the output file name (e.g. `rtcs.a`).

Similarly, different types of build targets exist for different application binary interface (ABI) settings. In MQX version 3.6, the StdABI targets are no-longer available as they are not supported by the latest versions build tools.

- **StdABI** – the compiler is set to generate functions calls with standard parameter (on-stack) passing interface. This was the only available option in the MQX versions prior 3.4.
- **RegABI** – starting in MQX release 3.4, the register parameter passing is enabled and is set as default in all example and stationery projects. The output libraries for the CodeWarrior builds have the "_regabi" postfix in the file name (e.g. `rtcs_regabi.a`). Register parameter passing option gives generally smaller and faster code.

Considering the naming convention stated above, the following build targets exist in MQX library build projects:

- **Debug RegABI** – register parameter passing ABI, easy to debug code
- **Release RegABI** – register parameter passing ABI, optimization set to maximum
- **Build all** – this target contains all other targets for convenient mass-build.

Note: Build configurations of tools which only support one ABI do not contain this postfix in the configuration name, nor in the output library name.

Build target name of any MQX application project makes a reference either to **Debug** or **Release** builds of the core libraries. On top of that the target names also specify board memory configuration which gets built. For example:

- Devices with internal Flash memory (e.g. MCF522xx):
  - **Int. Flash Release** – this target is suitable for final application deployment. When programmed to Flash, the application starts immediately after reset. Variables are allocated in internal SRAM memory.

  - **Int. Flash Debug** – same as above, only the Debug-compiled libraries are used. This target is suitable for debugging before deployment. On boards without external memory, this is the only target suitable for debugging larger applications.
- Boards with external MRAM memory (M52259EVB etc.):
  - **Ext. MRAM Debug** – solely for debugging purposes with code located in external MRAM memory (available e.g. on the M52259EVB). Variables are located in internal SRAM. Part of the external MRAM memory may also be used for additional RAM memory pools. Application executable is loaded to MRAM automatically by the debugger.
- Boards with external RAM memory (M54455EVB etc.):
  - **Ext. Ram Debug** – solely for debugging purposes with code located in external RAM memory (available as SDRAM e.g. on the M54455EVB). Both code and variables are located in this external memory. Application executable is loaded to RAM automatically by the debugger.
- Boards and devices with internal Flash memory and additional external RAM for data (TWR-K70F120M):
  - **Int Flash <mem>Data Debug –** The name of each target additionally defines a memory used as the default data storage. For example the application built with target named “Int Flash DDRData Debug“ will execute code out of internal Flash memory and will use the DDR memory for data storage.

2.4 Freescale CodeWarrior Development Studio “Classic”

Currently supported CW “Classic” versions:

- CodeWarrior for ColdFire V7.2 (ColdFire V2-V4 families)
- CodeWarrior for Microcontrollers V6.3 (ColdFire V2-V4 families)
- CodeWarrior for mobileGT V9.2 (PowerPC MPC5125)
- CodeWarrior for Power Architecture V8.8 (PowerPC MPC8308)

The support of the following CodeWarrior tools was discontinued in MQX 3.6.0

- CodeWarrior for Microcontrollers V6.2 (ColdFire V2-V4 families)
- CodeWarrior for ColdFire V7.1 (ColdFire V2-V4 families)

If you upgrade your toolset to the latest CodeWarrior version or if you upgrade MQX applications developed in MQX versions 3.5.0 or earlier, you should be aware of the following facts:

- Starting from the MQX version 3.5.0 the support for register parameter passing (RegABI) was introduced and was made the default option for all precompiled libraries. MQX version 3.6.0 has discontinued the support for the StdABI targets at all, as it is no longer supported by the latest CodeWarrior tools.
- If an older application project is upgraded to the latest MQX version, it should be changed to reference the new library names.
- The user’s code written in assembly language has to be reviewed in case the parameter passing option has been changed in application project. See CodeWarrior documentation for more details.

- Remember the MQX libraries and user applications always have to use the same parameter passing option.
- In transition to versions 7.2 and 6.3, the CodeWarrior has also changed its runtime libraries. The new EWL libraries have been introduced, replacing the former MSL libraries. The EWL libraries are implicitly liked to the target application according to project configuration options. All old MSL libraries should be removed from project. The libraries are API level compatible.

To rebuild the MQX libraries open and build the following CodeWarrior project:

`<install_dir>/config/<board>/<compiler>/build_libs.mcp`

The output directory is:

`<install_dir>``/lib/`*`<board>`*`.`*`<compiler>`*`/`*`<component>`*

Caution: Pay attention to use correct CodeWarrior Development Studio when opening any MQX projects. There are different CodeWarrior Studios for ColdFire V1 family (MCF51xx), ColdFire V2-V4 families (MCF52xx-MCF54xx), PowerPC (MCF830x) and PowerPC-mobileGT (MPC5125) processors.

2.5 Freescale CodeWarrior Development Studio version 10.x

In order to recompile the MQX core libraries the library projects have to be imported (opened) into the CodeWarrior 10.x workspace. Select menu *File/Import/General/Existing Projects into Workspace* and navigate to your MQX installation directory. Select and import the following projects (note that some of the projects may not be available for all supported boards):

- **BSP project** - `<mqx_path>\mqx\build\cw10\bsp_<board>`
- **PSP project** - `<mqx_path>\mqx\build\cw10\psp_<board>`
- **MFS project** - <`mqx_path>\mfs\build\cw10\mfs_<board>`
- **RTCS project** - `<mqx_path>\rtcs\build\cw10\rtcs_<board>`
- **Shell project** - `<mqx_path>\shell\build\cw10\shell_<board>`
- **USB DDK project** - `<mqx_path>\usb\device\build\cw10\usb_ddk_<board>`
- **USB HDK project** - `<mqx_path>\usb\host\build\cw10\usb_hdk_<board>`

Warning: The "Copy projects into workspace" check box the in the file importer must be **unchecked**. Accidental leaving this option enabled will corrupt the project and the libraries will not be able to build.

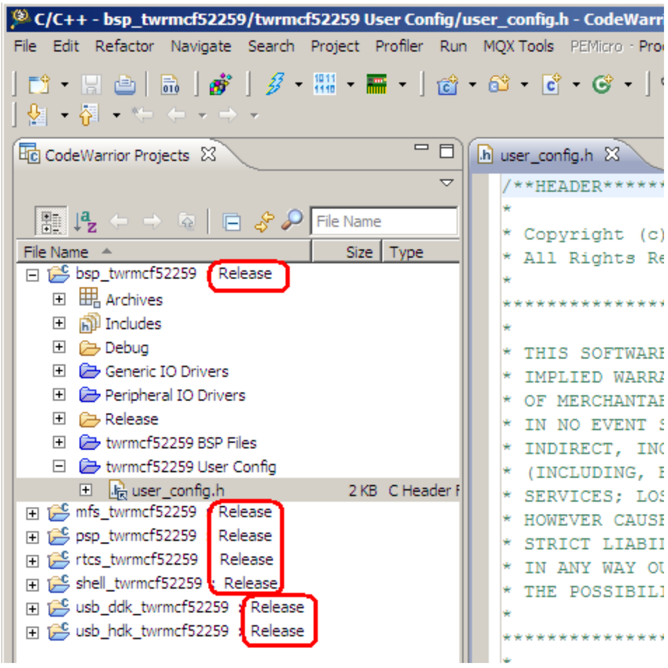
Note: In CodeWarrior 10.2 and MQX 3.8 or later, the MQX installation contains *Working Set Importer* plugin which simplifies importing the MQX libraries into the CodeWarrior workspace. Go to the `<mqx_path>\config\<board>\cw10` folder and simply drag the `<board>.wsd` file into the CodeWarrior project view. The projects will be imported with references and build configurations set up properly. You can also use the plug-in to export and import your custom working sets of projects between different workspaces or even different computers.

When all library projects are imported in your workspace, use mouse to select project items and select menu *Project/Build all*. The libraries have to be built in correct order:

- PSP and BSP
- MFS
- RTCS
- Shell and USB libraries

CodeWarrior compiles all MQX libraries in the build configuration which is currently selected (build configuration is also referred as “build target”). By default, the Debug build target is selected in all projects. In order to compile libraries in a Release configuration switch all libraries to Release target and build all again. See picture bellow.

See *FSL_MQX_in_CW_10_x.pdf* document for detailed information about importing and building MQX libraries and debugging MQX application in CodeWarrior 10.x. This document is a part of the MQX installation package.


C/C++ - bsp_twrmcf52259/twrmcf52259 User Config/user_config.h - CodeWarr
File Edit Refactor Navigate Search Project Profiler Run MQX Tools PEMicro Pro
CodeWarrior Projects
user_config.h
File Name
Size Type
bsp_twrmcf52259 Release
Archives
Includes
Debug
Generic IO Drivers
Peripheral IO Drivers
Release
twrmcf52259 BSP Files
twrmcf52259 User Config
user_config.h 2 KB C Header f
mfs_twrmcf52259 Release
psp_twrmcf52259 Release
rtcs_twrmcf52259 Release
shell_twrmcf52259 Release
usb_ddk_twrmcf52259 Release
usb_hdk_twrmcf52259 Release

```
/**HEADER*****
*
* Copyright (c)
* All Rights Re
*
**************
*
* THIS SOFTWARE
* IMPLIED WARRA
* OF MERCHANTAE
* IN NO EVENT S
* INDIRECT, INC
* (INCLUDING, E
* SERVICES; LOS
* HOWEVER CAUSE
* STRICT LIABII
* IN ANY WAY OU
* THE POSSIBILI
*
**************
```

2.6 IAR Embedded Workbench for ARM and ColdFire

The Freescale MQX RTOS provides support for following IAR Embedded Workbench versions:

- IAR EWARM version 6.30.1 (Kinetis ARM Cortex® M4)
- IAR EWCF version 5.3 (ColdFire M52259EVB, build projects are not part of release – contact IAR Systems for more information)

The support of the IAR EWARM version 5.50.6 tools was discontinued in MQX 3.7.0.

To rebuild the MQX libraries, open and batch-build the following IAR EWARM workspace:

<mqx_install_dir>/config/<board>/iar/ build_libs.eww

The output directory is:

`<mqx_install_dir>/lib/<board>.iar`

See detailed information about the MQX support in IAR tools in separate document *"Getting Started with Freescale MQX™ RTOS and IAR Embedded Workbench".* The document is included in the MQX installation in the `<mqx_install_dir>/doc/tools/iar/MQX-IAR-Getting-Started.pdf`.

2.7 ARM Keil uVision4 Development Environment Support

The Freescale MQX RTOS provides support for ARM Keil uVision4 (tested with version 4.22)

To rebuild the MQX libraries, open and batch-build the following uVision4 Multi-Project Workspace:

`<mqx_install_dir>/config/<board>/uv4/ build_libs.uvmpw`

The output directory is:

`<mqx_install_dir>/lib/<board>.uv4`

See detailed information about the MQX support in ARM tools in separate document *"Getting Started with Freescale MQX™ RTOS and MDK-ARM Keil uVision4".* The document is included in the MQX installation in the `<mqx_install_dir>/doc/tools/uv4/MQX-uVision4-Getting-Started.pdf`.

2.8 CodeSourcery G++ Development Environment Support

MQX support for GNU C and C++ based build tools is provided by CodeSourcery.

3 Creating New MQX Project

3.1 Freescale CodeWarrior Development Studio “Classic”

The Freescale MQX™ RTOS setup installs the MQX “New Project Stationeries” into CodeWarrior installation directory. New project stationeries are project templates which help you to create your application project.

Follow the steps bellow to create new MQX project:

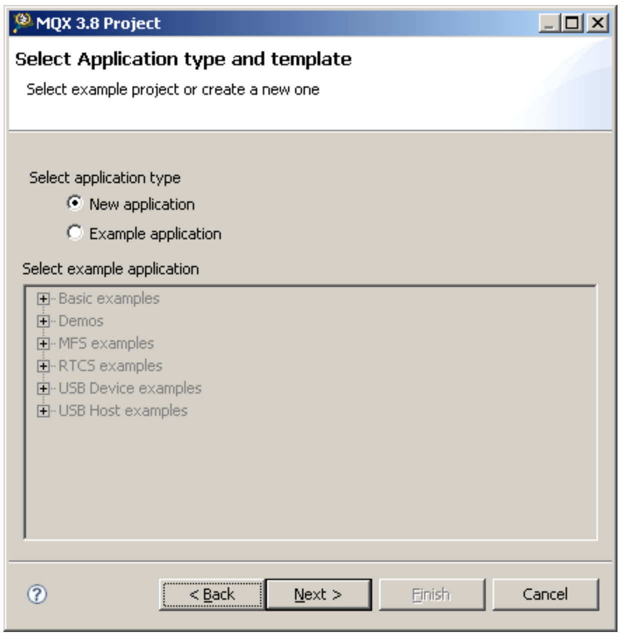
- Open CodeWarrior Development Studio and select the *File/New Project…* menu
- Select “Freescale MQX Stationery” and specify a name of your project
- Select Target platform and type of application you are developing for
- New application will be created for you. Note: All application projects (even the “MQX Only” one) includes the whole set of MQX libraries. This allows convenient usage of all MQX API functions without the need of changing project. The unused library code is stripped during final application linking. In case you do not need a certain library in your application, simply delete the library reference from the project.

3.2 Freescale CodeWarrior Development Studio version 10.x

The Freescale MQX™ RTOS setup installs the MQX “New Project Wizard” plug-in into CodeWarrior 10.x installation directory. The Project Wizard helps to select one of the supported evaluation boards, include appropriate MQX libraries and create initial application project.

The project wizard is available in the *File/New/MQX Projects* menu in the CodeWarrior 10.x IDE.

Follow the steps displayed by the Wizard, specify application name and select the target evaluation board. The wizard may create a new application or import one of existing examples to your workspace. Choose the “New application”:



MQX 3.8 Project

Select Application type and template

Select example project or create a new one

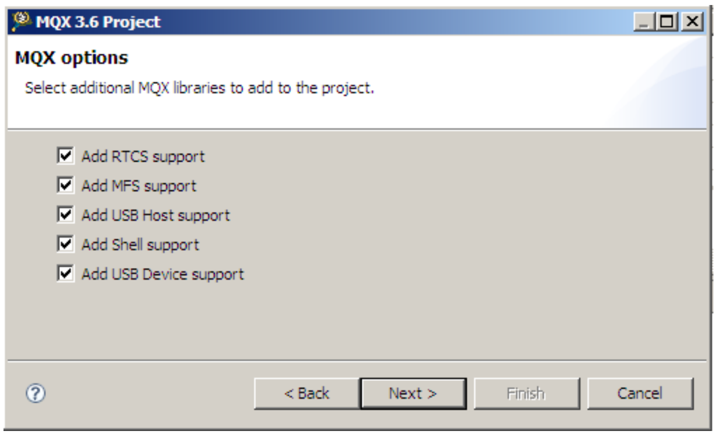
Select application type

[✓ New application]

Example application

Select example application

Select the MQX libraries to be included in your project:


MQX 3.6 Project

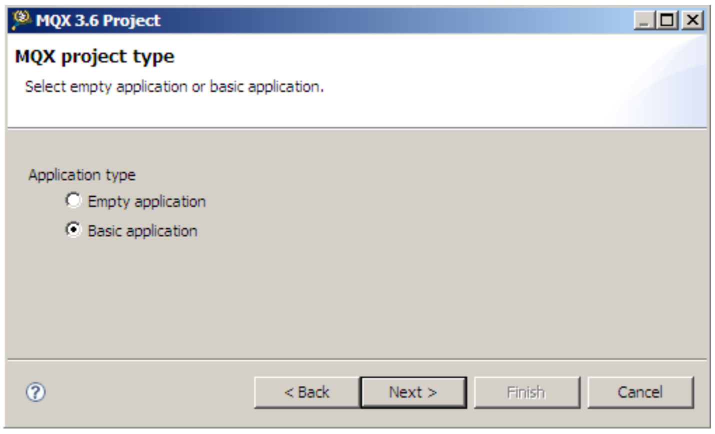
MQX options

Select additional MQX libraries to add to the project.

- [✓] Add RTCS support
- [✓] Add MFS support
- [✓] Add USB Host support
- [✓] Add Shell support
- [✓] Add USB Device support

The Wizard offers two types of project:

- Empty application – a simple “Hello world” application with selected MQX libraries included
- Basic application – an application showing basic code to initialize selected MQX components


MQX 3.6 Project

MQX project type

Select empty application or basic application.

Application type

- Empty application
- [✓ Basic application]

- If the “Basic” application is selected, the Wizard continues by several other steps to gather information about what initialization code to generate:
    - RTCS option – RTCS TCP IP stack is initialized and set to static or dynamic IP address based on user selection.
    - USB Device or Host option – Stack and selected class drivers are initialized based on user selection.
    - MFS option – the RAM-disk initialization code can be enabled

3.3 Other Development Tools

Process of creating new projects typically heavily depends on the development environment being used. Describing this process for tools other than Freescale CodeWarrior is out of scope of this document. For more information, refer to MQX support documentation provided by the tool vendor.

A general recommendation for starting a new MQX project in any environment is to clone one of the existing example applications, save it under a custom name and modify it to meet your specific needs. In this case, please be aware that there may be relative paths to support files referred in the project. This may apply to include search paths, linker command files, debugger configuration files etc. Make sure, you update the relative paths in the new “clone” of the project.

4 MQX Standard Input and Output Channel Setting

4.1 Default IO Channel

One of the I/O communication devices installed by MQX BSP may be used as the standard IO channel (e.g Serial line, IO Debug, telnet, USB CDC). The default console setting for each supported development board is described in the 7 *Board-specific Information Related to MQX*.

The Freescale Tower evaluation kits offer several ways to connect IO console. Most common options are listed below.

- Serial IO channel routed via TWR-SER or TWR-SER2 boards using RS232 connector. This port can be connected directly to the PC serial interface and used with a suitable terminal program (e.g. Hyper-Terminal)
- Serial IO channel routed via combined debugging and communication port (OSBDM/OSJTAG with build-in USB to Serial functionality) directly on the board. On the microcontroller side, the communication interface is connected to one of serial (SCI/UART) ports. On the PC-Host side you can use either a virtual USB serial port driver or a special USB communication terminal application. Refer to the development board documentation more details.
- DebugIO channel connecting PC and target processor using debugging probe IO functionality. See chapter: *Using the DebugIO Driver as the Default IO* Channel

Pre-defined default IO channel setting is specified for each board in the `mqx\source\bsp\<board_name>\board_name.h` header file. This setting can be overridden in the `user_config.h` file by adding the following code and rebuilding BSP library.

To set the default IO channel to the serial line to UART2 (mapped to `ttyc:`) use:

```
#define BSP_DEFAULT_IO_CHANNEL              "ttyc:"
#define BSP_DEFAULT_IO_CHANNEL_DEFINED
```

Ensure that the serial channel (ttyc: in this case) in enabled in `user_config.h` file:

```
#define BSPCFG_ENABLE_TTYC        1
```

4.2 Using the DebugIO Driver as the Default IO Channel

The standard input and output channel can be redirected to the DebugIO driver allowing processor to communicate with computer via the debugger probe. The MQX RTOS currently supports ARM CortexM Semihost and ITM technologies. Note that the communication has to be properly set up in the debugger on the PC host side.

The default IO channel to can be set to the DebugIO adding following code to the `user_config.h`

```
#define BSP_DEFAULT_IO_CHANNEL                "iodebug:"
#define BSP_DEFAULT_IO_CHANNEL_DEFINED
```

The debugIO driver has to be enabled:

```
#define BSPCFG_ENABLE_IODEBUG     1
```

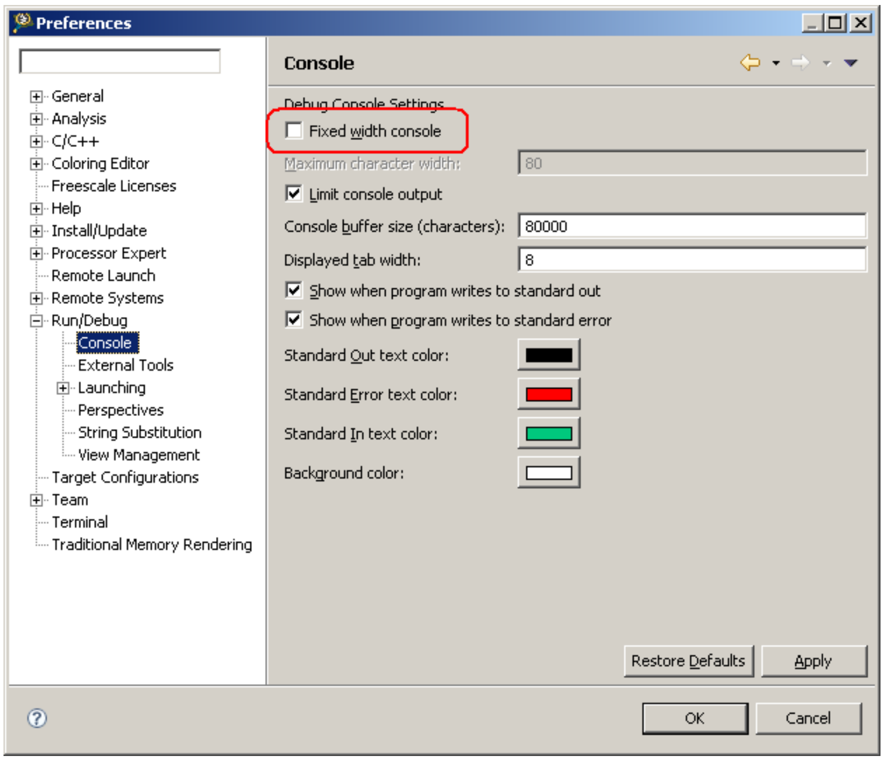
See the MQX I/O User Guide for more information on the DebugIO driver setting (ITM vs. semihost mode, buffer setting…). The tool-related setting is described in following chapters.

4.2.1 Freescale CodeWarrior Development Studio version 10

The Code Warrior 10 IDE supports semihost communication channel for output direction only (input is not supported by CW10.1). Internal DebugIO buffer should be used to speed up communication – see IO manual for detailed description.

The console can be opened from *Window/Show View/Console* CW10 menu.

Note: The *"Fixed width console"* option should be unchecked in the Run/Debug Console preferences in CW10.1.



Console

Debug Console Settings

☐ Fixed width console

Maximum character width: 80

☑ Limit console output

Console buffer size (characters): 80000

Displayed tab width: 8

☑ Show when program writes to standard out

☑ Show when program writes to standard error

Standard Out text color: ■

Standard Error text color: 🟥

Standard In text color: 🟩

Background color: □

4.2.2 IAR Embedded Workbench for ARM

The IAR EWARM IDE supports the Semihost communication channel for both input and output directions. See detailed information about setting up the debugger connection in separate document *"Getting Started with Freescale MQX™ RTOS and IAR Embedded Workbench".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/iar/MQX-IAR-Getting-Started.pdf`.

4.2.3 ARM Keil µVision4

The ARM Keil uVision4 IDE supports the ITM communication channel for both input and output directions. See detailed information about setting up the debugger connection in separate document *"Getting Started with Freescale MQX™ RTOS and MDK-ARM Keil uVision4".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/uv4/MQX-uVision4-Getting-Started.pdf`.

5 Task Aware Debugging Plug-in

MQX Task Aware Debugging plug-in (TAD) is an optional extension to a debugger tool which enables easy debugging of multi-tasking applications. It helps to visualize internal MQX data structures, task-specific information, I/O device drivers and other MQX context data.

The MQX TAD is available for the following platforms:

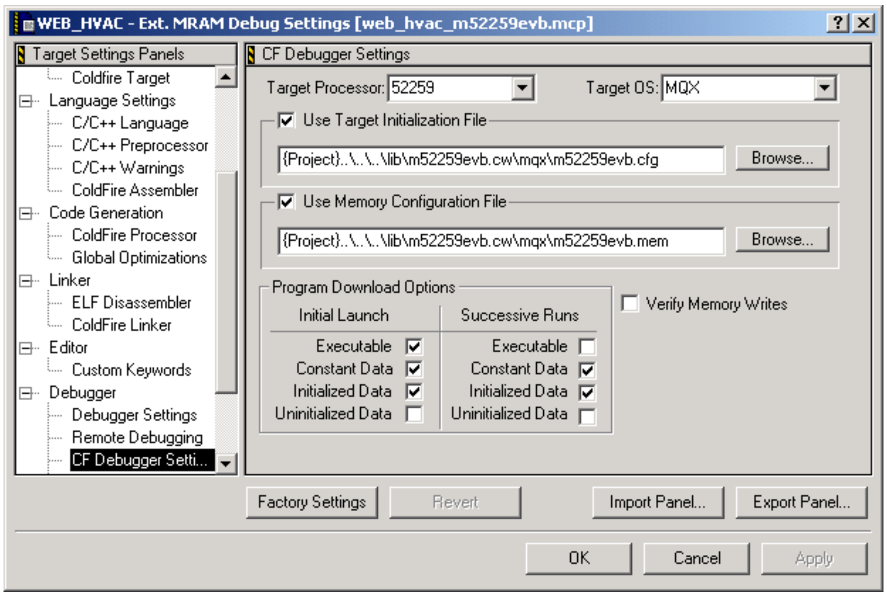
- Freescale CodeWarrior “Classic” which includes Development Studio for ColdFire version 7.x and Development Studio for PowerPC/mobileGT version 9.x.
- Eclipse-based CodeWarrior Development Studio version 10.x.
- IAR Embedded Workbench for ARM versions 5 and 6 and for ColdFire version 5

5.1 CodeWarrior Development Studio “Classic”

5.1.1 Installing CodeWarrior TAD

TAD plug-in DLL is installed into the selected CodeWarrior tool automatically during Freescale MQX™ RTOS setup process. In case plug-in was not properly installed (for example to a newly installed CodeWarrior studio, perform the following steps to install TAD manually:

- Locate the `tools\codewarrior_extensions\<compiler>` directory in Freescale MQX™ RTOS installation folder (by default `C:\Program Files\Freescale\Freescale MQX x.y`)
- Copy the entire content of `tools\codewarrior_extensions\<compiler>` directory to the CodeWarrior installation folder (e.g. `C:\Program Files\Freescale\CodeWarrior for ColdFire V7.2`)
- After the steps above are done, verify the TAD dll file exists at the new location: `<CodeWarrior>\bin\Plugins\Debugger\rtos\CFrtos_MQX.dll`
- Re-start the CodeWarrior Development Studio.
- In the CodeWarrior environment, you should be now able to enable MQX TAD by selecting *“MQX”* as *“Target OS”* in the *“CF Debugger Settings”* panel of project settings. All example applications coming with Freescale MQX™ RTOS are already configured so.



WEB_HVAC - Ext. MRAM Debug Settings [web_hvac_m52259evb.mcp]

Target Settings Panels
- Coldfire Target
- Language Settings
  - C/C++ Language
  - C/C++ Preprocessor
  - C/C++ Warnings
  - ColdFire Assembler
- Code Generation
  - ColdFire Processor
  - Global Optimizations
- Linker
  - ELF Disassembler
  - ColdFire Linker
- Editor
  - Custom Keywords
- Debugger
  - Debugger Settings
  - Remote Debugging
  - CF Debugger Setti...

CF Debugger Settings

Target Processor: 52259   Target OS: MQX

[✓] Use Target Initialization File
{Project}..\..\..\lib\m52259evb.cw\mqx\m52259evb.cfg   Browse...

[✓] Use Memory Configuration File
{Project}..\..\..\lib\m52259evb.cw\mqx\m52259evb.mem   Browse...

Program Download Options

Initial Launch		Successive Runs	
Executable	✓	Executable	
Constant Data	✓	Constant Data	✓
Initialized Data	✓	Initialized Data	✓
Uninitialized Data		Uninitialized Data	

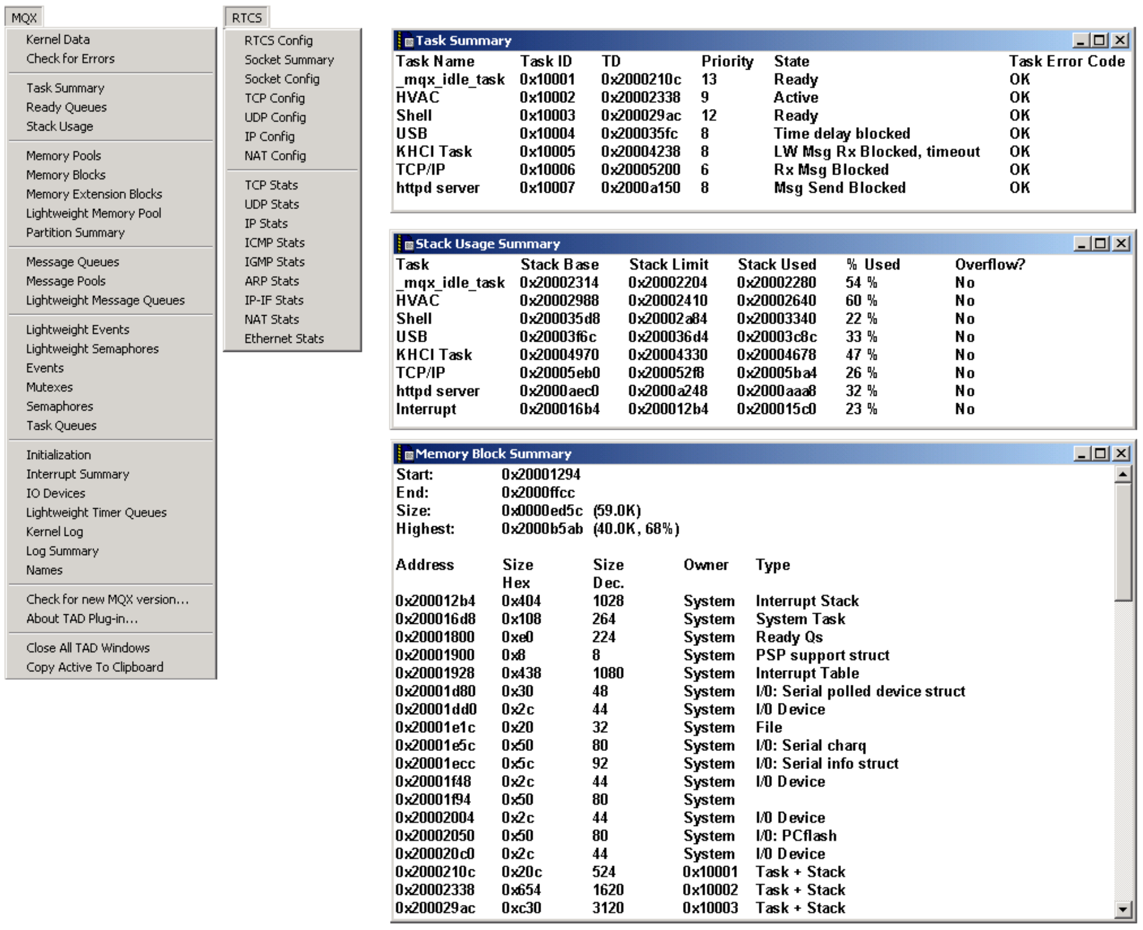
[ ] Verify Memory Writes

5.1.2 CodeWarrior TAD Features

The TAD plug-in enables more advanced and user-friendly debugging of MQX-based applications.

Using the MQX or RTCS menu in CodeWarrior IDE, several TAD “screens” may be opened during the debugging session. The most helpful and frequently used screens are shown in the picture below:

- *Task Summary* – overview about all tasks created in the MQX application.
- *Stack Usage Summary* – displays information about interrupt and task stacks. Typically, stack overflow is a root cause of vast majority of problems in MQX user applications.
- *Memory Block Summary* – displays address, size and type information about each memory block allocated in the default memory pool by the MQX system or applications. Additional memory pools (if used) may be displayed using “Memory Pools” screen.



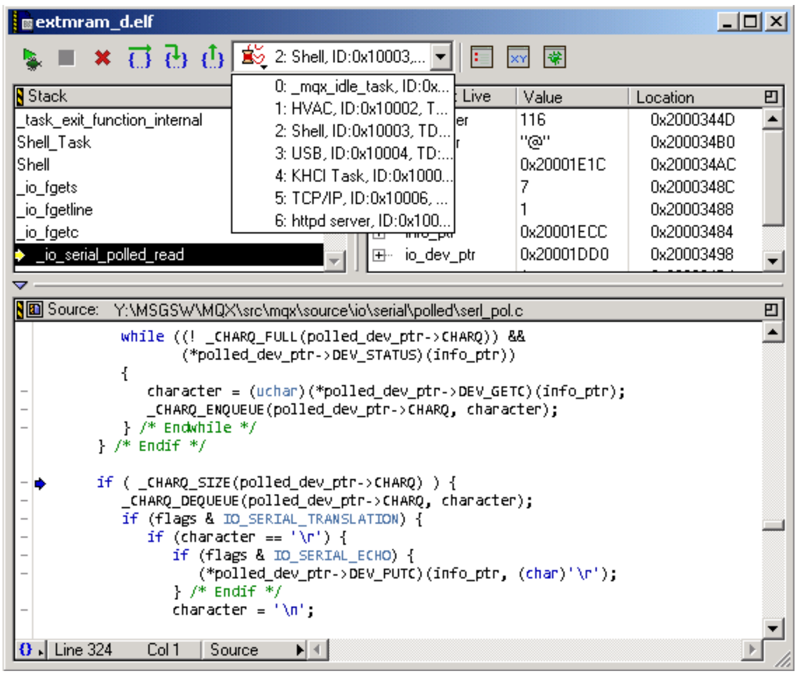
```
Task Name       Task ID   TD           Priority  State                          Task Error Code
_mqx_idle_task  0x10001   0x2000210c   13        Ready                          OK
HVAC            0x10002   0x20002338   9         Active                         OK
Shell           0x10003   0x200029ac   12        Ready                          OK
USB             0x10004   0x200035fc   8         Time delay blocked             OK
KHCI Task       0x10005   0x20004238   8         LW Msg Rx Blocked, timeout     OK
TCP/IP          0x10006   0x20005200   6         Rx Msg Blocked                 OK
httpd server    0x10007   0x2000a150   8         Msg Send Blocked               OK
```

```
Task            Stack Base   Stack Limit   Stack Used   % Used   Overflow?
_mqx_idle_task  0x20002314   0x20002204    0x20002280   54 %     No
HVAC            0x20002988   0x20002410    0x20002640   60 %     No
Shell           0x200035d8   0x20002a84    0x20003340   22 %     No
USB             0x20003f6c   0x200036d4    0x20003c8c   33 %     No
KHCI Task       0x20004970   0x20004330    0x20004678   47 %     No
TCP/IP          0x20005eb0   0x200052f8    0x20005ba4   26 %     No
httpd server    0x2000aec0   0x2000a248    0x2000aaa8   32 %     No
Interrupt       0x200016b4   0x200012b4    0x200015c0   23 %     No
```

```
Start:       0x20001294
End:         0x2000ffcc
Size:        0x0000ed5c  (59.0K)
Highest:     0x2000b5ab  (40.0K, 68%)

Address      Size     Size     Owner     Type
             Hex      Dec.
0x200012b4   0x404    1028     System    Interrupt Stack
0x200016d8   0x108    264      System    System Task
0x20001800   0xe0     224      System    Ready Qs
0x20001900   0x8      8        System    PSP support struct
0x20001928   0x438    1080     System    Interrupt Table
0x20001d80   0x30     48       System    I/0: Serial polled device struct
0x20001dd0   0x2c     44       System    I/0 Device
0x20001e1c   0x20     32       System    File
0x20001e5c   0x50     80       System    I/0: Serial charq
0x20001ecc   0x5c     92       System    I/0: Serial info struct
0x20001f48   0x2c     44       System    I/0 Device
0x20001f94   0x50     80       System
0x20002004   0x2c     44       System    I/0 Device
0x20002050   0x50     80       System    I/0: PCflash
0x200020c0   0x2c     44       System    I/0 Device
0x2000210c   0x20c    524      0x10001   Task + Stack
0x20002338   0x654    1620     0x10002   Task + Stack
0x200029ac   0xc30    3120     0x10003   Task + Stack
```

TAD also provides native debugger support for multi-tasking MQX environment. When an application is stopped at breakpoint or by pressing the (red-square) "Break" button, the name of active task is displayed in the drop-down list in the debugger window. You also have a chance to use the drop-down list to see current execution point of any other task.



extmram_d.elf

2: Shell, ID:0x10003,...

- 0: _mqx_idle_task, ID:0x...
- 1: HVAC, ID:0x10002, T...
- 2: Shell, ID:0x10003, TD...
- 3: USB, ID:0x10004, TD:...
- 4: KHCI Task, ID:0x1000...
- 5: TCP/IP, ID:0x10006, ...
- 6: httpd server, ID:0x100...

Stack:

- _task_exit_function_internal
- Shell_Task
- Shell
- _io_fgets
- _io_fgetline
- _io_fgetc
- _io_serial_polled_read

Live	Value	Location
	116	0x2000344D
	"@"	0x200034B0
	0x20001E1C	0x200034AC
	7	0x2000348C
	1	0x20003488
	0x20001ECC	0x20003484
io_dev_ptr	0x20001DD0	0x20003498

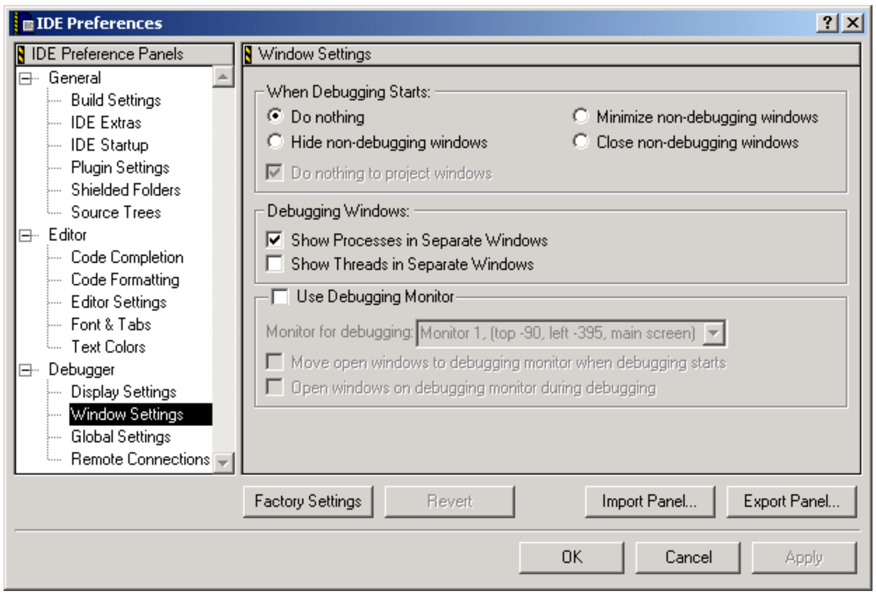
Source: Y:\MSGSW\MQX\src\mqx\source\io\serial\polled\serl_pol.c

```
            while ((! _CHARQ_FULL(polled_dev_ptr->CHARQ)) &&
                   (*polled_dev_ptr->DEV_STATUS)(info_ptr))
            {
               character = (uchar)(*polled_dev_ptr->DEV_GETC)(info_ptr);
               _CHARQ_ENQUEUE(polled_dev_ptr->CHARQ, character);
            } /* Endwhile */
         } /* Endif */

         if ( _CHARQ_SIZE(polled_dev_ptr->CHARQ) ) {
            _CHARQ_DEQUEUE(polled_dev_ptr->CHARQ, character);
            if (flags & IO_SERIAL_TRANSLATION) {
               if (character == '\r') {
                  if (flags & IO_SERIAL_ECHO) {
                     (*polled_dev_ptr->DEV_PUTC)(info_ptr, (char)'\r');
                  } /* Endif */
                  character = '\n';
```

Line 324 Col 1 Source

Note: This CodeWarrior feature needs to be enabled in general Debugger settings first. Go to the *Edit / Preferences* menu in the CodeWarrior main window and select *"Debugger / Window Settings"* panel. In the *"Debugging Windows"* section, enable the *"Show Processes in Separate Windows"* option and disable the *"Show Threads in Separate Windows"* option.



IDE Preferences

IDE Preference Panels

- General
  - Build Settings
  - IDE Extras
  - IDE Startup
  - Plugin Settings
  - Shielded Folders
  - Source Trees
- Editor
  - Code Completion
  - Code Formatting
  - Editor Settings
  - Font & Tabs
  - Text Colors
- Debugger
  - Display Settings
  - Window Settings
  - Global Settings
  - Remote Connections

Window Settings

When Debugging Starts:

- [✓ Do nothing]
- Minimize non-debugging windows
- Hide non-debugging windows
- Close non-debugging windows
- [✓ Do nothing to project windows]

Debugging Windows:

- [✓ Show Processes in Separate Windows]
- Show Threads in Separate Windows

Use Debugging Monitor

Monitor for debugging: Monitor 1, (top -90, left -395, main screen)

- Move open windows to debugging monitor when debugging starts
- Open windows on debugging monitor during debugging

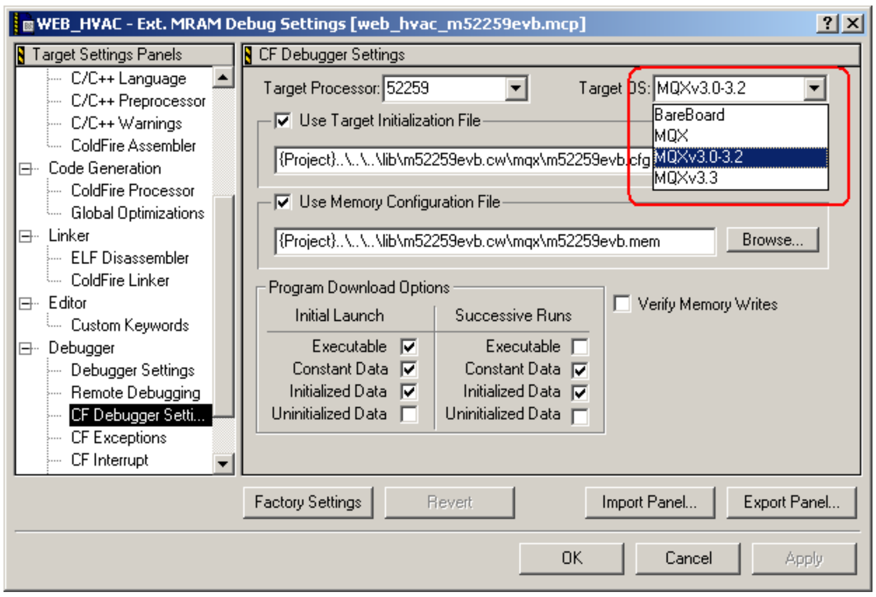
5.1.3 Backward Compatibility in TAD for CodeWarrior Classic

Each release of Freescale MQX™ RTOS brings an updated version of the TAD plugin and installs it into the CodeWarrior Development Studio selected by user. In the case that older version of MQX RTOS is installed on the same computer the older TAD plugin gets overwritten by the latest version.

Debugging different MQX versions on the same host

In some cases you may want to develop and debug different applications in different versions of the MQX. For example when a development starts and continues with MQX 3.1 while other applications are developed with MQX 3.3. This should never cause a problem because different MQX versions are completely independent one on each other. The only common resource between different versions is the latest TAD plugin installed in the CodeWarrior Studio.

TAD plugins are designed to be backward compatible, so updating to the latest MQX version should not cause any issue. In case of a problem with compatibility, old TAD plugin versions are available and you can switch to them manually. In project settings (for the selected build target), go to the *CF Debugger Settings* panel and select the TAD version from the *Target OS* drop down list.



CF Debugger Settings

Target Processor: 52259
Target OS: MQXv3.0-3.2

[✓] Use Target Initialization File
{Project}..\..\..\lib\m52259evb.cw\mqx\m52259evb.cfg

[✓] Use Memory Configuration File
{Project}..\..\..\lib\m52259evb.cw\mqx\m52259evb.mem

Program Download Options

Initial Launch		Successive Runs	
Executable	✓	Executable	
Constant Data	✓	Constant Data	✓
Initialized Data	✓	Initialized Data	✓
Uninitialized Data		Uninitialized Data	

[ ] Verify Memory Writes

5.2 CodeWarrior Development Studio version 10.x

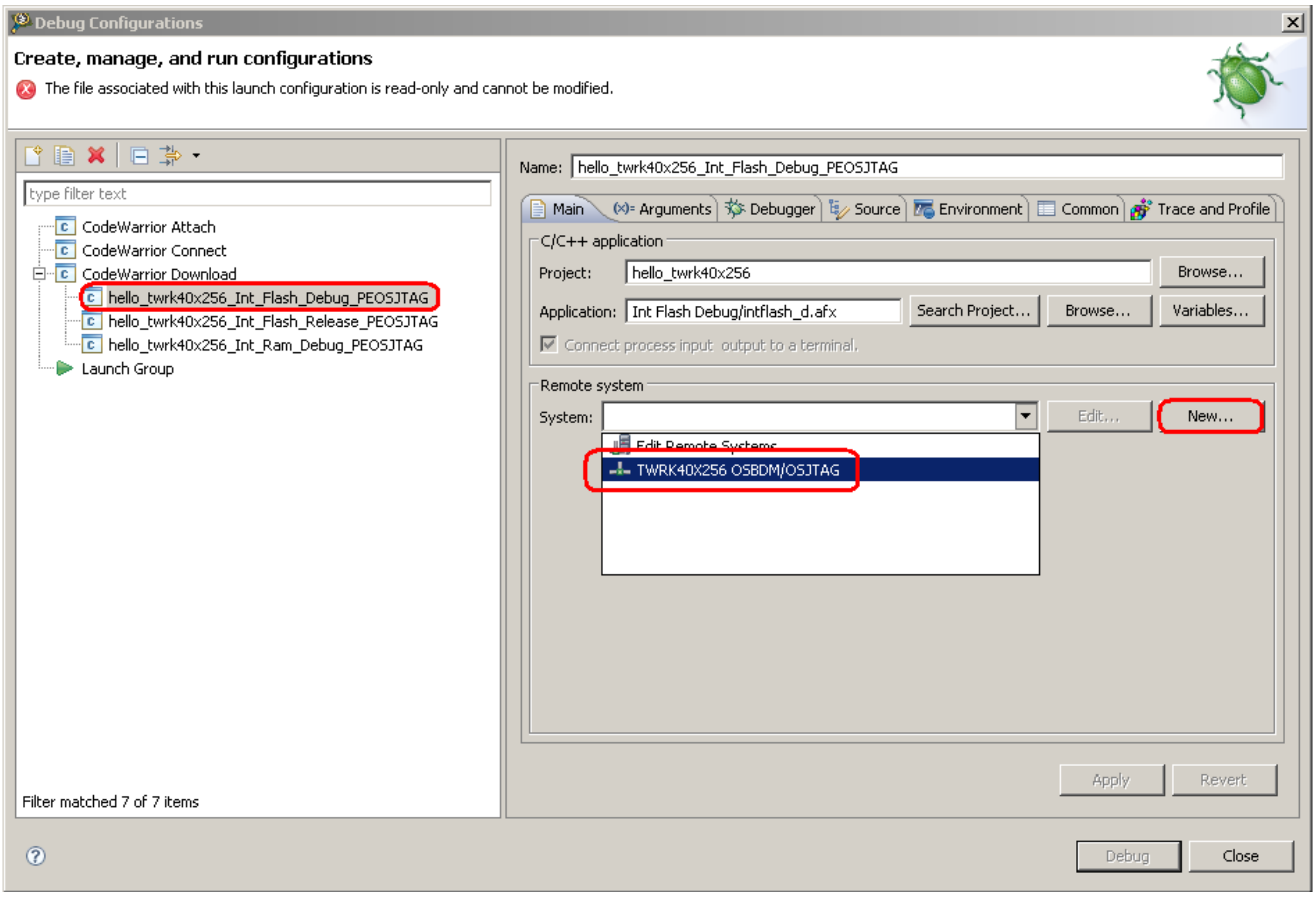
5.2.1 Debugging in CodeWarrior Development Studio version 10.x

Debugging in RAM & Flash:

The most convenient debugging of any MQX application can be done in external RAM memory. If external RAM is not available on the selected board, application code needs to be programmed into the on-chip or external Flash memory prior to debugging. This is done automatically in the CodeWarrior 10.x IDE.

Follow the steps bellow to debug an MQX application:

- Import (open) the application project in to CodeWarrior workspace using the *File / Import / General / Existing Project* menu.
- Build project using the *Project / Build Project* menu.
- Open "Debug Configurations" settings using the *Run/Debug Configurations* menu and select target you want to debug in the *CodeWarrior Download* tree.


Debug Configurations

Create, manage, and run configurations

The file associated with this launch configuration is read-only and cannot be modified.

type filter text
- CodeWarrior Attach
- CodeWarrior Connect
- CodeWarrior Download
  - hello_twrk40x256_Int_Flash_Debug_PEOSJTAG
  - hello_twrk40x256_Int_Flash_Release_PEOSJTAG
  - hello_twrk40x256_Int_Ram_Debug_PEOSJTAG
- Launch Group

Filter matched 7 of 7 items

Name: hello_twrk40x256_Int_Flash_Debug_PEOSJTAG

C/C++ application

Project: hello_twrk40x256

Application: Int Flash Debug/intflash_d.afx

Connect process input output to a terminal.

Remote system

System:

- Select connection type you want to use in the *Remote system* list. If your connection is not available in the list define new one using *New...* menu
  - Select Hardware and Simulator, Connection name and System type.

    - In System tab specify *Initialize target: as* `\lib\<board>\bsp\dbg\init_kinetis.tcl` and *Memory configuration: as* `\lib\<board>\bsp\dbg\<board>.mem`

- Press the “Debug” button in Debug configuration Window. The CodeWarrior will be switched to a debug perspective and will stop the program in the main() function.

5.2.2 CodeWarrior 10.x Task Aware Debugger plug-in

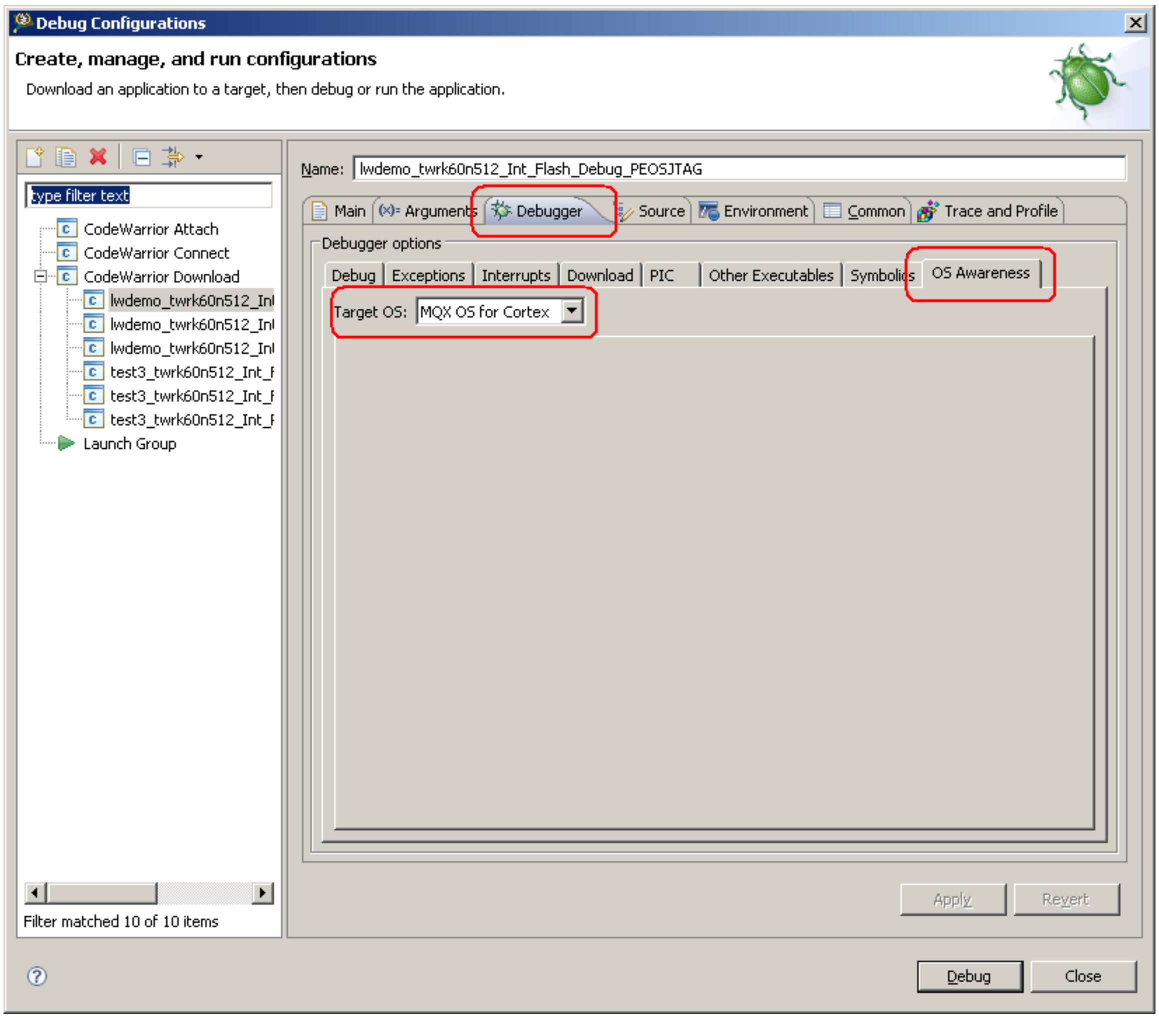
Freescale MQX™ RTOS introduces a new version of Task Aware Debugger Plug-in (TAD) for CodeWarrior 10.x Development Studio.

Installing CodeWarrior 10.x TAD and New Project Wizard Plug-in

TAD plug-in DLL is installed into the selected CodeWarrior tool automatically during Freescale MQX™ RTOS setup process. In case plug-in was not properly installed (for example to a newly installed CodeWarrior studio, perform the following steps to install TAD manually:

- Close The CodeWarrior 10.x IDE
- Locate the `tools\codewarrior_extensions\CW MCU v10.x` directory in the Freescale MQX™ RTOS installation folder (by default `C:\Program Files\Freescale\Freescale MQX` *`x.y`*)
- Navigate to `<MQX install dir>\tools\codewarrior_extensions\CW MCU v10.x` directory
- Open the command like console and execute the command:
  `install_cw10_plugin.bat <CW10.x install dir>`

  Note that the typical CodeWarrior 10.x installation folder is `C:\Program Files\Freescale\CW MCU v10.x.`
- Re-start the CodeWarrior 10.x IDE.
- Open “Debug Configurations” settings of your application project by selecting the *Run / Debug Configurations* menu. In the Debugger Configuration panel, select proper Launch Configuration
- For selected Launch Configuration, go to the “Debugger” tab and then activate the “OS Awareness” sub-tab.
- In the “Target OS” drop-down list box, select the MQX OS for your target platform.


Debug Configurations

Create, manage, and run configurations

Download an application to a target, then debug or run the application.

Name: lwdemo_twrk60n512_Int_Flash_Debug_PEOSJTAG

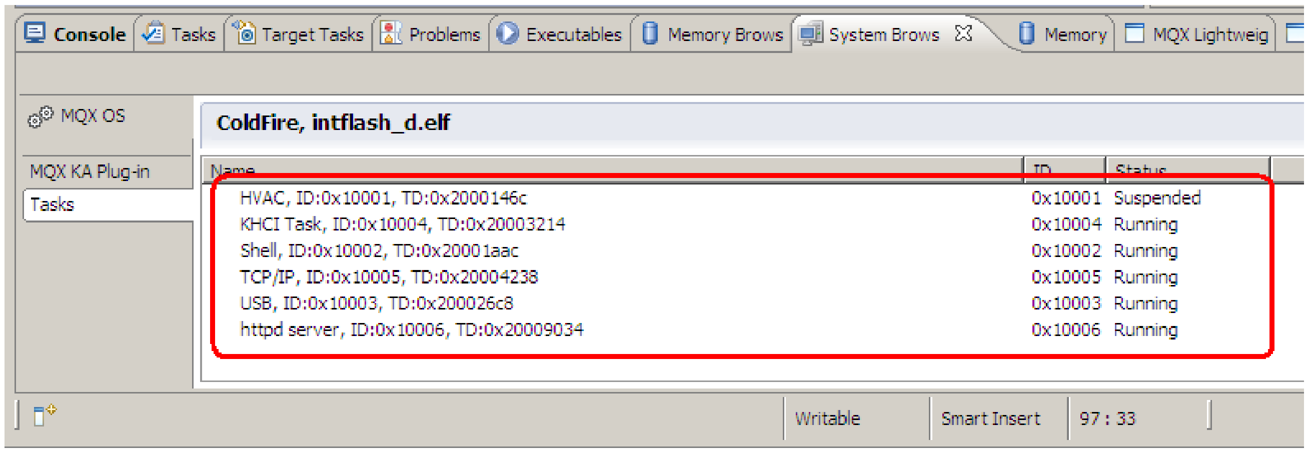
Target OS: MQX OS for Cortex

- All example applications coming with Freescale MQX™ RTOS are already configured for the MQX OS Awareness.

CodeWarrior 10.x TAD Features

The TAD plug-in for CodeWarrior 10.x provides the same set of features as the "Classic" CodeWarrior. Only the visual aspect of the TAD screens is different. In CodeWarrior 10.x, the TAD screens are displayed as Eclipse tabbed views.

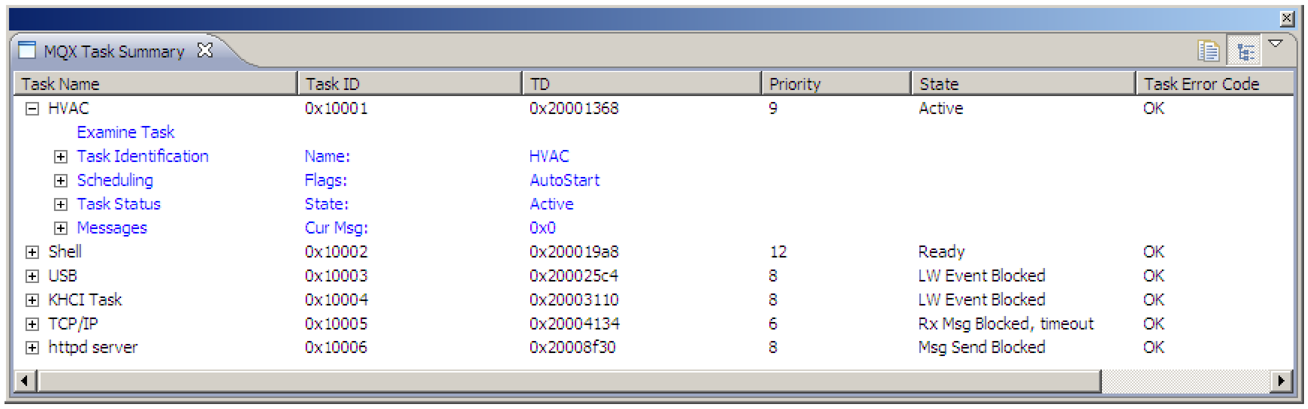
The MQX plug-in implements the *System Browser window* showing all running MQX tasks. Open the "Show View" dialog by selecting the *Window / Show View / Other…* menu. In the "Show View" tree view select the "System Browser" item in Debug tree. You can double-click any task entry in this view to activate the task in the CodeWarrior 10.x debugger.



ColdFire, intflash_d.elf

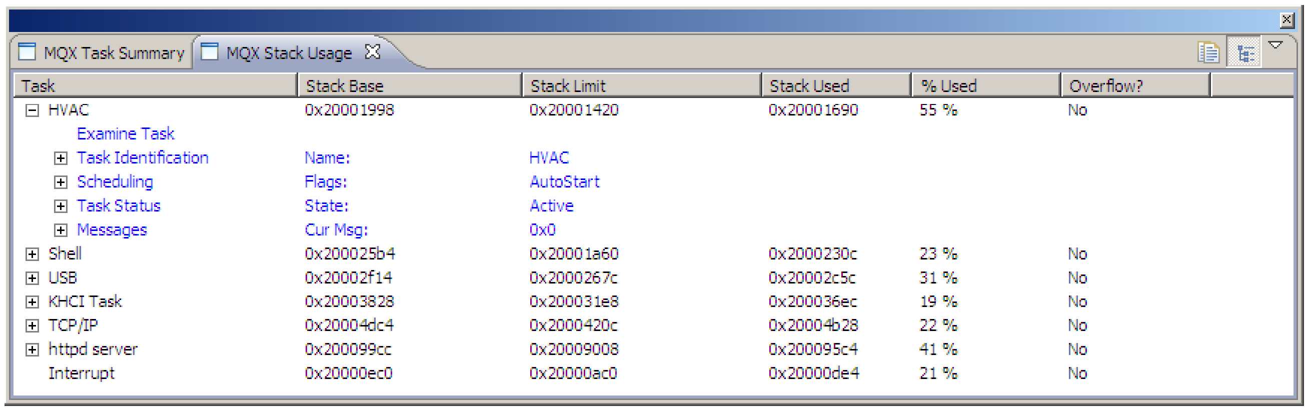
Name	ID	Status
HVAC, ID:0x10001, TD:0x2000146c	0x10001	Suspended
KHCI Task, ID:0x10004, TD:0x20003214	0x10004	Running
Shell, ID:0x10002, TD:0x20001aac	0x10002	Running
TCP/IP, ID:0x10005, TD:0x20004238	0x10005	Running
USB, ID:0x10003, TD:0x200026c8	0x10003	Running
httpd server, ID:0x10006, TD:0x20009034	0x10006	Running

The MQX Task Summary screen is available in the *MQX / Task Summary* menu



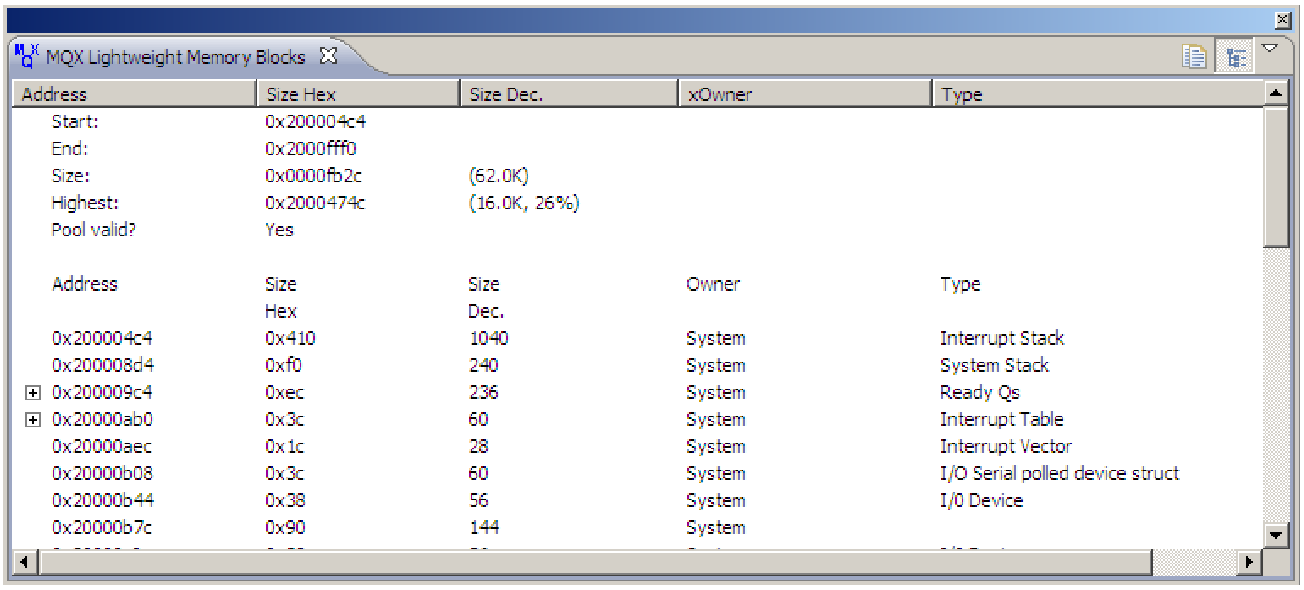
Task Name	Task ID	TD	Priority	State	Task Error Code
⊟ HVAC	0x10001	0x20001368	9	Active	OK
Examine Task					
⊞ Task Identification	Name:	HVAC			
⊞ Scheduling	Flags:	AutoStart			
⊞ Task Status	State:	Active			
⊞ Messages	Cur Msg:	0x0			
⊞ Shell	0x10002	0x200019a8	12	Ready	OK
⊞ USB	0x10003	0x200025c4	8	LW Event Blocked	OK
⊞ KHCI Task	0x10004	0x20003110	8	LW Event Blocked	OK
⊞ TCP/IP	0x10005	0x20004134	6	Rx Msg Blocked, timeout	OK
⊞ httpd server	0x10006	0x20008f30	8	Msg Send Blocked	OK

The MQX Stack Usage screen is available in the *MQX / Stack Usage* menu.



Task	Stack Base	Stack Limit	Stack Used	% Used	Overflow?
⊟ HVAC	0x20001998	0x20001420	0x20001690	55 %	No
Examine Task					
⊞ Task Identification	Name:	HVAC			
⊞ Scheduling	Flags:	AutoStart			
⊞ Task Status	State:	Active			
⊞ Messages	Cur Msg:	0x0			
⊞ Shell	0x200025b4	0x20001a60	0x2000230c	23 %	No
⊞ USB	0x20002f14	0x2000267c	0x20002c5c	31 %	No
⊞ KHCI Task	0x20003828	0x200031e8	0x200036ec	19 %	No
⊞ TCP/IP	0x20004dc4	0x2000420c	0x20004b28	22 %	No
⊞ httpd server	0x200099cc	0x20009008	0x200095c4	41 %	No
Interrupt	0x20000ec0	0x20000ac0	0x20000de4	21 %	No

The MQX Memory Block Summary screen is available in the *MQX / Lightweight Memory Blocks* menu.



Address	Size Hex	Size Dec.	xOwner	Type
Start:	0x200004c4			
End:	0x2000fff0			
Size:	0x0000fb2c	(62.0K)		
Highest:	0x2000474c	(16.0K, 26%)		
Pool valid?	Yes			
Address	Size Hex	Size Dec.	Owner	Type
0x200004c4	0x410	1040	System	Interrupt Stack
0x200008d4	0xf0	240	System	System Stack
⊞ 0x200009c4	0xec	236	System	Ready Qs
⊞ 0x20000ab0	0x3c	60	System	Interrupt Table
0x20000aec	0x1c	28	System	Interrupt Vector
0x20000b08	0x3c	60	System	I/O Serial polled device struct
0x20000b44	0x38	56	System	I/0 Device
0x20000b7c	0x90	144	System	

5.3 IAR Embedded Workbench for ARM and ColdFire

TAD is currently available for the following IAR Embedded Workbench CSpy Debugger versions:

- IAR EWARM version 6.30.x (Kinetis ARM Cortex® M4)
- IAR EWCF version 5.3

See detailed information about setting up the TAD in separate document *"Getting Started with Freescale MQX™ RTOS and IAR Embedded Workbench".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/iar/MQX-IAR-Getting-Started.pdf`.

5.4 ARM Keil µVision4

TAD is currently available for ARM Keil uVision4 Debugger (tested with version 4.22)

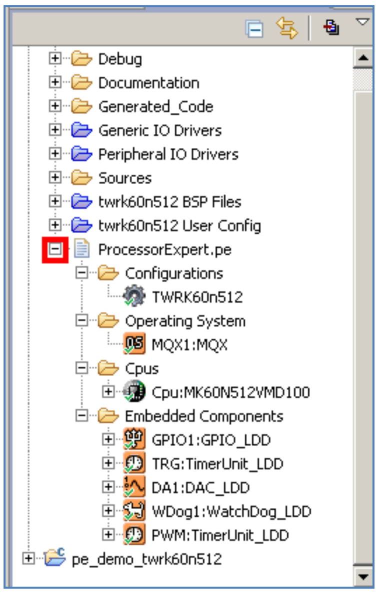
See detailed information about setting up the TAD in separate document *"Getting Started with Freescale MQX™ RTOS and MDK-ARM Keil uVision4".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/uv4/MQX-uVision4-Getting-Started.pdf`.

6 Integrating Processor Expert Drivers into MQX BSP

6.1 Introduction

The Processor Expert tool which is available in CodeWarrior 10.x allows configuration and driver code generation by using graphical user interface. The special RTOS Adapter component in Processor Expert and updated BSP code coming in MQX enables to integrate the Processor Expert drivers into the BSP library. The Processor Expert drivers can be used by the MQX application just like other drivers from the BSP. Currently, this integration is supported only for Freescale Kinetis platforms except MK70 device.

Processor Expert Logical Device Drivers (LDD) available for Kinetis family can be added into "PE ready" BSPs and used in the end user application.



For more details about Processor Expert refer to *Processor Expert User Manual* which can be found in:

```
<CW_MCU_10.x_Install_Directory>/MCU/Help/PDF/ProcessorExpertHelp.pdf
```

6.2 Processor Expert-Ready BSPs

All Kinetis BSPs (except for bsp_twrk70f120m) are modified to enable hosting of Processor Expert components.

For TWR-K40X256 board it is

```
<install_dir>/mqx/build/cw10/bsp_twrk40x256_pe/.project
<install_dir>/mqx/build/cw10/bsp_twrk40x256_pe/ProcessorExpert.pe
```

For TWR-K53N512 board it is

```
<install_dir>/mqx/build/cw10/bsp_twrk53n512_pe/.project
<install_dir>/mqx/build/cw10/bsp_twrk53n512_pe/ProcessorExpert.pe
```

For TWR-K60N512 board it is

```
<install_dir>/mqx/build/cw10/bsp_twrk60n512_pe/.project
<install_dir>/mqx/build/cw10/bsp_twrk60n512_pe/ProcessorExpert.pe
```

All preconfigured BSPs contain:

1. Pre-configured CPU component.
2. MQX RTOS Adapter component which influences the code generated by CPU component and other peripheral components.
3. Set of peripheral components (GPIO_LDD, TimerUnit_LDD, DAC_LDD and WatchDog_LDD) which are used in pe_demo application. Other components may be added to the BSP project as needed.

6.3 Processor Expert MQX Demo Application

The Demo application which demonstrates the integration of Processor Expert drivers is available here:

For TWR-K40X256 board it is

```
<install_dir>/demo/pe_demo/cw10/pe_demo_twrk40x256/.project
```

For TWR-K53N512 board it is

```
<install_dir>/demo/pe_demo/cw10/pe_demo_twrk53n512/.project
```

For TWR-K60N512 board it is

```
<install_dir>/demo/pe_demo/cw10/pe_demo_twrk60n512/.project
```

The application executes the following tasks.

1. The sine signal of specified period is generated on DAC0 pin. The signal amplitude sweeps between minimal to maximal value. It can be observed by scope on DAC0_OUT - A32 pin on TWR-ELEV FUNCTIONAL or TWR-PROTO board.
2. The PWM signal is generated using FlexTimer FTM0 Channel 0. It can be observed by scope on PWM0 - A40 pin on TWR-ELEV FUNCTIONAL or TWR-PROTO board.
3. To signalize that application is running it toggles LEDs (D9-D11) on board using GPIO driver
4. The ewm_task task is periodically refreshing watchdog within a time-out period at which watchdog can be refreshed.

See the `FSL_MQX_in_CW_10_x.pdf` document for detailed information about demo application, importing and building MQX library projects and debugging MQX application in CodeWarrior 10.x. This document is a part of the MQX installation package.

6.4 Processor Expert and Default Clock Settings in Kinetis BSPs

The MQX 3.8 introduces the low power management features on Kinetis platforms. Key building blocks of this solution are the Low Power Manager and Clock Manager modules. The Clock Manager allows runtime switching between clock configurations statically defined at the BSP level.

Three predefined clock configurations are available for Kinets-based platforms:

- 96MHz normal run mode (MGG PEE mode)
- 12MHz normal run mode (MCG PEE mode – used also for auto-trimming the internal oscillator)
- 2MHz low power run mode (MCG BLPI mode)

The code behind setting the clock configuration in the Kinetis BSPs was generated by the Processor Expert tool and it is not trivial do be changed manually (see *bsp_cm.c* and *.h* files). Use the Processor Expert to generate new or change the existing clock configurations.

Refer to separate document *"How-to Change Default Clock Settings in Kinetis BSPs".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/cw10/Howto_SetupKinetisClock_UsingPE.pdf`.

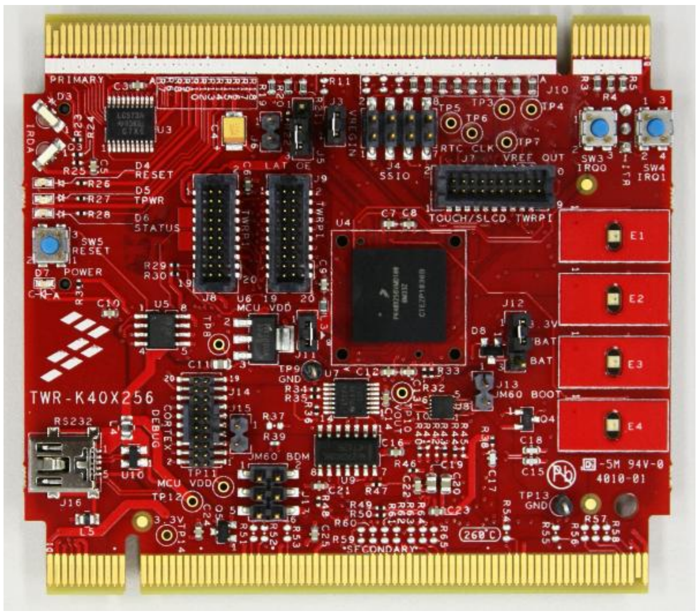
7 Board-specific Information Related to MQX

This section provides more details about all boards and BSPs supported by current MQX distribution.

All jumper and other hardware switches not specifically described below are expected in factory-default positions. Please refer to the board User's Guide for the default settings.

7.1 Kinetis

7.1.1 TWR-K40X256



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttya:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings:

- To use TWR-LCD board with eGUI
    - TWR-LCD board, SW5 all switches to ON (enable touch screen)
    - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
    - TWR-K40X256 – open J7 3-4, 5-6, 7-8
    - TWR-K40X256 – open J14 15-16
    - TWR- K40X256 board, to enable navigation buttons open J4 1-2

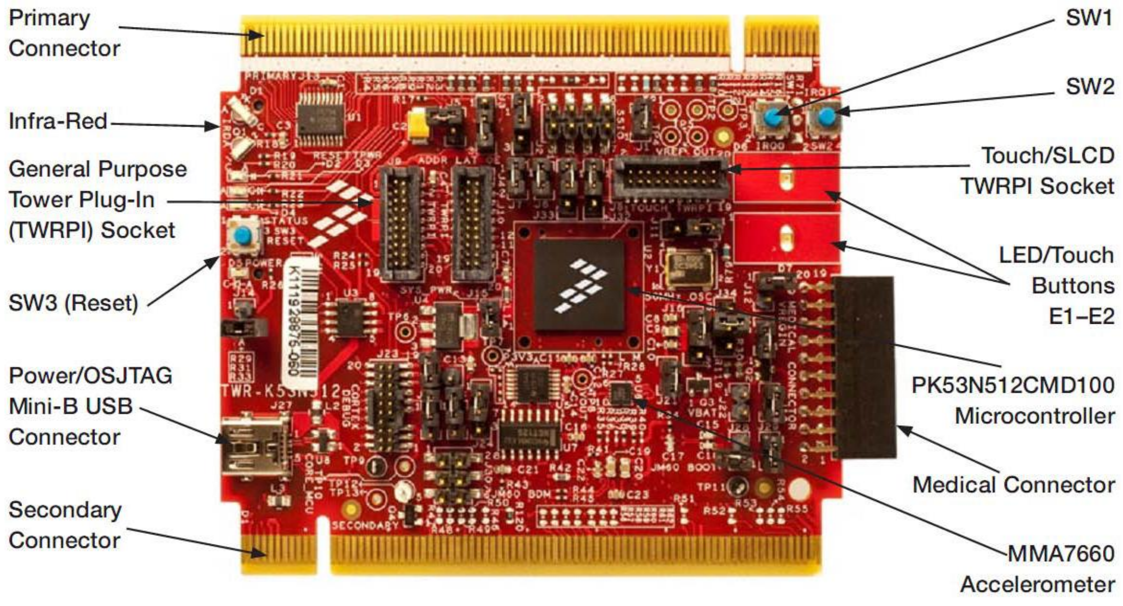
Known Issues:

- The FlexBus FB_OE_B signal is directly connected to OE pin of the address latch on the TWR-MEM card. This prevents using FlexBus for communication with MRAM and CF-CARD on TWR-MEM card.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

Other Notes:

- The default console interface (ttya:) is routed to OSBDM-COM (USB mini connector). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable the TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk40x256\twrk40x256.h` file.

7.1.2 TWR-K53N512



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttya:	OSJTAG- USB mini connector
BSP Timer	Systick	

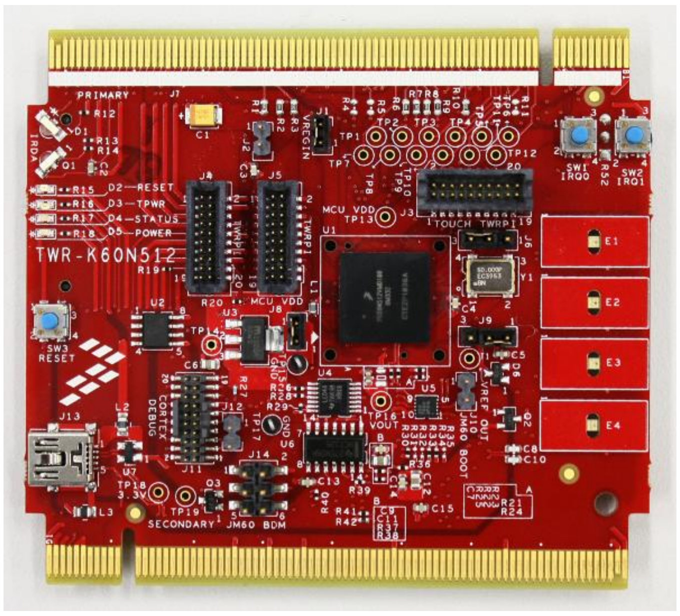
Important jumper settings (board REV C)

- For standalone operation (using clock from TWR-K53N512 board)

  - To use 50MHz, Jumper J11 on position 1-2
- To enable Ethernet communication (use with TWR-SER)
  - TWR-K53N512, Jumper 11 on position 2-3 – processor clock taken from the TWR-SER board. Jumper 32 on position 1-2, jumper 33 on position 1-2
  - TWR-SER CLK_SEL 3-4
  - TWR-SER CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER – ETH-CONFIG J12 9-10 to select RMII communication mode
  - Important: Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.

- For using USB Host mode, jumpers on position
  - TWR-SER board, J16 on position 1-2(VB_HOST)
  - TWR-SER board, J10 on default position 1-2(USB host)
- For using USB Device mode, jumpers on position
  - TWR-SER board, J16 on position 3-4(VB_DEV)
  - TWR-SER board, J10 on position 2-3(USB device)
- For using RAM disk, jumpers on position
  - TWR-MEM board, J16 remove
  - TWR-MEM board, J11 remove (default)
- For using TWRPI-SLCD:
  - TWR-K53N512, jumper 32 on position 1-2, jumper 33 on position 1-2

7.1.3 TWR-K60N512



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttyf:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings (board REV C)

- For standalone operation
  - TWR-K60N512 - Jumper J6 on position 1-2
- To enable Ethernet communication (use with TWR-SER):
  - TWR-K60N512 - Jumper J6 on position 2-3 - processor clock taken from the TWR-SER board
  - TWR-SER - CLK_SEL 3-4
  - TWR-SER - CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER - ETH-CONFIG J12 9-10 to select RMII communication mode
  - **Important:** Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.
- To use TWR-LCD board with eGUI
  - TWR-LCD board, SW5 all switches to ON (enable touch screen)
  - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
  - TWR-K60N512 – open J3 13-14

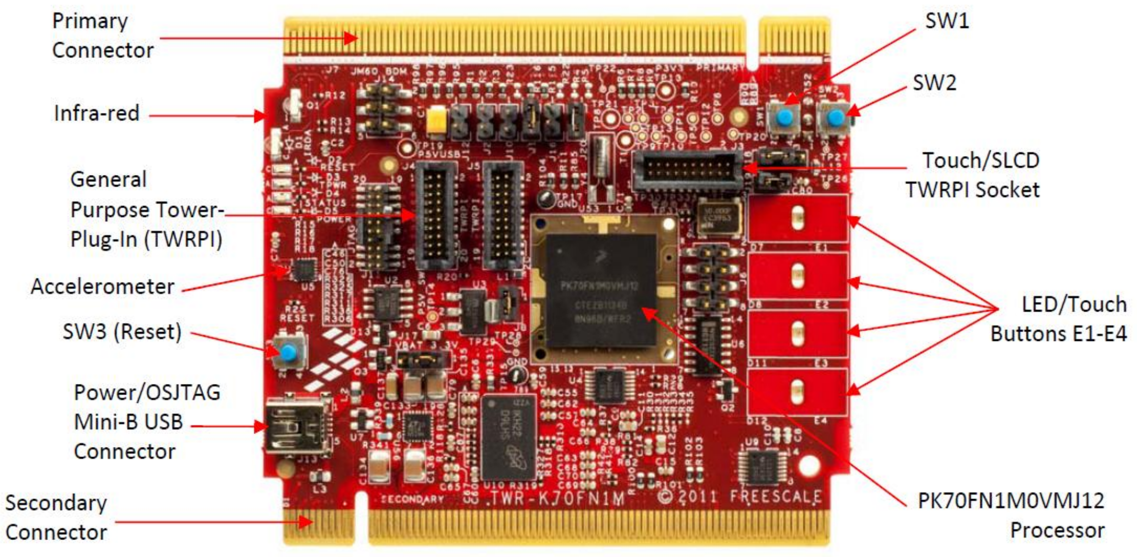
Known Issues:

- The FlexBus FB_AD9 (PTC6) signal on the TWR-K60N512 REV C board is directly connected to IRDA sensor. This prevents using FlexBus for communication with MRAM and CF-CARD on TWR-MEM card.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.
- TWR-LCD board doesn't work correctly when navigation buttons are used:
  - Usage of center navigation button on TWR-LCD board is in conflict with LCD RESET signal. Both signals are shared on main elevator A11 and A63.

Other Notes:

- The default console interface (ttyf:) is routed to OSBDM-COM (USB mini connector J13). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk60n512\twrk60n512.h` file.

7.1.4 TWR-K70F120M



Core Clock	120 MHz	
Bus Clock	60 MHz	
Default Console	ttyf:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings (board REV A)

- For standalone operation
  - TWR-K70FN1M - Jumper J18 on position 1-2
- To enable Ethernet communication (use with TWR-SER):
  - TWR-K70FN1M - Jumper J18 on position 2-3 - processor clock taken from the TWR-SER board
  - TWR-SER - CLK_SEL 3-4
  - TWR-SER - CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER - ETH-CONFIG J12 9-10 to select RMII communication mode
  - **Important:** Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.

Known Issues:

- FlexCAN1 pins PTC16 and PTC17 are shared with NAND flash memory pins on the TWR-K70FN1M REV A board. Even these pins are correctly set for the FlexCAN1 functionality CAN1_RX / CAN1_TX signals are not correctly routed to the TJA1050 High speed CAN transceiver on the TWR-SER board. This prevents correct FlexCAN example functionality.
- As RTC and CTS signals are not routed correctly from the TWR-K70FN1M REV A board to the RS485 connector of the TWR-SER board the RS485 demo application does not work correctly.

- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

Board-specific build targets:

- Int Flash SramData (Debug and Release) – this target enables to build applications that execute code from internal flash and use internal SRAM for application data.
- Int Flash DDRData (Debug and Release) – this target enables to build applications that execute code from internal flash and use external DDR2 memory for application data.

  See chapter 2.3 Build Targets for more details about standard build targets.

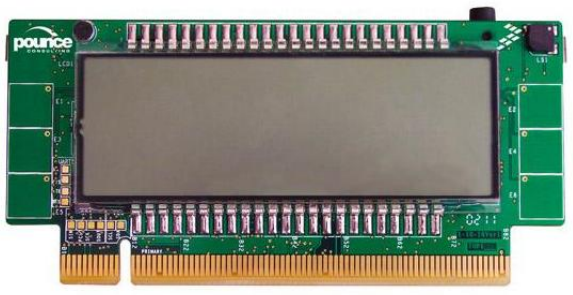
Other Notes:

- The default console interface (ttyf:) is routed to OSBDM-COM (USB mini connector J13). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyc:" in the `mqx\source\bsp\twrk70f120m\twrk70f120m.h` file.

7.1.5 KwikStik (MK40X256)

The KwikStik BSP was tested in following hardware configuration:

- KwikStik processor board (tested with Rev. 4)
- TWR-SER serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM memory extension board.(optional)



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttyf:	RS232, DB9 on TWR-SER board. The board have to be powered from USB connected to TWR-ELEV card
BSP Timer	Systick	

Important jumper settings:

- For using RAM disk, jumpers on position
  - TWR-MEM board, J16 remove
  - TWR-MEM board, J11 remove (default)Board-specific build targets:

Known issues:

- RTC clock - The VBAT pin is not being powered (R117 is DNP in the schematics) on the KwikStik Rev. 1 – 4 boards, Rev. 5 boards are solving this issue. Enabling RTC in older revision boards prevents using the card (board is automatically rebooted during startup)
- SD Card driver - The SD card connector's data pins (DATA0~DATA3) are connected to the wrong processor pins (SDHC0_D4~SDHC0_D7). The problem is present on KwikStik Rev. 1 – 4 boards, the Rev. 5 boards are solving this issue.

7.2 ColdFire V1

7.2.1 TWR-MCF51AG

Core Clock	34 MHz	
Bus Clock	17 MHz	
Default Console	ttya:	RS232
BSP Timer	TPM3	

Jumper settings (board REV C)

- SW5 (DIP_SW): 1-on, 2-on, 3-on, 4-on
- J1: on

- J2: on
- J4: 1-2, 3-4, 5-6, 7-8
- J5: on
- J6: on
- J8: off
- J9: off
- J12: 2-3
- J13: 2-3
- J14: on
- J16: off
- J17: 1-2, 3-4, 5-6, 7-8
- J18: on
- J19: on
- Use J3 (TGT_BDM) for connecting the P&E USB Multilink debugger

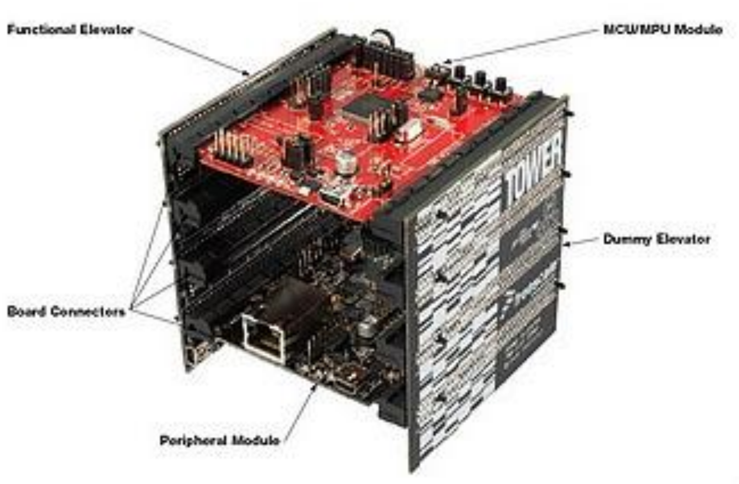
Known Issues:

- TWR-LCD board doesn't work correctly:
  - Due to missing connection from primary elevator B67 (LCD DC signal) to MCU, LCD can't receive commands and thus no data can be displayed.
  - Navigation buttons (E, S, CTR) will not work correctl
    y due to missing connection between MCU and primary elevator (A9-A11).

7.2.2 TWR-MCF51CN-KIT

TWR-MCF51CN-KIT (Rev.A) consists of

- MCF51CN128 microcontroller module board
- TWR-ELEV four-storey elevator boards
- TWR-SER serial board
- [optional] TWR-MEM memory extension board
- [optional] TWR-LCD display board



Core Clock	50 MHz	
Bus Clock	25 MHz	
Default Console	ttyb:	RS232
BSP Timer	MTIM1	

Important jumper settings:

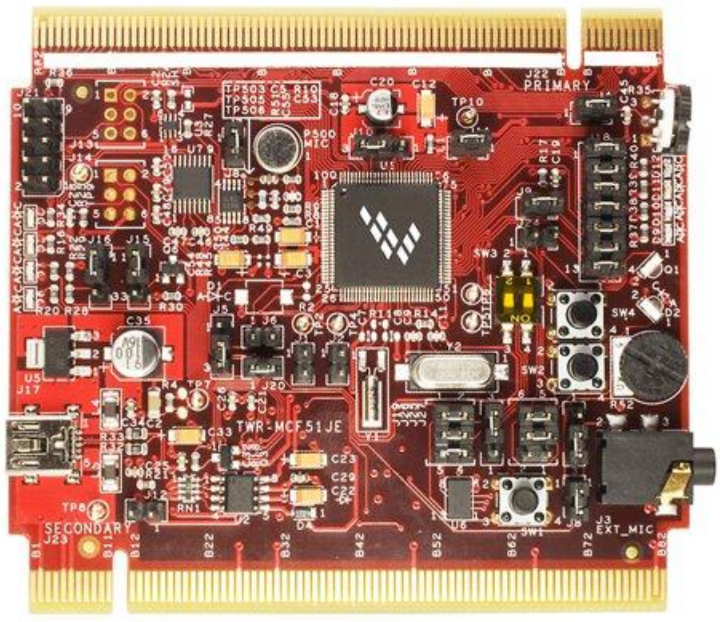
- For a basic operation, make sure the following settings is applied:
  - TWR-SER board, J2 on default position 1-2 (PHY CLK_SEL 25MHz)
  - TWR-SER board, J3 shunt removed (CLKIN_SEL)
  - TWR-SER board, J15 on default position 1-2 (SER_SEL enabling RS232 operation)
  - TWR-SER board, J17 on default position 1-2 (RXD_SEL enabling RS232 operation)
  - TWR-SER board, J18 shunt removed (RTS_SEL no RS232 flow control)
  - TWR-SER board, J19 on default position 1-2 (TXD_SEL enabling RS232 operation)
  - TWR-MCF51CN board, J3 jumpers on position 7-8 and 9-10 to connect UART to the TWR-SER board
- To use 25 MHz clock source (BSPCFG_USE_25MHZ_XTAL must be set 1)
  - TWR-MCF51CN board, J11 on position 1-2
  - TWR-MCF51CN board, J12 on position 1-2
- To use 32.768 kHz clock source (BSPCFG_USE_32KHZ_XTAL must be set 1)
  - TWR-MCF51CN board, J11 on position 3-4
  - TWR-MCF51CN board, J12 on position 2-3
- To enable ADC sensing of the potentiometer in various MQX examples (e.g. security demos)
  - TWR-MCF51CN board, J2 on position 2-3 routes ADP3 to MCU_ADP3
- To enable external MRAM memory (available on Memory Storey board)
  - TWR-MCF51CN board, J13 on position 2-3 to enable address latch

    - TWR-MEM board, J10 on position 1-2
    - TWR-MEM board, J11 shunt removed
- To enable CompactFlash Card operation (available on Memory Storey board)
    - TWR-MCF51CN board, J13 on position 2-3 to enable address latch
    - TWR-MEM board, J10 shunt removed
    - TWR-MEM board, J11 on position 1-2
    - TWR-MEM board, J16 on position 1-2
- To enable correct Ethernet duplex operation
    - TWR-SER board, J12 shunt on pins 15-16
- To enable SD Card operation
    - TWR-MCF51CN board, J5 remove three accelerometer shunts ACCX, ACCY, ACCZ
    - TWR-MCF51CN board, J3 install three SPI shunts to route SPI signals to memory board
    - TWR-MCF51CN board, J2 on position 1-2 to route SPI clock signal to memory board
    - TWR-MEM board, J3 on position 1-2 to enable SD Card CS signal
    - TWR-MEM board, J13 on position 1-2 to enable SD Card write protect signal
- To select either CS0 or CS1 for SPI Flash
    - TWR-MEM board, J14 on position 1-2 (CS0)
- To use TWR-LCD board with eGUI
    - TWR-LCD board, SW5 all switches to ON (enable touch screen)
    - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
    - TWR-MCF51CN board, J3 remove 7-8, 9-10, 11-12, 13-14 (disable UART)
- TWR-MCF51CN board, to enable navigation buttons set SW1 dip 2 OFF, J5 remove 5-6

Board-specific build targets:

See chapter 2.3 Build Targets for more details about standard build targets. There are two variants of each standard build target in the application projects: one with default debugger setting for on-board OSBDM interface, one for external P&E BDM interface.

7.2.3 TWR-MCF51JE



Core Clock	48 MHz	16MHz Xtal used
Bus Clock	24 MHz	
Default Console	ttya:	RS232
BSP Timer	TPM1	

Jumper settings (board REV C)

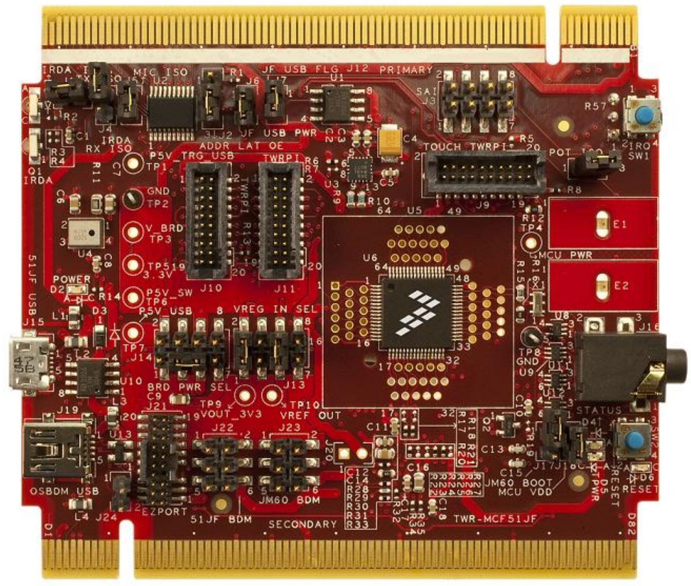
- SW3: 1-off, 2-off
- J1: off
- J2: off
- J4: on
- J5: 1-2
- J6: off
- J7: on
- J8: on
- J9: on
- J10: 2-3
- J11: on
- J12: off
- J15: 1-2
- J16: 1-2

- J18: 1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 13-14
- J19: 1-2, 3-4, 5-6
- J20: off
- J24: off
- J25: 5-6
- J26: 2-3
- J27: 1-2
- To use TWR-LCD board with eGUI
    - TWR-LCD board, SW5 all switches to on (enable touch screen)
    - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
    - TWR-MCF51JE board, J9 open (IRDA off)
    - TWR-MCF51JE board, J18 remove 9-10, 11-12, 13-14 (accelerometer off)
    - TWR-MCF51JE board, to enable navigation buttons set:
        - J4 open (potentiometer OFF),
        - J7 open (audio-in disabled),
        - J25 open 1-2

7.2.4 TWR-MCF51JF128-KIT

The MCF51JF128 BSP was tested with following hardware configuration:

- TWR-MCF51JF128 Rev. A processor board
- TWR-SER Rev. C serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM Rev. B memory extension board. [optional]



Core Clock	48 MHz	
Bus Clock	24 MHz	
Default Console	ttya:	RS232 on TWR-SER board
BSP Timer	MTIM1	

Important jumper settings:

For basic operations, make sure following jumper settings are applied:

- For using CHIP in normal mode
  - TWR-MCF51JF board, J17 no Shunt (Disable Bootload)
- For using CRC module
  - TWR-SER board, J3 on position 2-3 (external clock 50MHz)
  - TWR-MCF51JF board, J2 on default position 1-2 (external clock 50MHz)
- For using USB MICRO AB5 port for USB DCD module, connecting by supported cable
  - TWR-MCF51JF board, J13 on default position 1-2(P5V_TRG_USB)
  - TWR-MCF51JF board, J13 on default position 5-6(P5V_JF_USB)
- For using USB Device Charger Detection, jumpers on position
  - TWR-MCF51JF board, J8 removed
  - Pin1 on J8 connect to position 5-6 on J13(PTD5 connect to JF_VREGIN)
- For using ADC, connect to POTENTIOMETER, jumpers on position
  - TWR-MCF51JF board, J8 on position 1-2(ADC0_SE12)
- For using USB host or device, jumpers on position
  - TWR-MCF51JF board, J6 on position 1-2(JF_USB_ENA)
  - TWR-MCF51JF board, J7 on position 1-2(JF_USB_FLGA)
- For using USB Host mode, jumpers on position
  - TWR-SER board, J16 on position 1-2(VB_HOST)
  - TWR-SER board, J10 on default position 1-2(USB host)
- For using USB Device mode, jumpers on position
  - TWR-SER board, J16 on position 3-4(VB_DEV)
  - TWR-SER board, J10 on position 2-3(USB device)
- For using RAM disk, jumpers on position
  - TWR-MEM board, J16 remove
  - TWR-MEM board, J11 remove (default)
- For using USBDCD jumpers on position
  - Remove J8 jumper and short between pin 1 of J8 and pin 5 of J13 on TWR-MCF51JF board.
- For using SD Card

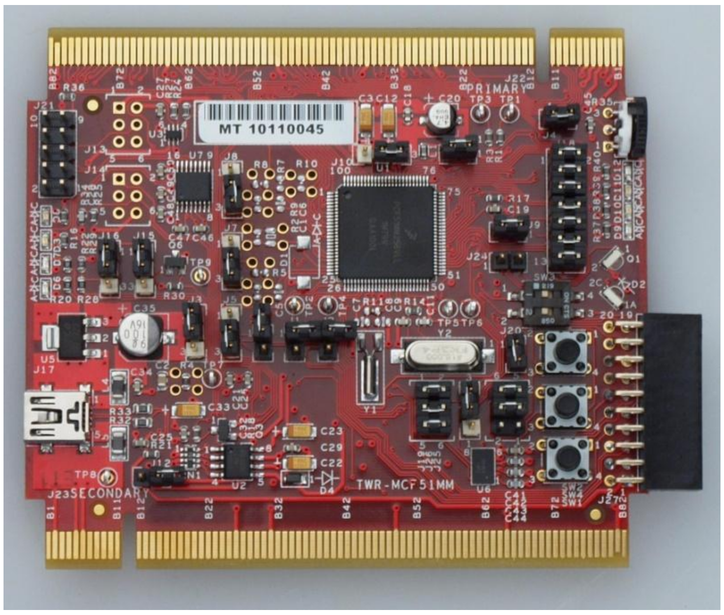
- TWR-MEM board, J3 (SD_CS) jumper on position 1-2 to enable SD Card CS signal
- TWR-MEM board, J12 (SD_SEL1) remove jumper from 1-2 and insert jumper on 3-4

Board-specific build targets:

- Internal Flash (Debug and Release) - these targets enable to build applications suitable for booting the system up from Internal Flash memory. After the reset the code will be executed from Internal Flash
- See chapter 2.3 Build Targets for more details about standard build targets.

7.2.5 TWR-MCF51MM-KIT

All jumper and other hardware switches not specifically described below are expected in factory-default positions. Please refer to the board User's Guide for the default settings.



Core Clock	48 MHz	16MHz Xtal used
Bus Clock	24 MHz	
Default Console	ttya:	RS232
BSP Timer	TPM1	

The MCF51MM BSP was tested in the following configuration:

- TWR-MCF51MM Rev. B processor board
- TWR-SER Rev. B serial board
- TWR-ELEV Rev. A four-storey elevator boards
- TWR-MEM Rev. B memory extension board

Important jumper settings:

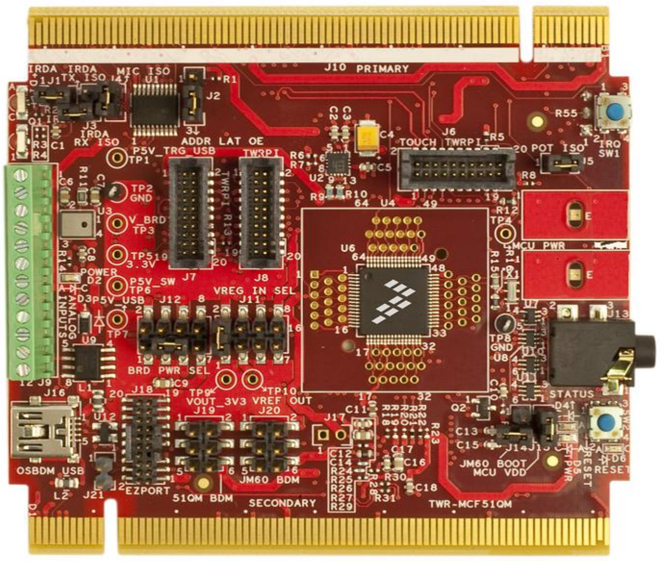
- For a basic operation, make sure the following settings is applied:
  - TWR-MCF51MM board, J3 jumper on 2-3 (connects VINP0 to GND)
  - TWR-MCF51MM board, J4 shorted (POT R35 to ADP4)
  - TWR-MCF51MM board, J5 jumper on 2-3 (DACO to DACO_TEST TP7)
  - TWR-MCF51MM board, J6 opened
  - TWR-MCF51MM board, J7, J8 jumper on 2-3
  - TWR-MCF51MM board, J9 jumper on 1-2 (IR circuit)
  - TWR-MCF51MM board, J10 jumper on 2-3
  - TWR-MCF51MM board, J11 jumper on 1-2
  - TWR-MCF51MM board, J12 opened
  - TWR-MCF51MM board, J14 opened for SD card operation, jumper installed for potentiometer
  - TWR-MCF51MM board, J15, J16 both shorted 1-2
  - TWR-MCF51MM board, J18 all shorted
  - TWR-MCF51MM board, J19 jumper (1-2, 3-4, 5-6)
  - TWR-MCF51MM board, J20 jumper 1-2
  - TWR-MCF51MM board, J24 opened
  - TWR-MCF51MM board, J25 jumper (1-2, 3-4)
  - TWR-MCF51MM board, J26 jumper (2-3)
- To enable USB operation in HOST mode
  - TWR-MCF51MM, J3 1-2 not connected (jumper removed)
  - TWR-SER board, J16 jumper on pins 1-2
  - TWR-SER board, J10 jumper on pins 1-2
- To enable USB operation in DEVICE mode
  - TWR-SER board, J16 jumper on pins 3-4
  - TWR-SER board, J10 jumper on pins 2-3
- To enable SD Card operation
  - TWR-MCF51MM board, J4 opened
  - TWR-MEM board, J3 (SD_CS) jumper on position 1-2 to enable SD Card CS signal
  - TWR-MEM board, J12 (SD_SEL1) remove jumper from 1-2 and insert jumper on 3-4
- To enable CF Card operation
  - TWR-MEM board, J16 on position 2-3
- To enable external MRAM memory (available on Memory Storey board)
  - TWR-MEM board, J10 on position 1-2
  - TWR-MEM board, J16 on position 2-3
- To use TWR-LCD board with eGUI

- TWR-LCD board, SW5 all switches to ON (enable touch screen)
- TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
- TWR-MCF51MM board, J9 open (IRDA OFF)
- TWR-MCF51MM board, J18 remove 9-10, 11-12, 13-14 (accelerometer off)
- TWR-MCF51MM board, to enable navigation buttons set:
    - J4 open (potentiometer OFF)
    - J19 open (disable accelerometer self test)

7.2.6 TWR-MCF51QM128-KIT

The MCF51QM128 BSP was tested with following hardware configuration:

- TWR-MCF51QM128 Rev. A processor board
- TWR-SER Rev. C serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM Rev. B memory extension board.



Core Clock	48 MHz	
Bus Clock	24 MHz	
Default Console	ttya:	RS232 on TWR-SER board
BSP Timer	MTIM1	

Important jumper settings:

For basic operations, make sure following jumper settings are applied:

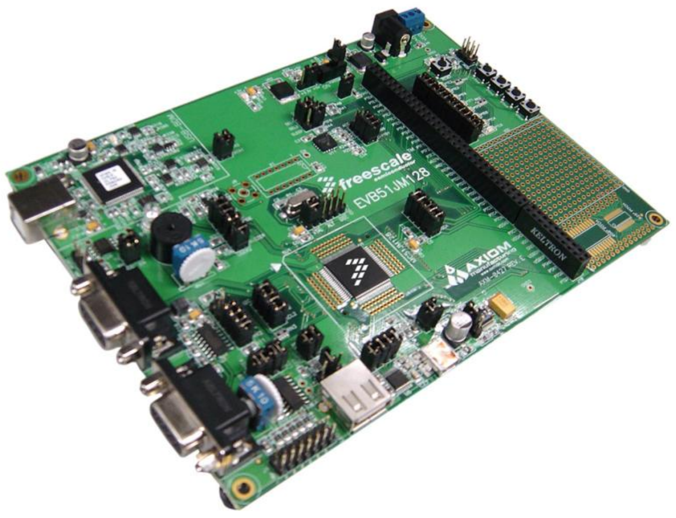
- For using CHIP in normal mode

- TWR-MCF51QM board, J14 no Shunt (Disable Bootload)
- For using CRC module
  - TWR-SER board, J3 on position 2-3 (external clock 50MHz)
  - TWR-MCF51QM board, J2 on default position 1-2 (external clock 50MHz)
- For using ADC, connect to POTENTIOMETER, jumpers on position
  - TWR-MCF51QM board, J5 on position 1-2(ADC0_SE12)
- For using RAM disk, jumpers on position
  - TWR-MEM board, J13 remove
  - TWR-MEM board, J8 remove (default)

Board-specific build targets:

- Internal Flash (Debug and Release) - these targets enable to build applications suitable for booting the system up from Internal Flash memory. After the reset the code will be executed from Internal Flash
- See chapter 2.3 Build Targets for more details about standard build targets.

7.2.7 EVB51JM128 Board



Core Clock	48 MHz	
Bus Clock	24 MHz	
Default Console	ttyb:	RS232
BSP Timer	TPM1	

Important jumper settings (board rev. E):

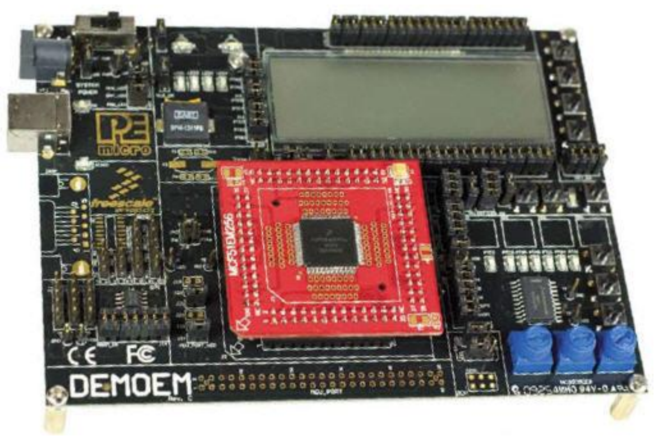
- For USB Host operation

- VBSEL 1-2 H
- HOST_EN both shunts on
- OTG_EN all shunts off
- USPD shunt removed

Board-specific build targets:

See chapter 2.3 Build Targets for more details about standard build targets. The build targets in application projects are configured properly for on-board USB P&E BDM interface.

7.2.8 DEMOEM Board



Core Clock	50.332 MHz	32.768 kHz low-frequency crystal Y1 used
Bus Clock	25.166 MHz	
Default Console	ttya:	P&E USB
BSP Timer	MTIM1	

Important jumper settings:

No change to default factory settings is needed.

Board-specific build targets:

See chapter 2.3 Build Targets Notes for more details about standard build targets. The build targets in application projects are configured properly for on-board USB P&E BDM interface.

Other notes:

By default, the UART serial interface of the MCF51EM processor is connected through a virtual communication channel implemented in on-board USB P&E BDM interface. You need to use a *Terminal Window Utility* on the host PC to get access to console and all shell-based applications. The utility is available on the DVD-ROM accompanying the DEMOEM board.

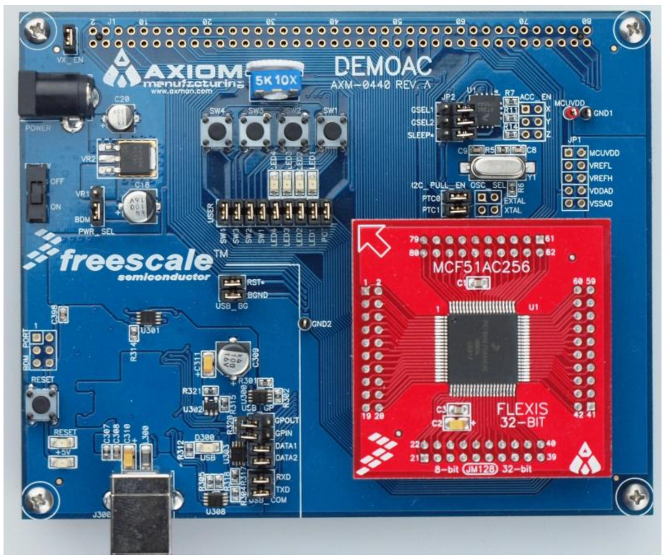
You can choose to use the UART port as a standard RS232 interface. The advantage of this approach is that you can use serial interface during application debugging. You may need to populate the following components:

- DB9F - DB9 Female Right Angle .318 D-Sub Connector
- C12,C13,C14,C15,C16 - 0.1uf Capacitor
- U2 - MAX3232CSE+

In order to run MQX SPI example you also need to populate SPI Flash or MRAM memory:

- U4 – for example M95512-RMN6P

7.2.9 DEMOAC Board with MCF51AC256



Core Clock	40 MHz	
Bus Clock	20 MHz	
Default Console	ttya:	P&E USB
BSP Timer	TPM1	

Important jumper settings:

No change to default factory settings is needed.

Board-specific build targets:

See chapter 2.3 Build Targets for more details about standard build targets. The build targets in application projects are configured properly for on-board USB P&E BDM interface.

7.3 ColdFire V2

7.3.1 TWR-MCF52259-KIT

TWR-MCF52259-KIT (Rev.A) consists of

- MCF52259 microcontroller module board
- TWR-ELEV four-storey elevator boards
- TWR-SER serial board
- [optional] TWR-MEM memory extension board
- [optional] TWR-LCD display board

Core Clock	80 MHz	
Bus Clock	40 MHz	
Default Console	ttyb:	RS232
BSP Timer	PIT0	

Important jumper settings:

- For a basic operation, make sure the following settings is applied:
  - TWR-SER board, J2 on default position 1-2 (PHY CLK_SEL 25MHz)
  - TWR-SER board, J3 shunt removed (CLKIN_SEL)
  - TWR-SER board, J15 on default position 1-2 (SER_SEL enabling RS232 operation)
  - TWR-SER board, J17 on default position 1-2 (RXD_SEL enabling RS232 operation)
  - TWR-SER board, J18 shunt removed (RTS_SEL no RS232 flow control)
  - TWR-SER board, J19 on default position 1-2 (TXD_SEL enabling RS232 operation)
- To enable external MRAM memory (available on Memory Storey board)

  - TWR-MEM board, J10 on position 1-2
  - TWR-MEM board, J11 shunt removed
- To enable correct Ethernet duplex operation
  - TWR-SER board, J12 shunt on pins 15-16
- To enable USB operation in HOST mode
  - TWR-SER board, J16 shunt on pins 1-2
  - TWR-SER board, J10 shunt on pins 1-2
- To enable USB operation in DEVICE mode
  - TWR-SER board, J16 shunt on pins 3-4
  - TWR-SER board, J10 shunt on pins 2-3
- To enable SD Card operation
  - TWR-MEM Board, J3 on position 1-2 to route QSPI_PCS0 to SD Card Chip Select
  - TWR-MEM Board, J4 remove shunt on pins 1-2 to disable QSPI_PCS0 routing to serial Flash
  - TWR-MEM board, J13 on position 1-2 to enable SD Card write protect signal
- To enable CompactFlash Card operation (available on Memory Storey board)
  - TWR-MEM board, J16 on position 2-3
- To use write protect detection signals with SD Card on the Memory Storey board
  - TWR-MCF52259 board, turn off switch 3 on SW2 dip-switch.
- To select either CS0 or CS1 for SPI Flash
  - TWR-MEM board, J14 on position 1-2 (CS0)
- To use TWR-LCD board with eGUI
  - TWR-LCD board, SW5 all switches to ON (enable touch screen)
  - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
  - TWR-MCF52259 board, to enable navigation buttons set SW2 dip 2 and SW2 dip 3 to OFF

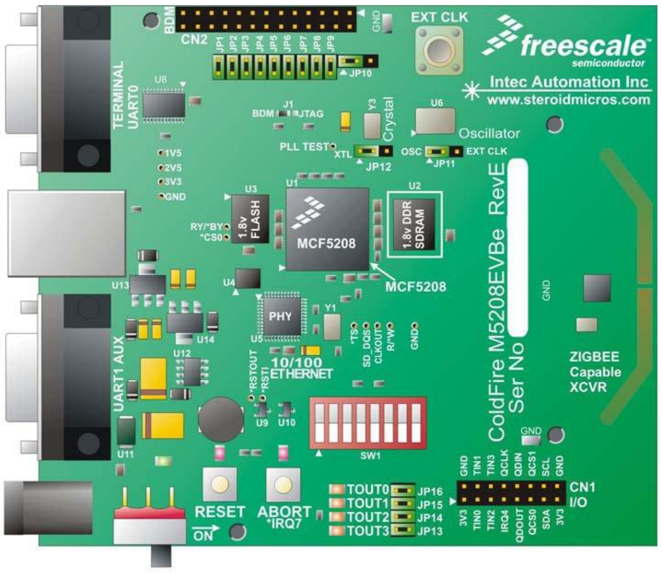
Board-specific build targets:

- None. See Freescale chapter 2.3 Build Targets for more details about standard build targets. The Ext. **MRAM Debug** target can be used only with Memory Storey Board.

Other notes:

The OSBDM Firmware compatibility issue may affect application debugging. See Freescale MQX Release Notes for more details about OSBDM Firmware Compatibility.

7.3.2 M5208EVB(E)



Core Clock	166,666 MHz	
Bus Clock	83,333 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Jumper settings (M5208EVBE board REV E)

- SW: 1- off, 2- off, 3-off, 4- off, 5- off, 6- off, 7- off, 8- off (default setting)
- JP1-on, JP2-on, JP3-off, JP4-on, JP5-on, JP6-on, JP7-on, JP8-on, JP9-on
- JP10: 1-2
- JP11: 1-2
- JP12: 1-2
- JP13-on, JP14-on, JP15-on, JP16-on

NOTE:

- The M5208EVBE board is the RoHS compliant version of the M5208EVB board. The M5208 BSP was tested with the M5208EVBE board only.

7.3.3 M52223EVB

Important jumper settings:

- None.

Board-specific build targets:

- None. See for more details about standard build targets.

7.3.4 M52233DEMO

Core Clock	60 MHz	
Bus Clock	30 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- None

Board-specific build targets:

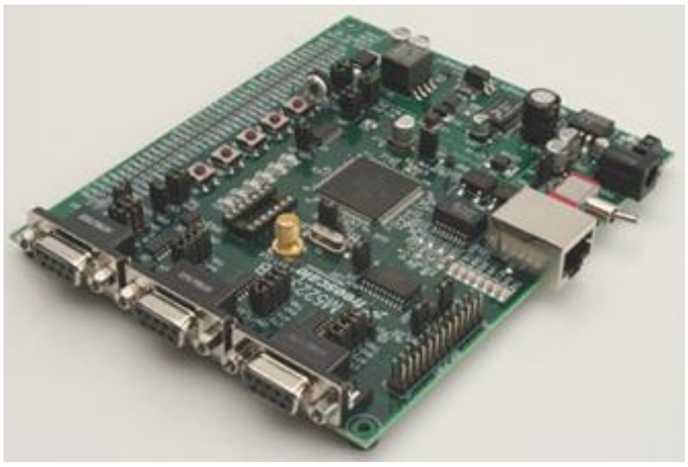
- None. See chapter *2.3 Build Targets* for more details about standard build targets.

Known issues:

- A problem of BDM communication loss was occasionally observed when debugging M52235DEMO applications. This issue is not related to MQX RTOS.

  Workaround: This issue may be solved by decreasing the BDM communication speed by factor 2 or higher in the "*Remote Debugging*" settings panel for PEMICRO_USB connection. Press the "*Edit Connection*" button and specify the *Speed* factor value "2". See the screenshot bellow in the **Error! Not a valid bookmark self-reference.** section.

7.3.5 M52235EVB



Core Clock	60 MHz	
Bus Clock	30 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- To enable CAN demo operation (disables UART2)
  - COM_SELA at position 2-3
  - COM_SELB at position 2-3
  - COM_SELC at position 2-3

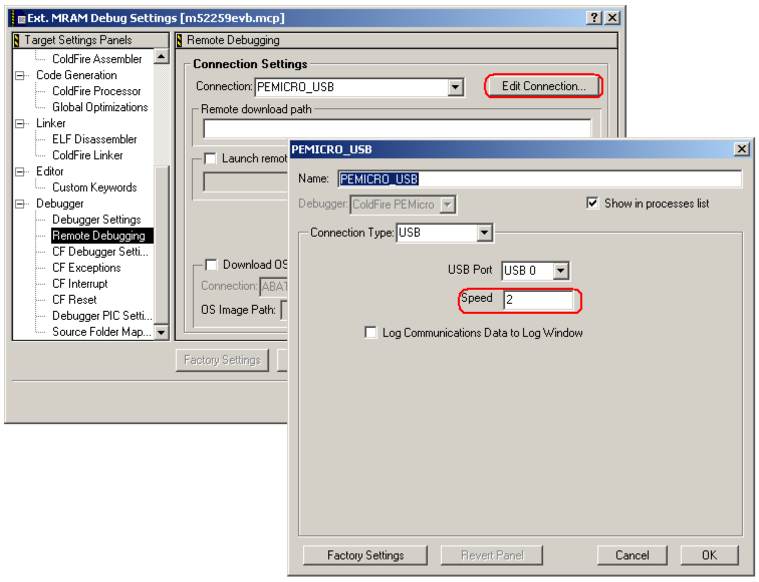
Board-specific build targets:

- None. See 2.3 Build Targets for more details about standard build targets.

Known issues:

- A problem of BDM communication loss was occasionally observed when debugging M52235EVB applications. This issue is not related to MQX RTOS.

  Workaround: This issue may be solved by decreasing the BDM communication speed by factor 2 or higher in the "*Remote Debugging*" settings panel for PEMICRO_USB connection. Press the "*Edit Connection*" button and specify the *Speed* factor value "2"


Ext. MRAM Debug Settings [m52259evb.mcp]

Target Settings Panels: ColdFire Assembler; Code Generation: ColdFire Processor, Global Optimizations; Linker: ELF Disassembler, ColdFire Linker; Editor: Custom Keywords; Debugger: Debugger Settings, Remote Debugging, CF Debugger Setti..., CF Exceptions, CF Interrupt, CF Reset, Debugger PIC Setti..., Source Folder Map...

Remote Debugging

Connection Settings

Connection: PEMICRO_USB

Remote download path

PEMICRO_USB

Name: PEMICRO_USB

Debugger: ColdFire PEMicro

Show in processes list: ✓

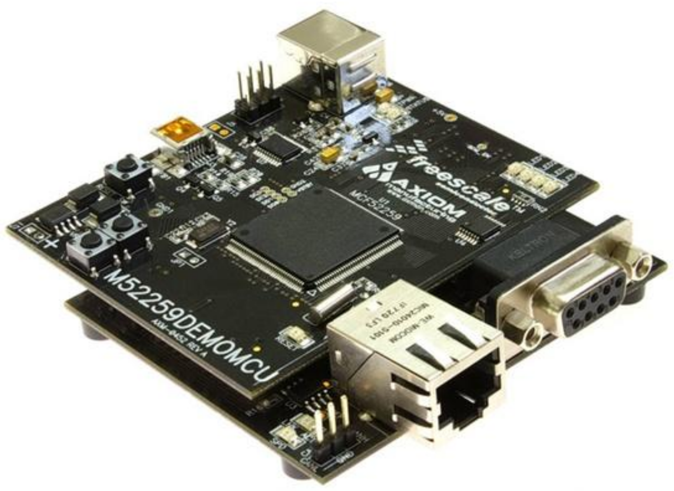
Connection Type: USB

USB Port: USB 0

Speed: 2

Log Communications Data to Log Window: ☐

7.3.6 M52259DEMO



Core Clock	80 MHz	
Bus Clock	40 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- None.

Other notes:

The FEC_MDC pin is shared with GPIO signal used to sense the SW1 button state. The Ethernet link status monitoring is not functional in demos which use SW1 button (all HVAC demos).

The OSBDM Firmware compatibility issue may affect application debugging. See Freescale MQX Release Notes for more details about OSBDM Firmware Compatibility.

Board-specific build targets:

- None. See chapter 2.3 Build Targets for more details about standard build targets.

7.3.7 M52259EVB

Core Clock	80 MHz	
Bus Clock	40 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- To enable MDIO/MDC communication between processor and Ethernet PHY device (needed in RTCS applications to detect Ethernet link status)
    - J9 at position 2-3 (FEC_MDC)
    - J10 at position 2-3 (FEC_MDIO)
- To enable RTC operation from external crystal
    - H2 at position 1-2
- To enable RTC sourced from battery
    - J17 at position 1-2

Board-specific build targets:

None. See chapter 2.3 Build Targets for more details about standard build targets.

Other information:

Firmware source code for Altera CPLD is available in
<MQX install dir>\mqx\source\io\pccard\m52259evb_pccard_cpld directory.

7.3.8 M52277EVB

Core Clock	136 MHz	
Bus Clock	68 MHz	
Default Console	ttya:	USB port J26 (UART0)
BSP Timer	PIT0	

Important jumper settings (board rev B, schematic D1):

- For USB operation
    - **J7** shunt on position 3-4 (VBUSON)
    - **J9** shunt on position 1-2 (USB_VBUC_OC)

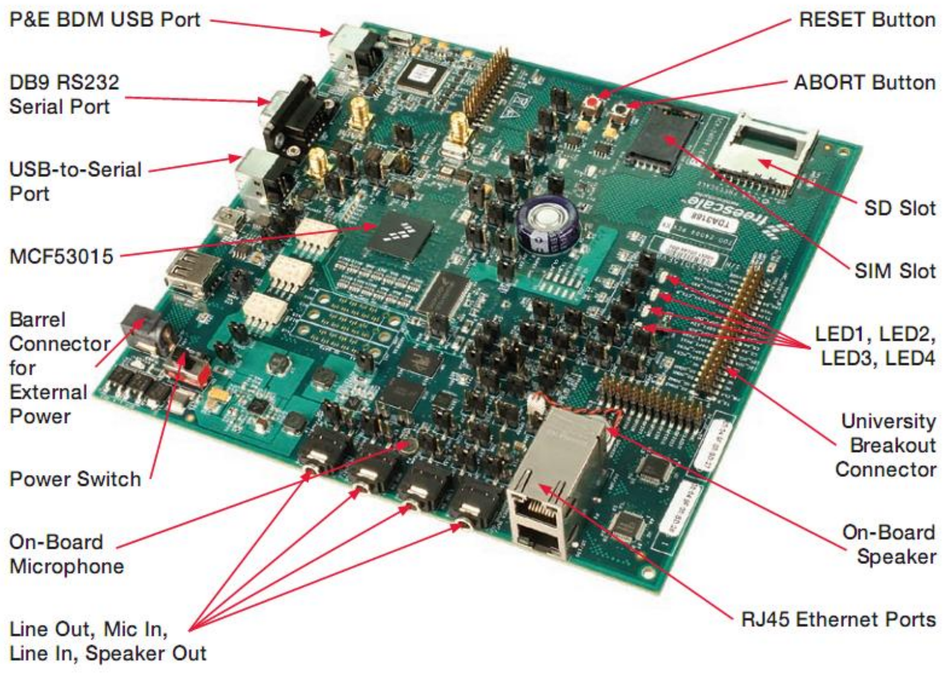
Board-specific build targets:

- Ext Flash (Debug and Release) – This target enables a standalone operation from on-board flash memory. External SDRAM memory is used for variables by default. The linker command file can be changed easily to allocate variables in the internal SRAM memory.

See chapter 2.3 Build Targets for more details about standard build targets.

7.4 ColdFire V3

7.4.1 M53015EVB



Core Clock	240 MHz	
Bus Clock	80 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

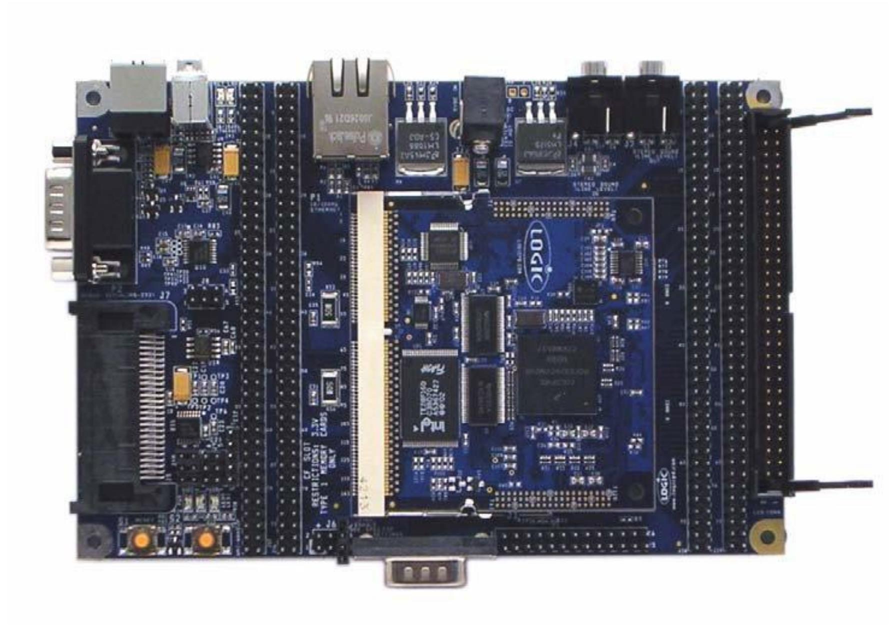
Important jumper settings (board REV A)

- For USB host operation
  - J44 shunt on position 1-2 (VBUS_OC)
  - J45 shunt on position 1-2 (VBUS_EN)
- For booting from NOR flash:
  - J33 shunt on position 1-2 (select Top flash boot mode)
- For UART operation
  - J12 shunt on position 2-3 (select U1RXD)
  - J13 shunt on position 2-3 (select U1TXD)
  - J12 shunt on position 1-2 (select U2RXD)
  - J13 shunt on position 1-2 (select U2TXD)

Board specific build targets:

Ext Flash (Debug and Release) - This target enables a standalone operation from on-board flash memory. External SDRAM memory is used for variables by default. The linker command file can be changed easily to allocate variables in the internal SRAM memory.

7.4.2 M5329EVB



Core Clock	240 MHz	
Bus Clock	80 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Jumper settings

- For UARTB operation:
  - Place 3 jumpers on J9 pins 1-2, 3-4, 5-6
- For CAN operation:
  - Place 3 jumpers on J10 pins 1-2, 3-4, 5-6
- Do no put jumpers simultaneously on J9 and J10

Known Issues:

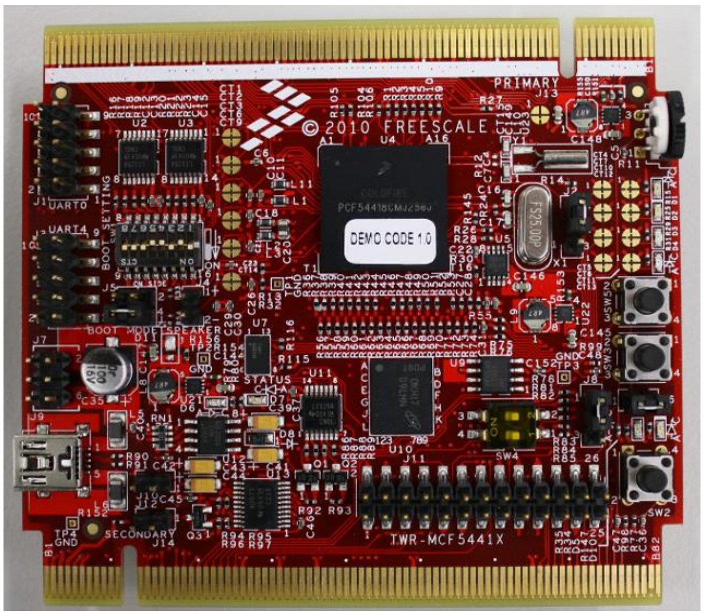
- Rev. A is not supported by MQX

7.5 ColdFire V4

7.5.1 TWR-MCF54418-KIT

The MCF54418 Patch supports the following hardware configuration:

- TWR-MCF54418 Rev. D processor board
- TWR-SER2 Rev. C serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM Rev. B memory extension board



Core Clock	250 MHz	
Bus Clock	125 MHz	
Default Console	ttyd:	RS232
BSP Timer	PIT0	

Important - both processor and serial board (TWR-SER2) has to be plugged into the Elevator bus. The MCF54418 processor is using external clock generated by Ethernet PHY on the Serial card

Jumper settings:

TWR-MCF54418 Rev.D board (use highlighted setting for basic MQX operation)

- **J2 on position 1-2**		Input Clock Selection
	1-2	external clock (RMII clock from TWR-SER2 board)
	2-3	onboard 25MHz clock
- **J8 on position 1-2**		TCK/PSTCLK Routing:
	1-2	PSCCLK routed to pin 24 of BDM header J11

2-3 PSTCLK routed to pin 6 of BDM header J11

- **J6 on position 1-2** To enable PE micro debugger
- **J5 on position 3-4** Boot Mode Selection

1-2 & 3-4	Internal RCON
3-4	External RCON
No Jumper	Serial Boot

- **J4 no jumper** 8 Ohm speaker connector
- **J10 no jumper** IRQ active at high level
- **J12 no jumper** MCU Reset In
- **SW4:** Both off
- **SW1:** 1-on, 2-off, 3-on, 4-off, 5-off, 6-on, 7-on, 8-on (booting from NAND)

TWR-SER2 rev C Board: (use highlighted setting for basic MQX operation)

- **J1 on position 2-3** RS232/485 RX Select (UART1)

1-2	RS485 Mode (connects RX to RO)
2-3	RS232 Mode (connects RX to R1OUT)

- **J2 on position 2-3** RS232/485 TX Select (UART1)

1-2	RS485 Mode (connects TX to DI)
2-3	RS232 Mode (connects TX to T1IN)

- **J4 no jumper** Can Isolation

1-2	Connects CAN_S to S
3-4	Connects CAN_TX to TXD
5-6	Connects CAN_RX to RXD

- **J7 on positions 1-2, 3-4** **JS16 RS232 Isolation (UART0)**

1-2	Connects RX to S08JS16 RXD
3-4	Connects TX to S08JS16 TXD

- **J8 no jumper** Power Down Port B

1-2	Disables Ethernet PHY B

- **J9 no jumper** Power Down Port A

1-2	Disables Ethernet PHY A

- **J11 no jumper** RS485 Config (UART1)

1-2	Loopback Mode (connects RE to DE)
3-4	Loopback Mode (connects TX0_P to RX0_P)
5-6	Loopback Mode (connects TX0_N to RX0_N)
7-8	NC
9-10	5V Supply to DB9

- **J13 on position 1-2** RS232/485 Disable (UART 1)
  - 1-2 Disables RS485
  - 2-3 Disables RS232
- **J16 no jumper** VBUS OC Isolation
  - 1-2 Connects USB VBUS OC to Elevator
- **J19 no jumper** UART2 Connector
- **J20 no jumper** UART3 Connector
- **J21 no jumper** VBUS EN Isolation
  - 1-2 Connects USB VBUS EN to Elevator
- **J22 no jumper** RS232 (UART2) Isolation
  - 1-2 Connects TX to T1IN
  - 3-4 Connects RX to R1OUT
  - 5-6 Connects RTS to T2IN
  - 7-8 Connect CTS to R2OUT
- **J23 no jumper** RS232 (UART3) Isolation
  - 1-2 Connects TX to T1IN
  - 3-4 Connects RX to R1OUT
  - 5-6 Connects RTS to T2IN
  - 7-8 Connect CTS to R2OUT
- **J24 no jumper** USB Device Mode
  - 1-2 Device Mode (capable of powering Tower System)
- **SW1** 1-on ,2-on (MII MODE pull-up, RXDV) 3,4,5,6,7,8 off
- **SW2** 1-on, 3-on (MII MODE pull-up, 50MHz) 2,4,5,6,7,8 off

TWR-MEM Rev.A – TWR MEM can operate only with TWR-SER2 card in default setting. Use TWR–SER board for SDHC operation

For eSDHC operation:

- J12: (SD-SEL1) on position 1-2 to enable SD Card detect signal
- J12: (SD-SEL1) on position 5-6 to enable SD Card data[1] signal
- J12: (SD-SEL1) on position 7-8 to enable SD Card data[2] signal
- J12: (SD-SEL1) on position 9-10 to enable SD Card cmd pull up
- J12: (SD-SEL1) on position 11-12 to enable SD Card data[0] pull up
- J2: (SD-SEL2) on position 2-3 to enable SD Card data[3] pull down
- J3: (SD-CS) on position 1-2 to enable SD Card data[3] signal
- J13: on position 1-2 to enable SD Card write protect signal

Board-specific build targets:

Ext Flash (Debug and Release) - these targets enable to build applications suitable for booting the system up from external NAND Flash memory. After the reset the initialization code of the application (bootstrap) is loaded from NAND Flash to SRAM. This initialization code copies the rest of the application to the DDR RAM memory and executes the application there. Note, that this could take a while especially if a large application is started. See NAND Flashing procedure bellow. There are two variants of Ext Flash target in the application projects: one for external P&E BDM interface (PEBDM Ext Flash) and one for on-board OSBDM interface (OSBDM Ext Flash). The OSBDM debugging connection does not work correctly with the old version of the OSBDM firmware. Please update the OSBDM firmware before using this target. Note that when using the OSBDM interface the TWR-MCF54418-KIT still has to be powered by the USB attached to the primary elevator.

See chapter 2.3 Build Targets for more details about standard build targets.

NAND Flashing Procedure:

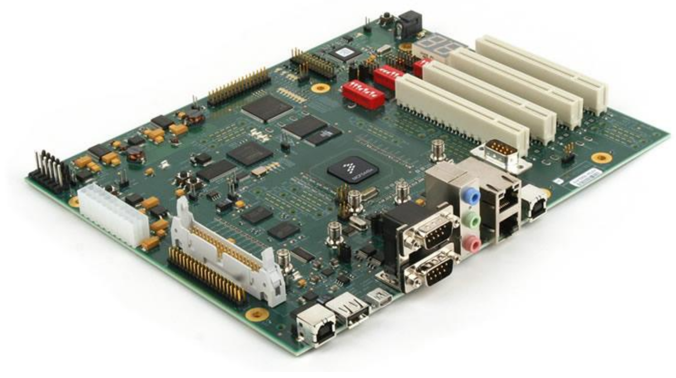
The CodeWarrior Development Studios for ColdFire V7.2.2 and for MCU v10.x do not provide NAND Flashing functionality, this functionality is planned for future releases. The MQX release contains standalone CFFlashprog utility which enables NAND Flashing using P&E Micro BDM interface (OSBDM is not supported). The NAND Flashing Procedure is as follows:

- Connect the P&E Micro BDM cable to the board and switch on the power
- Compile the PEBDM Ext Flash target in the selected CodeWarrior project
- Ensure you have SW1 DIP switch set correctly for the NAND booting (see jumper setting above)
- Locate <output_name>.rbin file which should be created in the application output directory
- Open Windows Command Line Prompt (run cmd.exe) and change directory to `“<install_dir>\tools\flash_programmer\CFFlashprog”`
- Use ***cf.exe NAND erase M54418TWR_nand 0 200000*** command to erase first 2Mbytes of NAND Flash memory
- Use ***cf.exe NAND write M54418TWR_nand 0 200000 1 < path_to_rbin_file>”*** command to write application code to the NAND Flash memory
- For more detailed description of the NAND Flash tool see `<install_dir>/tools/flash_programmer/CFFlashprog/ReadMe.txt`

Known Issues:

- Some Compact Flash cards does not work correctly with TWR-MEM and MQX CF Card driver. There may be several reasons:
  - An issue in the TWR-MEM CPLD code REV A causes incorrect communication with some types of cards (e.g. Kingston). A fixed CPLD firmware is available in `<install_dir>/mqx/source/io/pccardtwr_mem_pccard_cpld/` folder. The firmware can be loaded to the TWR-MEM CPLD using Altera Quartus II design tool and BLASTER connection cable.
  - M54418 MQX CF card driver incorrectly detects the card in the slot. If you experience this behavior, connect two pull-up resistors between card detect pins (CF_CD1, CF_CD2) and 3.3V VCC.

7.5.2 M54455EVB



Core Clock	266 MHz	
Bus Clock	133 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- To assure correct external Flash mapping as it is assumed by MQX setup
  - SW1[3] must be ON (default setting) so the FLASH0_CS is mapped to FB_CS0 and FLASH1_CS to FB_CS1
- To enable both Ethernet interfaces
  - **SW1[5]** set ON to enable PHY1 (disables ATA interface)
- To enable ATA operation
  - **SW1[5]** set OFF (disables second PHY device)
- For USB Host operation
  - The on-chip transceiver is supported only (external ULPI is not supported). Use USB Host connector to connect devices.
- For USB Device operation
  - Use USB Dual connector only to connect to host device.

Board-specific build targets:

- Flash0 Boot (Debug and Release) – On M54455EVB, this target enables to build applications suitable for booting the system up from the Flash0 memory. The initialization code of the application is located in Flash0. After reset, it copies the rest of the application to SDRAM memory and continues execution there.
  Configuration file to be used in CodeWarrior FlashProgrammer tool with this target is available as `tools/flash_programmer/config/M54455EVB_EXTFLASH0_512KB.xml`
- Flash1 Image (Debug and Release) – This target builds potentially large applications in a form of run-able image. This image can be flashed into Flash1 on M54455EVB and may be started from uBoot or other kind of bootloader. After start, this image copies itself to SDRAM

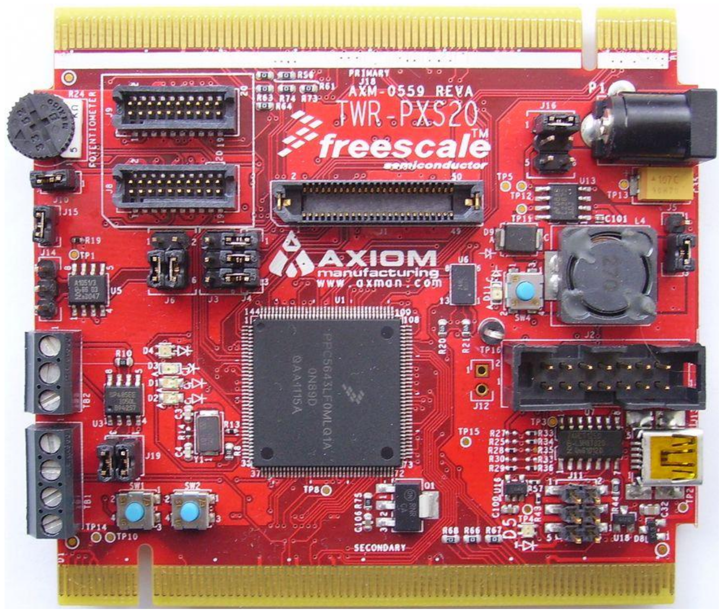
and continues execution there.
Configuration file to be used in CodeWarrior FlashProgrammer tool with this target is available as `tools/flash_programmer/config/M54455EVB_EXTFLASH1_16MB.xml`

See chapter 2.3 Build Targets for more details about standard build targets.

7.6 PowerPC

7.6.1 TWR-PXS20

The TWRPXS20 BSP supports TWR-PXS20 REV.A board.



System Clock	80 MHz	
Crystal oscillator	40 MHz	
Default Console	ttya:	OSJTAG virtual serial
BSP Timer	E200 decrementer	Optionally PIT0 (QPIT)

Important jumper settings

- 5V power supply (J16)
  - 1-2 powered from elevator
  - 3-4 (default) powered from onboard mini USB
  - 5-6 powered from onboard DC-in connector
- 3V3 power supply (J5)
  - 1-2 powered from elevator
  - 2-3 (default) onboard 3V3 regulator
- FlexCAN
  - By default FlexCAN 1 connected to TWR_SER board is used
  - To use FlexCAN 0 connected to J14 connector, position J6 jumpers to 3-5 and 4-6 and set J5 as closed (factory default positions)

Microcontroller mode

The PXS20 microcontroller may operate either in lock-step mode (LSM) or decoupled parallel mode (DPM). In LSM both cores execute the same code at the same time (for safety applications) while in DPM cores execute code independently on each other, so there are effectively two processors available. The mode is selected by LSM/DPM bit in shadow FLASH area. Factory setting is LSM. The mode cannot be changed in runtime, thus proper mode needs to be set prior running application on the MCU.

The TWRPXS20 BSP expects microcontroller to work in LSM. As this is the factory default, there is no need for switching the microcontroller mode.

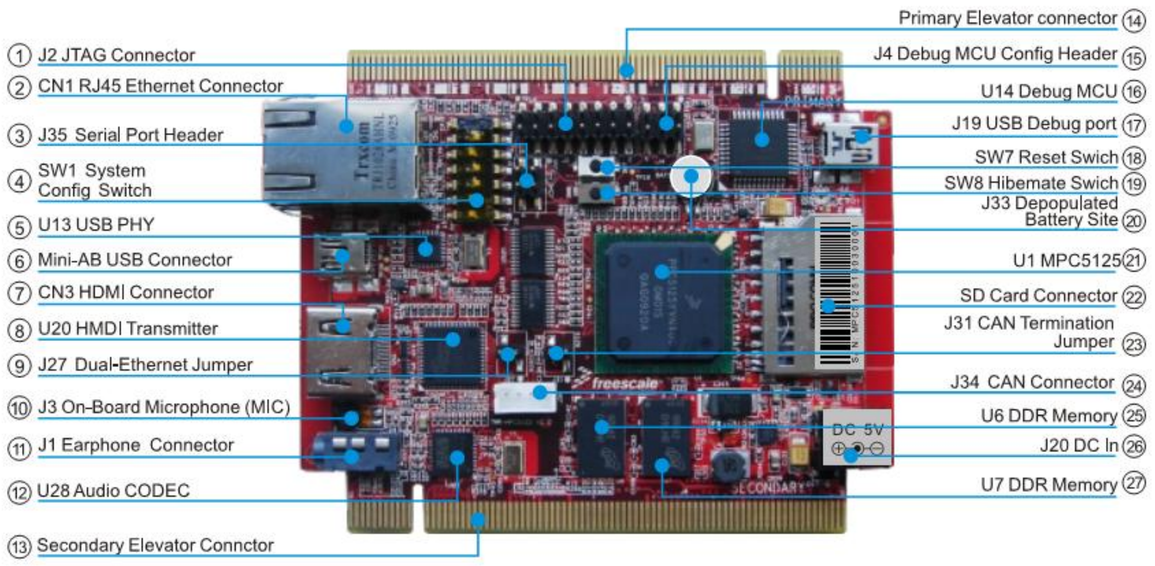
Unsupported peripherals

The current BSP does not contain support for analog peripherals (no ADC support). Onboard SPI accelerometer does not need any extra support on from BSP (expect of SPI), however the accelerometer used is currently not currently covered by “accelerometer” demo application.

7.6.2 TWR-MPC5125-KIT

TWR-MPC5125 processor board with

- [optional] TWR-SER serial board
- [optional] TWR-ELEV four-storey elevator boards



Core Clock	400 MHz	
Bus Clock	200 MHz	
Default Console	ttyb:	USB-to-serial port (J19)
BSP Timer	Decrementer	

The MPC5125 has two integrated Fast Ethernet Controllers (FEC1 & FEC2).

- Enabling FEC1
  - FEC1 is enabled by default and uses Reduced Media Independent Interface (RMII) mode to communicate to the Ethernet transceiver on the TWR-MPC5125 module

itself. The Ethernet port is accessible through the RJ45 jack on the TWR-MPC5125 module.

- Enabling FEC2
    - **FEC2 signals are multiplexed with USB1 on the MPC5125. Therefore only one of these functions can be selected at a time**. By default, the USB1 function is enabled in order the factory-default Linux image boots up correctly. The TWR-MPC5125 module uses multiplexers to route the signals to the appropriate location depending on what is desired at the time. The path of the signals is selected by **jumper J27**.

Default console channel

The USB-to-serial port available on the TWR-MPC5125 board (port J19) is installed as `ttyb:` device and is used as default console channel. The RS232 interface of TWR-SER board is installed as `ttyj:` device.

Important jumper settings:

- To use msCAN applications
    - J31, short 1-2 (CAN2 termination ON)
- Enabling both Ethernet controllers (2nd port available on the optional SER board)
    - TWR-MPC5125 J4, short 1-2, remove 3-4 (Serial-to-USB bridge enabled).
    - TWR-MPC5125 J27, short 1-2 (Enable second Ethernet port).
    - TWR-MPC5125 SW1 System Config Switch (default): 1-off, 2-on, 3-off, 4-off, 5-on, 6-on
    - TWR-SER J2, short 3-4 (Route the 50MHz clock to the Ethernet PHY).
    - TWR-SER J3, short 2-3 (Route the 50MHz to CLOCKIN0).
    - TWR-SER J12, short 9-10 (Enable RMII mode).
    - TWR-SER J15 on default position 1-2 (SER_SEL enabling RS232 operation)
    - TWR-SER J17 on default position 1-2 (RXD_SEL enabling RS232 operation)
    - TWR-SER J18 shunt removed (RTS_SEL no RS232 flow control)
    - TWR-SER J19 on default position 1-2 (TXD_SEL enabling RS232 operation)

System Config Switch setting:

SW1	Position Reset Configuration	Signal Description	Default
6	RST_CONF_ROMLOC0	Boot Device Select	
		0 = LPC Boot, 1 = NAND (NFC) boot	1
5	RST_CONF_BMS	Boot Mode Select	
		0 = boot low, 1 = boot high	1
4	RST_CONF_LPC_DBW0	LPC Data Port Size	
3	RST_CONF_LPC_DBW0	00=8bit, 01=16bit,	

		10=reserved, 11=32bit	00
2	RTC_CONF_LPCWA	LPC Word/Byte Address Mode	
		0 = word address mode,	1
		1 = byte address mode	
1	RTC_CONF_LPCMX LPC	Multiplex Mode	
		0 = Non multiplex mode,	0
		1 = multiplexed mode	

Board-specific build targets:

- **Ext Ram (Debug and Release) –** The applications built with these targets are intended to be run from pre-initialized RAM. There are three ways of how to execute the applications:
  - Use the CodeWarrior "Debug" feature to get the board initialized, to load the application to RAM and start a debugging session.
  - Use MQX NAND Flash bootloader to load an application binary to NAND Flash memory using TFTP and execute it. See mqx_install_dir>/mqx/example/bootloader for detailed application description.
  - Use uBoot firmware to load an application binary (or S-record) to a proper location in RAM and execute it by a "go" command to the base address. These steps can be automated in uBoot.

- **Ext NAND (Debug and Release)** – these targets have the same memory layout and content as application stored in NAND using Ext NAND Download target. The application can be debugged using this target.
- **Ext NAND Download (Debug and Release)** – CodeWarrior 9.2 for MobileGT does not provide a tool for flashing application to the NAND Flash Memory. Ext NAND Download target was developed to enable this functionality for MQX based projects. After this target is executed it writes the application into the NAND Flash memory and then finishes. The target application cannot be debugged in this mode, use Ext NAND Target for debugging instead.

NOTE:

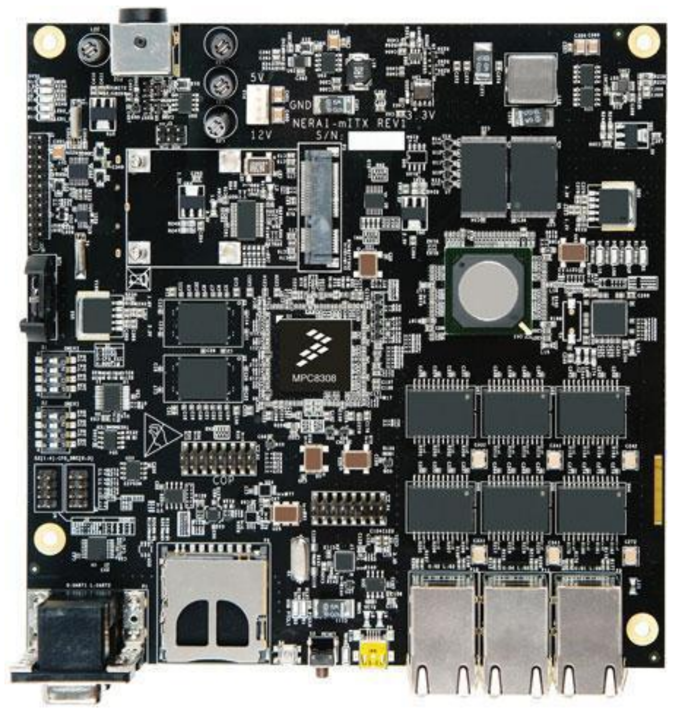
- Ext NAND Download targets always use Block 0 to store boot loader and Block 1 of NAND Flash device to store Kernel Code. Make sure these block are not bad before running any application. User can check for bad blocks by running nandflash demo available in MQX examples.
- The Ext NAND Download targets should be executed on stand-alone board (without TWR-SER and TWR-ELEV modules connected).
- The CodeWarrior 10.x brings incompatible version of USB TAB drivers - please reinstall the CW 9.2 USB drivers using `"c:\Program Files\Freescale\CodeWarrior for MobileGT V9.2\ccs\drivers\usb\x32\setup.bat"` if you experience difficulties with debugging.

See chapter 2.3 Build Targets for more details about standard build targets.

7.6.3 MPC8308 RDB

The MPC8308 supports the following hardware configuration:

- MPC 8308 RDB Rev. 2



Core Clock	400 MHz	
Bus Clock	133 MHz	
Default Console	ttya:	RS232-1
BSP Timer	Decrementer	

Important jumper settings:

- Dip Switch S1 All switches in ON position
- Dip Switch S2 All switches in ON position
- J4 RS-232C #2 1-3, 2-4 (default)
- J7 Power on reset 2-3 (default)

Board-specific build targets:

- uBoot (Debug and Release) - these targets build applications suitable for booting the system up from external NAND Flash memory via U-Boot.
- Ext RAM (Debug and Release) - these targets build applications suitable for downloading directly to SDRAM from CodeWarrior.

See the MQX Getting Started document chapter 2.2 for more details about standard build targets.

Using U-Boot to load and execute in SDRAM

MQX binary images can be loaded into flash using U-Boot. This requires u-boot flashed to the onboard NAND Flash and a network connection between the board and the host computer.
The board comes pre-programmed from the factory with U-Boot. On power up, the processor boots from the NAND Flash and u-boot takes control. U-Boot initializes the board and enables SDRAM. For information on U-Boot see the board user's guide.

- Configure your CodeWarrior project to generate a binary image. Go to the target settings panel, click EPPC linker, and change the Binary File: drop down item to One.
- On the host computer, open a TFTP Server of your choice. If you do not have a TFTP Server, there are a number of free TFTP servers available for download from the internet.
- Set the TFTP directory to C:\tftp.
- Copy your application binary file to C:\tftp.
- Ensure switches are set correctly to allow for nand booting. See the board user's guide.
- Power up the board.
- At the U-Boot prompt type the following commands to set the network connection settings (change the IP address to appropriate values for your environment):

```
set ipaddr 192.168.1.9
set serverip 192.168.1.101
set gatewayip 192.168.1.1
set netmask 255.255.255.0
save
```

- Transfer the binary file to the on-board memory. At the U-Boot prompt, type:

```
tftp 10000 uboot_d.bin
```

- Erase the NAND Flash blocks 0x80000 to 0x81400:

```
nand erase 0x80000 0x81400
```

- Write from SDRAM to the NAND Flash:

```
nand write 0x10000 0x80000 0x1400000
```

- Then make a command to boot MQX:

```
set mqxboot “nand read 0x10000 0x80000 0x1400000; go 0x10000”
save
```

- To make the MQX application the auto-boot option:

```
set bootcmd run mqxboot
save
```