Freescale MQX™ RTOS 3.7.0 Release Notes

PRODUCT:	Freescale MQX™ RTOS
PRODUCT VERSION:	3.7.0
DESCRIPTION:	Freescale MQX™ RTOS, version 3.7.0
RELEASE DATE:	April 8th, 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

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

freescale™
semiconductor

Rev. 15
04/2011

Table of Contents

Freescale MQX™ RTOS 3.7.0 Release Notes ... i

1 Read Me First ... 2

1.1 Requirements ... 2
1.2 Special instructions ... 3

2 What is New? ... 5

3 Release Content ... 8

4 MQX Release Overview ... 11

4.1 MQX RTOS PSP ... 12
4.2 MQX RTOS BSPs ... 12
4.3 MQX MFS ... 16
4.4 MQX RTCS ... 16
4.5 MQX USB Host ... 16
4.6 MQX USB Device ... 16
4.7 MQX Shell ... 17
4.8 Changing the MQX Source Files ... 17
4.9 Building the MQX Libraries ... 17
4.10 Example Applications ... 17

5 What is Missing? ... 20

6 Known Issues and Limitations ... 21

6.1 Performance of Code Running in MRAM ... 21
6.2 Network Communication Performance and Delayed ACKs ... 21
6.3 Default Kernel Configuration of Small-RAM Devices ... 21
6.4 USB Host HUB Examples ... 21
6.5 USB Host HUB Functionality with MCF5222x ... 22
6.6 ColdFire V1 Code Footprint in Debug Build Targets ... 22
6.7 OSBDM / OSJTAG Firmware Compatibility ... 22
6.8 Supporting “Hot Device Uninstall” in MQX I/O Subsystem ... 22
6.9 Spurious Interrupts on ColdFire Platforms ... 23
6.10 CodeWarrior version 10.1 ... 23
6.11 TWR-MEM Compact Flash interface issues ... 23
6.12 Kinetis Processor Support Package issues ... 24
6.13 IAR Embedded Workbench limitations ... 24
6.14 Prinf floating point (%f) does not work on IAR and Kinetis ... 24
6.15 USB EHCI stack cannot handle cached memory ... 24
6.16 FlashX driver cannot read data across different flash areas ... 24

7 Change Log ... 26

1 Read Me First

This release note documents the Freescale MQX™ RTOS version 3.7.0 released for Freescale ColdFire, PowerPC and Kinetis ARM®CortexM4 processor families.

1.1 Requirements

1.1.1 Development Tools

This Freescale MQX™ RTOS Release was compiled and tested with the following development tools:

- CodeWarrior Development Studio for Microcontrollers Version 6.3.1 (Build 10105)
  - Support available for ColdFire V1 devices
  - See build projects in `cwmcu63` subdirectories

- CodeWarrior Development Studio for ColdFire Architectures Version 7.2.2 (Build 11038)
  - Support available for ColdFire V2-V4 devices
  - See build projects in `cwcf72` subdirectories

- CodeWarrior Development Studio for Microcontrollers Version 10.1 (MCU build 110204)
  - Support available for Kinetis and ColdFire devices
  - See build projects in `cw10` subdirectories

- CodeWarrior Development Studio for MobileGT Version 9.2 (Build 81027) with MPC5125 Service Pack 1 (Build 090311)
  - Support available for PowerPC mobileGT MPC5125
  - See build projects in `cwmpc92` subdirectories

- IAR Embedded Workbench for ARM Version 6.10
  - Support available for Kinetis ARM®CortexM4 devices
  - See build projects in `iar` subdirectories

- IAR Embedded Workbench for ColdFire Version 5.3
  - Build projects are not part of Freescale MQX release – integration is provided by IAR Systems (www.iar.com)

- Sourcery G++ GNU Toolchain and the Eclipse IDE
  - Build projects are not part of Freescale MQX release – integration is provided by CodeSourcery (www.codesourcery.com)

1.1.2 System Requirements

The system requirements are defined by the development tools requirements. There are no special host system requirements for hosting the Freescale MQX™ RTOS distribution itself.

Minimum PC configuration:

As required by Development and Build Tools

Recommended PC configuration:

2 GHz processor – 2 GB RAM - 2 GB free disk space.

Software requirements:

OS: As required by Development and Build tools (Windows XP SP2 or later)

1.1.3 Target Requirements

The Freescale MQX™ RTOS in this release supports the evaluation boards mentioned below. There are no special requirements for the target hardware which would be out of scope of what each board requires for its operation (power supply, cabling, jumper settings etc). More details about board-specific setup for MQX operation are available in “MQX Getting Started” document.

Evaluation boards supported:

Kinetis ARM® Cortex M4

- TWR-K40X256 Evaluation Board
- TWR-K60N512 Evaluation Board

ColdFire V1

- TWR-MCF51AG Evaluation Board
- TWR-MCF51CN Evaluation Board
- TWR-MCF51JE Evaluation Board
- TWR-MCF51MM Evaluation Board
- DEMOEM Evaluation Board based on MCF51EM256
- DEMOAC Evaluation Board based on MCF51AC128
- EVB51JM128 Evaluation Board

ColdFire V2

- M5208EVB Evaluation Board
- M52223EVB Evaluation Board
- M52233DEMO Evaluation Board
- M52235EVB Evaluation Board
- M52259EVB Evaluation Board
- M52259DEMOKIT Evaluation Board
- TWR-MCF52259 Evaluation Board
- M52277EVB Evaluation Board

ColdFire V3

- M53015EVB Evaluation Board
- M5329EVB Evaluation board

ColdFire V4

- TWR-MCF54418 Evaluation Board
- M54455EVB Evaluation Board

PowerPC:

- TWR-MPC5125 Evaluation Board

1.2 Special instructions

1.2.1 Setup Installation instructions

Run the self-extracting MQX installer application and proceed according to instructions on screen.

Rebuilding libraries is also recommended in case MQX is not installed in default location (which is C:\Program Files\Freescale\Freescale MQX 3.7). Otherwise, any time the application is started under debugger, the debugger may ask for a path to MQX source code files.

For build instructions please refer to “**MQX Getting Started**” document chapter “Building the MQX Libraries”.

2 What is New?

This section describes the major changes and new features implemented in this release.

- New Board support packages:
  - MCF51AG-based TWR-MCF51AG BSP
  - MCF51JE-based TWR-MCF51JE BSP
  - MCF5208-based M5208EVB BSP
  - MCF5329-based M5329EVB BSP
  - MCF54418-based TWR-MCF54418 BSP
- Kinetis BSPs TWRK60X512 and TWRK40N256 modified:
  - Clock speed changed from 48MHz to 96MHz
  - Programmable Gain Amplifier – PGA enabled in ADC driver
  - Flash driver extended by FlexNVM functionality (EEPROM emulation)
- TWR-MPC5125 BSP was extended by:
  - USB Host and Device
  - NAND, SDHC, SD Card, SPI and I2C drivers
  - NAND boot targets and Nand Flashing targets are available. See details in MQX Getting Started document chapter TWR-MPC5125-KIT BSP.
- CodeWarrior Development Studio for Microcontrollers Version 10.1 support
  - TAD and New project wizard for Coldfire and Kinetis Platforms
  - MQX libraries, example and demo projects
- Kinetis BSP projects are CodeWarrior 10.1 Processor Expert Ready
  - Processor Expert drivers are enabled in MQX RTOS environment
  - Two BSPs with Processor Expert drivers enabled `<mqx>/build/twrk40n256_pe` and `<mqx>/build/twrk60x513_pe`
  - Example application demonstrating PE functionality `\demo\pe_demo`
  - Processor Expert drivers are supported for Kinetis platform only
- Kinetis scheduler enhanced
  - The _int_install_kernel_isr() call to install a non-MQX interrupt service routine into interrupt table is supported
  - The RAM-based vector table is supported.
- New smaller and faster version of GPIO driver prepared – LWGPIO driver. The driver is currently available for MCF52259 and MK40 and MK60 based BSPs only. See `doc/MQXIOUG.pdf` documentation and `\mqx\examples\lwgpio` example application for details. The driver will be ported to other platforms in future MQX releases.
- Serial driver was extended by new options allowing RS485 half duplex communication. New ioctl command for handling HW flow control (RTS/CTS signal) and SCI parity were added.

- The output directory for MQX PSP and BSP libraries has been changed – instead of `lib/<board_name >/mqx` are newly used `lib/<board_name>/psp` and `lib/<board_name>/bsp` folders. See `doc/FSL_MQX_3_7_Porting_Guide.pdf` for instructions if you are porting the project based on older MQX releases.
- Mem_copy and mem_zero functions were set to speed optimized version by default. The code is longer but up to seven times faster than original implementation.
- Byte-swapping macros optimized and unified in PSP library (psp.h)
- Handling of multicast packets has been simplified in RTCS, no functional change.
- Task priorities chaged from 5 to 8 in several example applications. Note that task priorities below 7 have special functionality with regards to interrupt masking.
- HTTP server reworked. The server is able to operate in one of three modes: polled mode, static task pool or dynamic task created per session. See RTCS documentation for details.
- MFS USB example application modified to support multiple USB sticks simultaneously
- USB Audio Class support added into USB Host stack for KHCI based platforms. Example application prepared for ColdFire V2 and Kinetis based platforms (`usb\host\examples\audio\audio_stream`)
- TCPIP stack (RTCS) is newly available for platforms without Ethernet controller. Ready for WiFi and PPP support.
- CodeSize script and reports updated. The script is able to generate detailed reports for PSP, BSP and newly also RTCS, MFS and USB libraries
- User documentation was updated

Bugfixes

- Management of free memory for Kinetis devices using _mem_alloc_at or _lwmem_alloc_at allocation has been fixed. Using _mem_alloc_at could lead to losing part of free memory.
- USB Virtual_com example code was fixed. Communication buffers are now allocated in un-cached memory.
- Serial driver: The IO_IOCTL_SERIAL_TRANSMIT_DONE command was corrected. The IO_IOCTL_SERIAL_SET_HW_SIGNAL and IO_IOCTL_SERIAL_SET_PARITY commands were implemented for all platforms
- SDHC sporadic read errors problem solved for Kinetis and MCF54418 devices. Default baud-rate has been reduced. Adaptive baudrate setting will be implemented in future MQX released.
- The MCF54418 GPIO driver interrupt related functionality has been fixed.
- The NAND ECC and back block byte handling in the MCF54418 NAND flash driver has been fixed. Wrong ECC size was defined in BSP header file.
- Several USB Host stack EHCI bug-fixes improving stack stability and compatibility with same mass storage devices (Error handling during enumeration improved, improved packet receiving algorithm)
- MFS free cluster counting has been fixed.

- ROM section setting corrected in the linker files for MCF51CN, MCF51AC and MCF51JM BSPs.
- MFSCFG_FAT_CACHE_SIZE set to value 2 for all configurations to improve USB Mass Storage write speed for devices with EHCI peripheral module.
- IO_IOCTL_SERIAL_TRANSMIT_DONE result was inverted bug was fixed
- Register definition file MCF54455 platform was corrected (mcf5445_edma_struct)
- QSPI baud rate calculation source code was unified across all platforms. The QWR setting was corrected.
- USB EHCI Host does not handle data properly, the packets are sometime rejected – fixed in asynchronous scheduler; the code accessing overlay area was removed
- M53015EVB board does not start without debugger – fixed in BSP initialization code for this platform
- psp_prv.s file kernel logging related compilation check was corrected
- HVAC and Security Webserver demo applications was updated to run on Kinetis platforms and IAR compiler
- SNMP compilation macros have been fixed. The RTCSCFG_ENABLE_SNMP macro was added to all SNMP related files.
- MPC5125 RTC example build error has been resolved
- LWDNS behavior was corrected for TWRK60X512 platform. Endian swapping macro was incorrectly used.
- GPIO driver – fclose() on Kinetis based platforms was corrected. The function incorrectly uninstalled the interrupts for other open pins.
- USB Host stack enumeration error handling has been improved. The stack is trying to re-assert transaction if error occurs.
- USB Host stack reviewed to correctly handle Big to Little endian conversion. USB MSD device class endianess problem fixed on Kinetis platform.
- ADC driver for ColdFire V1 platforms was corrected – wrong use of lwevent in conversion complete lead to incorrect signaling of the end of measurement
- Compilation errors if RTCS_ENABLE_IP_REASSEMBLY was enabled were corrected
- The file lwe_gets.c (lw event get signaled) was not included in PSP libraries for all platforms
- Function FLEXCAN_Rx_message() returned FLEXCAN_NO_MESSAGE when FlexCAN message box overrun. The function has been changed to return FLEXCAN_MESSAGE_OVERWRITTEN.

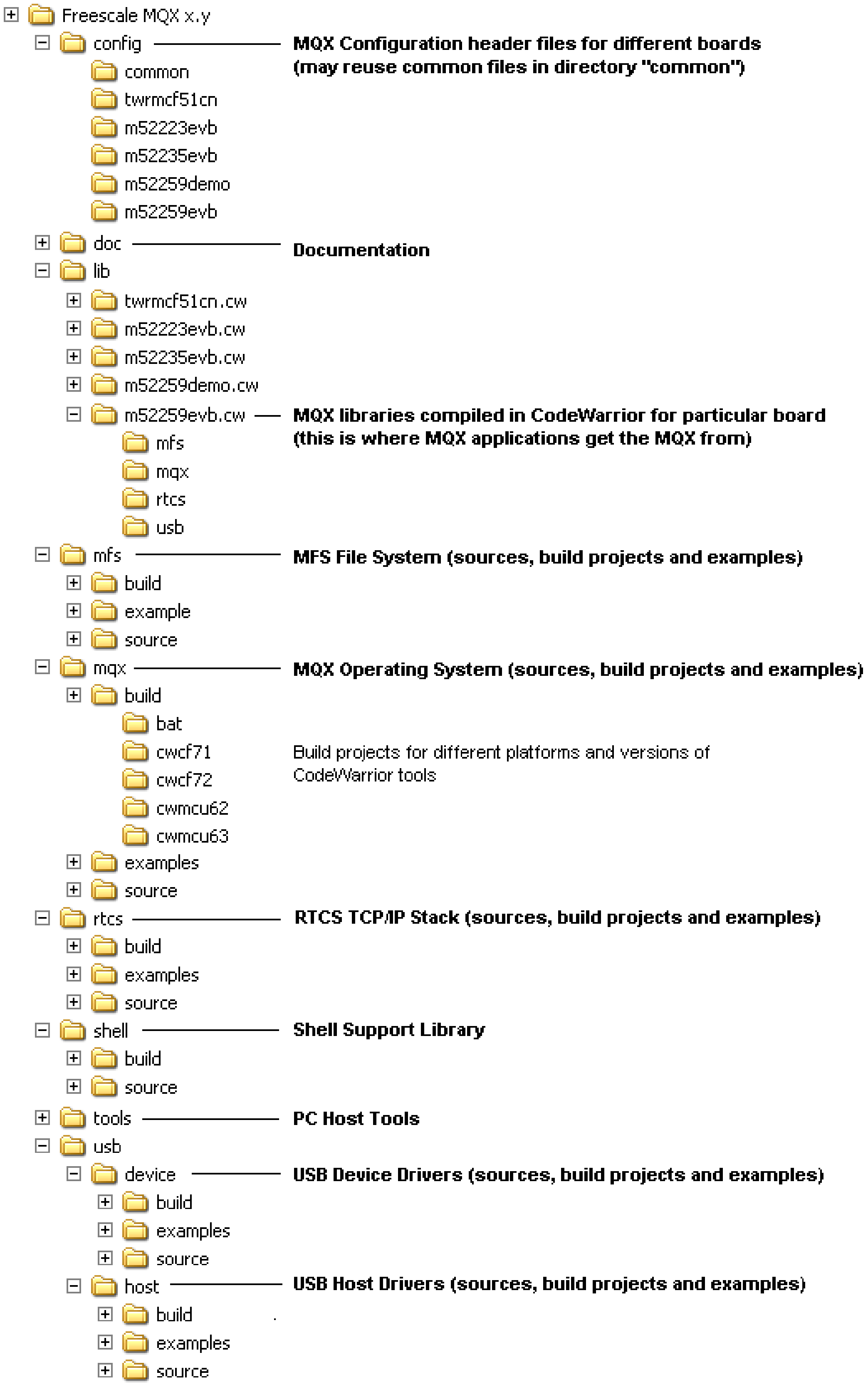
3 Release Content

This section gives an overview about the release content.

Deliverable	Location	Status
Pre-compiled MQX Libraries	**<install_dir>/lib/...**	
MQX PSP - platform support package	.../lib/<board>/psp	Updated
MQX BSP - board support package	.../lib/<board>/bsp	Updated
MQX RTCS (TCP/IP network stack)	.../lib/<board>/rtcs	Updated
MQX MFS (File System)	.../lib/<board>/mfs	Updated
MQX USB Host Drivers	.../lib/<board>/usb/host	Updated
MQX USB Device Drivers	.../lib/<board>/usb/device	Updated
MQX Shell Library	.../lib<board>/shell	Updated
Configuration Files and Mass-Build Projects	**<install_dir>/config/...**	
Configuration and mass-build project for all supported boards	.../config/<board>	Updated
MQX PSP, BSP Source Code and Examples	**<install_dir>/mqx/...**	
MQX PSP source code for Kinetis	.../mqx/source/psp/cortex	Updated
MQX PSP source code for ColdFire	.../mqx/source/psp/coldfire	Updated
MQX PSP source code for PowerPC	.../mqx/source/psp/powerpc	Updated
MQX PSP build projects	.../mqx/build/<*compiler*>/psp_<board>	Updated
MQX BSP Source Code	.../mqx/source/bsp/<board>	Updated
MQX BSP build projects	.../mqx/build/<*compiler*>/bsp_<board>	Updated
MQX example applications	.../mqx/examples/...	Updated
RTCS Source Code and Examples	**<install_dir>/rtcs/...**	
RTCS source code	.../rtcs/source	Updated
RTCS build projects	.../rtcs/build/<*compiler*>/rtcs_*	Updated
RTCS example applications	.../rtcs/examples	Updated
MFS Source Code and Examples	**<install_dir>/mfs/...**	
MFS source code	.../mfs/source	Updated
MFS build projects	.../mfs/build/<*compiler*>/mfs_*	Updated
MFS example applications	.../mfs/examples	Updated
USB Host Drivers Source Code and Examples	**<install_dir>/usb/host/...**	
USB Host source code and class drivers	.../usb/host/source	Updated
HUB Class Driver	.../usb/host/source/classes/hub	Updated
Human Interface Device (HID) Class Driver	.../usb/host/source/classes/hid	Updated
Mass Storage (MSD) Class Driver	.../usb/host/source/classes/msd	Updated
Printer Class Driver	.../usb/host/source/classes/printer	Updated
CDC Class Driver	.../usb/host/source/classes/cdc	Updated
USB Host build projects	.../usb/host/build/<*compiler*>/usb_hdk_*	Updated
USB Host example applications (HID, MSD, HUB)	.../usb/host/examples	Updated
USB Device Drivers Source Code and Examples	**<install_dir>/usb/device/...**	
USB Device source code	.../usb/device/source	Updated
USB Device build projects	.../usb/device/build/<*compiler*>/usb_ddk_<board>	Updated
USB Device example applications (HID, MSD, CDC, PHDC)	.../usb/device/examples	Updated
Shell Library Source Code	**<install_dir>/shell/...**	
Shell source code	.../shell/source	Updated
Shell build projects	.../shell/build/<*compiler*>/shell_<board>	Updated
CodeWarrior Support	**<CodeWarrior_dir>/...**	
MQX New project stationery for classic CodeWarrior	<cw_dir>/stationery/Freescale MQX 3.7	Updated

MQX New project wizard for CW 10	<cw10_dir>/eclipse/plugins *and*	Updated
	<cw10_dir>/MCU/lib/wizard_data/mqx/3.7	
MQX Task-aware Debugger plug-in for classic CodeWarrior	<cw_dir>/bin/plugins/debugger/rtos/	Updated
MQX Task-aware Debugger plug-in for CW10	<cw10_dir>/MCU/bin/plugins/debugger/rtos	Updated
TAD and Stationery files for manual copy-installation into another CodeWarrior	.../tools/codewarrior_extensions/...	Updated
PC Host Tools	**<install_dir>/tools**	
TFS Make Utility	…/tools/mktfs.exe	from 3.0
Check for Latest Version tool	…/tools/webchk.exe	from 3.0
AWK interpreter (GNU General Public License)	…/tools/gawk.exe	from 3.1
SNMP code generation scripts	…/tools/snmp/*.awk	from 3.1
Timing HTML report tool (for mqx/examples/benhmrk/timing)	…/tools/timing.exe	Updated
Code size HTML report tool (for mqx/examples/benhmrk/codesize)	…/tools/codesize.exe	from 3.6.0
TAD string and configuration files	…/tools/tad	Updated
Demo Applications	**<install_dir>/demo**	
Various demo applications described in detail by step-by-step 'Lab' documents.	…/demo/...	Updated
Documentation	**<install_dir>/doc**	
User Guides and Reference Manuals for MQX RTOS, RTCS, MFS, IO Drivers, USB etc.	…/doc	Updated

The following picture shows the Freescale MQX™ RTOS directories installed to the user host computer (subdirectories reduced for clarity):



```
Freescale MQX x.y
    config ———— MQX Configuration header files for different boards
                (may reuse common files in directory "common")
        common
        twrmcf51cn
        m52223evb
        m52235evb
        m52259demo
        m52259evb
    doc ———— Documentation
    lib
        twrmcf51cn.cw
        m52223evb.cw
        m52235evb.cw
        m52259demo.cw
        m52259evb.cw ——— MQX libraries compiled in CodeWarrior for particular board
                         (this is where MQX applications get the MQX from)
            mfs
            mqx
            rtcs
            usb
    mfs ———— MFS File System (sources, build projects and examples)
        build
        example
        source
    mqx ———— MQX Operating System (sources, build projects and examples)
        build
            bat
            cwcf71      Build projects for different platforms and versions of
            cwcf72      CodeWarrior tools
            cwmcu62
            cwmcu63
        examples
        source
    rtcs ———— RTCS TCP/IP Stack (sources, build projects and examples)
        build
        examples
        source
    shell ———— Shell Support Library
        build
        source
    tools ———— PC Host Tools
    usb
        device ———— USB Device Drivers (sources, build projects and examples)
            build
            examples
            source
        host ———— USB Host Drivers (sources, build projects and examples)
            build
            examples
            source
```

4 MQX Release Overview

This is MQX RTOS release done by Freescale Semiconductor. It is targeting various Freescale ColdFire microcontrollers, Kinetis ARM® CortexM4 microcontrollers and Freescale Power Architecture.

The Freescale MQX™ RTOS is based on MQX version 2.50 as it was released by the company ARC International. The Freescale MQX™ RTOS release includes MQX real time kernel, core system components, integrated TCP/IP network stack - RTCS, file system – MFS and USB Host and Device stacks. These components used to be released and sold separately by ARC.

	MQX PSP+BSP Libraries	MFS Library (FAT File System)	RTCS Library (TCP/IP Stack)	Shell Library	USB Host Library	USB Device Library	UART (polled and interrupt driven	I2C (polled and interrupt driven)	SPI0 (polled and interrupt driven)	LWGPIO	GPIO	ADC	FLASHX (internal flash)	FLASHX (external NOR Flash)	NAND flash driver	ESDHC	Compact Flash Card driver	Resistive Touch-screen	SD Card driver (SPI or SDHC based)	RTC, IRTC (Real Time Clock)	DAC	DIU	FlexCAN / msCAN	FEC (Ethernet driver)	USB Host/Device driver
Kinetis																									
TWRK40X256	●	●	1)	●	●	●	●	●	●	●	●	●	●			●			●	●			●		●
TWRK60N512	●	●	●	●	●	●	●	●	●	●	●	●	●			●			●	●			●	●	●
ColdFire V1																									
TWRMCF51AG	●	●	1)	●			●	●	●		●	●	●						●	●					
TWRMCF51CN	●	●	●	●			●	●	●		●	●	●				●		●	●				●	
TWRMCF51JE	●	●	1)	●	●	●	●	●	●		●	●	●				●		●	●	●				●
TWRMCF51MM	●	●	1)	●	●	●	●	●	●		●	●	●				●		●	●	●				●
M51ACDEMO	●	●	1)	●			●	●	●		●	●	●												
M51EMDEMO	●	●	1)	●			●	●	●		●	●	●							●					
M51JMEVB	●	●	1)	●	●	●	●	●	●		●	●	●							●					●
ColdFire V2-V4																									
M5208EVBE	●	●	●	●			●	●	●		●	●	●											●	
M52223EVB	●	●	1)	●	●	●	●	●	●		●	●	●												●
M52233DEMO	●	●	●	●			●	●	●		●	●	●							●				●	
M52235EVB	●	●	●	●			●	●	●		●	●	●							●			●	●	
M52259EVB	●	●	●	●	●	●	●	●	●	●	●	●	●				●			●			●	●	●
M52259DEMO	●	●	●	●	●	●	●	●	●	●	●	●	●							●			●	●	●
TWRMCF52259	●	●	●	●	●	●	●	●	●	●	●	●	●				●		●	●			●	●	●
M52277EVB	●	●	1)	●	●	●	●	●	●		●			●					●	●			●		●
M53015EVB	●	●	●	●	●	●	●	●	●		●			●		●			●	●				●	●
M5329EVB	●	●	●	●	●	●	●	●	●		●			●	●		●		●	●				●	●
TWRMCF54418	●	●	●	●	●	●	●	●	●		●				●	●	●		●	●				●	●
M54455EVB	●	●	●	●	●	●	●	●	●					●						●				●	●
PowerPC																									
TWRMPC5125	●	●	●	●	●	●	●	●	●						●	●	●		●	●		●	●	●	●

● New in this release

1) Onchip ethernet not available, RTCS can be used with PPP or custom ene tdriver (spi, uart, etc. based)

4.1 MQX RTOS PSP

This version of Freescale MQX™ RTOS contains ARM® CortexM, ColdFire and PowerPC Platform Support Packages. Contact Embedded Access Inc. (www.embedded-access.com) for commercial-grade support and support of other Freescale platforms.

The Platform-specific code from `/mqx/source/psp/<platform>` is built together with generic MQX core files. These two parts combined form a static library generally referred as “psp” which, when linked to user application, enables the RTOS features to be used.

4.2 MQX RTOS BSPs

This release of Freescale MQX™ RTOS includes Board Support Packages for boards mentioned above.

The Board-specific code from `/mqx/source/bsp/<board>` is built with I/O driver files from `/mqx/source/bsp/io`. Together these two parts form a static library generally referred as “bsp” which, when linked to user application, enable the board and operating system to boot up and use the I/O driver services.

The following section gives an overview about drivers supported in the MQX BSPs.

4.2.1 I/O Drivers Supported

The following table gives an overview about I/O drivers available in the last MQX release. Like the whole I/O subsystem, the drivers are optional part of the MQX RTOS and their installation can be enabled or disabled in the BSP startup code. To save Code memory and RAM, most of the drivers are disabled in the `/config/<board>/user_config.h` file by default. Only the drivers required by demonstration applications (in the `/demo` folder) are enabled by default.

Note: When BSPCFG_ driver-enabling macros are not set in the `/config/<board>/user_config.h` file, the default setting is taken from BSP-specific header file located in `/mqx/source/bsp/`*`<board>`*`/`*`<board>`*`.h`.

It is an application programmer decision to enable automatic installation of the driver in the BSP startup code (by enabling appropriate BSPCFG_ENABLE_*XXX* macro in `user_config.h`) or manually in the application code.

TFS – Trivial Filesystem

Tiny filesystem which can be used as a simple read-only file repository instead of fully featured MFS. TFS is not installed in BSP startup code. It is a task for the application to initialize the TFS and pass a pointer to the filesystem image data. The **mktfs** tool is available (both as executable and PERL script) to generate the image from the existing directory structure. The RTCS HTTP example demonstrates use of TFS.

I2C I/O Driver

This driver supports a polled I2C interface in master mode. If enabled in user configuration the I2C device drivers are installed during the BSP startup code as “**i2c0:**” and “**i2c1:**”. Example application is provided in the MQX source tree.

SPI I/O Driver

This driver supports various SPI interfaces available in ColdFire V1 to V4 (SPI, QSPI, DSPI).

The driver supports master mode of operation only. If enabled in user configuration the SPI device drivers are installed during the BSP startup code as “**spi0:**” and “**spi1:**” (polled-mode) and “**ispi0:**” and “**ispi1:**” (interrupt-mode). Example application is provided in the MQX source tree.

FlexCAN Driver

This driver provides a C language API to the FlexCAN peripheral module. Example application is provided in the MQX source tree.

msCAN Driver

This driver provides a C language API to the msCAN peripheral module. Example application is provided in the MQX source tree.

RTC Driver

This driver provides a C language API to the Real Time Clock peripheral module and functions helping to synchronize clock time between RTC and MQX system. If enabled in user configuration, the RTC module is initialized and MQX time is renewed automatically during BSP startup.
The Real Time Counter module of MCF51xxx family is supported by the RTC driver as well. Some missing features of this module (comparing it to Real Time Clock module of MCF52xx) are emulated in the software driver.

Serial I/O Driver

The standard SCI (UART) driver supports both polled and interrupt-driven modes. If enabled in user configuration, the serial devices are installed as “**ttya:**”, “**ttyb:**” and “**ttyc:**” (polled mode) and “**ittya:**”, “**ittyb:**” and “**ittyc:**” (interrupt mode) automatically during BSP startup.

GPIO I/O Driver

This I/O driver provides a uniform interface to all GPIO pins available on a particular device. If enabled in user configuration, the GPIO driver is installed as “**gpio:**” automatically by the BSP startup. This driver will be replaced by LWGPIO for newly ported platforms.

LWGPIO I/O Driver

This driver provides a C language API to all GPIO pins available on a particular device. This driver is significantly faster and have smaller footprint than GPIO driver.

ADC Driver

This I/O driver provides a uniform interface to ADC channels. ADC uses internal PIT or other kind of timer for periodic sampling. If enabled in user configuration, the ADC driver is installed as “**adc:**” (“adc1:”, “adc2:” for platforms with multiple adc channels)device automatically by the BSP startup.

DAC Driver

This driver provides C language API to DAC peripheral module. The driver is adopted from the Freescale Processor Expert toolbox. The DAC driver is installed and used directly from the application code.

Flash I/O Driver

This I/O driver provided a standard interface to internal or external Flash memory. If enabled in user configuration, the Flash driver (called FlashX) is installed as “**flashx:**” device automatically by the BSP startup code (“flash0”, “flash1” etc. device names are used for FlashX device installed for external Flash memory).

For devices with internal Flash memory, the FlashX driver depends on several parameters passed in form of global symbols from application or from Linker Command File. See the example application provided in the MQX source tree.

ENET Driver

The low-level Ethernet driver is used by the RTCS TCP/IP software stack. The driver is initialized directly by the application before RTCS is first used. The RTCS Shell and HTTP examples demonstrate use of this driver.

PCCard I/O Driver

This I/O driver provides a low-level access to the PCCard functionality using Flexbus and CPLD circuit. The CPLD code can be found in `<install_dir>/mqx/source/io/pccard/<card_name>`. If enabled in the user configuration, the PCCard device driver is installed as "**pccarda:**" automatically during the BSP startup.

PCFlash I/O Driver

The Compact Flash Card I/O driver is installed on top of the PCCard low-level driver and enables standard disk drive operations. The MFS file system can be installed on top of this device. If enabled in user configuration, the PCFlash device driver is installed as "**pcflasha:**" automatically during the BSP startup.

SD Card I/O Driver

This I/O driver implements a subset of SD protocol v2.0 (SDHC). The driver can use MQX SPI driver or MQX SDHC driver to communicate with SD Card device. The driver should be installed at the application level, passing it a lower-layer driver handle. The MFS file system can be installed on top of this device.

ESDHC I/O Driver

This I/O driver covers the eSDHC peripheral module and provides low-level communication interface for various types of cards including SD, SDHC, SDIO, SDCOMBO, SDHCCOMBO, MMC and CE-ATA.

Resistive Touch-screen Driver

This I/O driver accesses the ADC and GPIO modules to detect touch events and acquire touch coordinates on a resistive touch-screen unit.

DIU Display driver

This driver provides a generic C language API to frame buffer-based display units. The driver is initialized and used from a user-application.

4.2.2 Serial Console

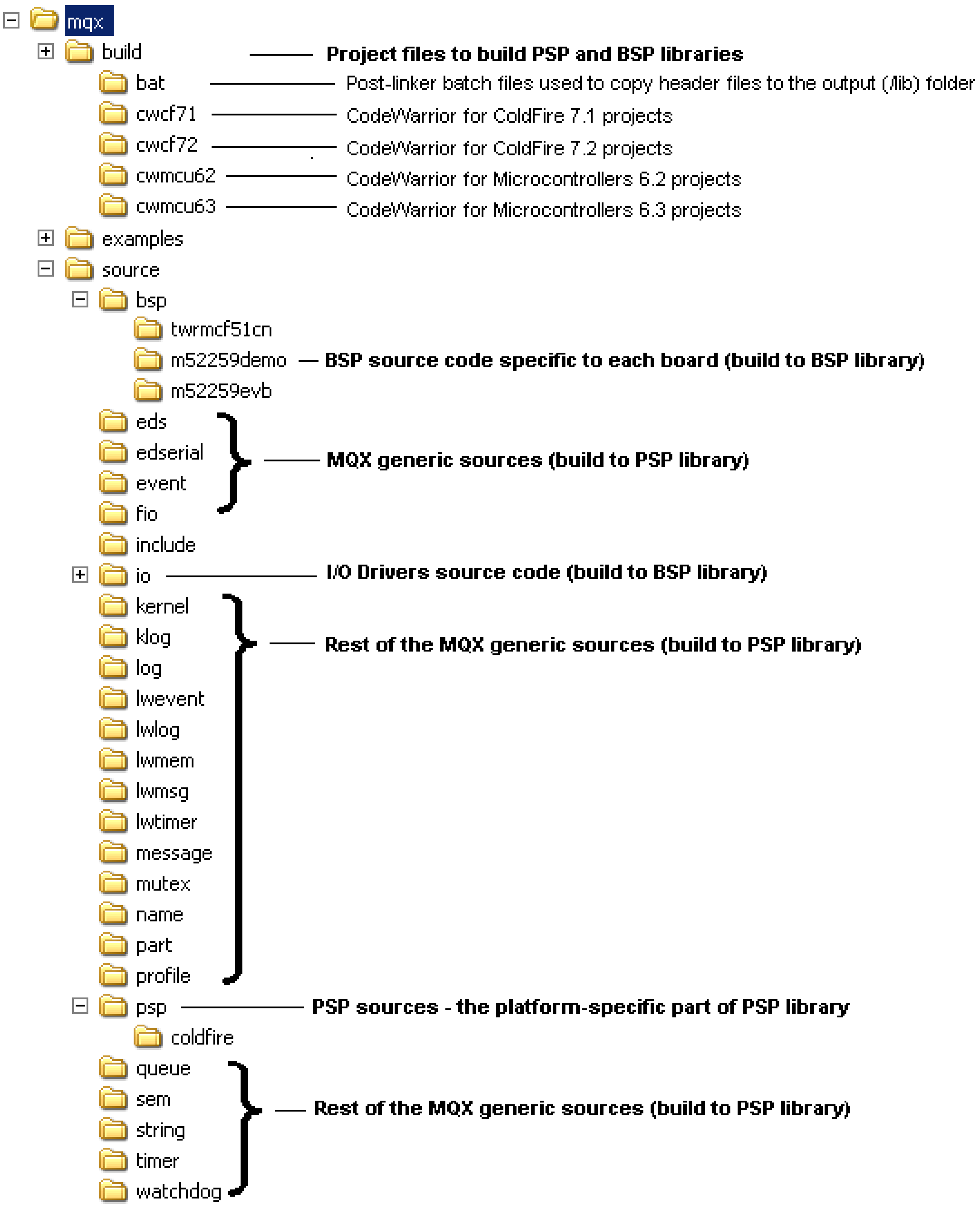
One of the I/O communication devices installed by MQX BSP may be used as the default serial console. By default, the console is used with the serial (uart) "tty" device.

The Freescale Tower-concept processor cards offer several ways to implement serial console port. Typically, you can use the "elevator" bus cards to connect the processor board with TWR-SER, TWR-SER2 or other kind of communication card. Some of the Tower cards provide also a combined debugging and communication USB port (OSBDM/OSJTAG) directly on the board. On the microcontroller side, the communication interface is connected to one of available SCI (tty) ports. On the PC-Host side you can use either a virtual serial port driver or a special USB communication terminal application. Refer to the development board documentation for more details.

The console port is selected by defining BSP_DEFAULT_IO_CHANNEL macro in the `user_config.h` configuration file or in the BSP-specific header file (see example in `mqx\source\bsp\m52259evb\m52259evb.h` file).

4.2.3 MQX PSP and BSP Directory Structure

The RTOS files are located in the `mqx` subdirectory of the Freescale MQX™ RTOS installation. The directory structure is briefly described in the picture below.



```
⊟ mqx
  ⊞ build          ——— Project files to build PSP and BSP libraries
      bat          ——— Post-linker batch files used to copy header files to the output (/lib) folder
      cwcf71       ——— CodeWarrior for ColdFire 7.1 projects
      cwcf72       ——— CodeWarrior for ColdFire 7.2 projects
      cwmcu62      ——— CodeWarrior for Microcontrollers 6.2 projects
      cwmcu63      ——— CodeWarrior for Microcontrollers 6.3 projects
  ⊞ examples
  ⊟ source
    ⊟ bsp
        twrmcf51cn
        m52259demo — BSP source code specific to each board (build to BSP library)
        m52259evb
      eds      }
      edserial }
      event    } ——— MQX generic sources (build to PSP library)
      fio      }
      include
    ⊞ io       ——— I/O Drivers source code (build to BSP library)
      kernel   }
      klog     } ——— Rest of the MQX generic sources (build to PSP library)
      log      }
      lwevent  }
      lwlog    }
      lwmem    }
      lwmsg    }
      lwtimer  }
      message  }
      mutex    }
      name     }
      part     }
      profile  }
    ⊟ psp      ——— PSP sources - the platform-specific part of PSP library
        coldfire
      queue    }
      sem      } ——— Rest of the MQX generic sources (build to PSP library)
      string   }
      timer    }
      watchdog }
```

4.3 MQX MFS

The Freescale MQX™ RTOS includes the MFS File System libraries. The MFS is based on the version 2.20 released by ARC.

The MFS files from `/mfs/source` directory are built with into a static library. When linked to user application the MFS library enables the application to access FAT12, FAT16 or FAT32 formatted drives.

4.4 MQX RTCS

The Freescale MQX™ RTOS includes the RTCS TCP/IP stack libraries. The RTCS is based and is API-level compatible with version 2.97 released by ARC.

The RTCS files from `/rtcs/source` directory are built with into a static library. When linked to user application the RTCS library enables the application to provide and consume network services of the TCP/IP protocol family.

4.5 MQX USB Host

This release of Freescale MQX includes the USB Host drivers and USB class drivers. The USB code is based on version 1.2.0 released by ARC.

The USB HDK (Host Development Kit) files from `/usb/host/source` directory are built with into a static library. When linked to user application the USB HDK library enables the application to communicate with various USB devices attached on the USB bus.

The HDK contains the following USB class drivers:

- USB Hub class used to attach multiple devices to a single host port. If enabled at the application level, the HUB support is fully transparent. User application only needs to be modified to be ready to handle multiple USB devices simultaneously. Keyboard/Mouse example application is provided.
- Human-interface Class (HID) used to access mouse, keyboard and similar devices
- Mass storage device (MSD) Class used to access USB drives
- Communication Device Class (CDC) used as a serial communication device implementing virtual “tty” ports
- PHDC for medical applications
- Basic Printer class

4.6 MQX USB Device

This release of Freescale MQX includes the USB Device drivers and example applications implementing various USB devices.

The USB DDK (Device Development Kit) files from `/usb/device/source` directory are built into the static library. When linked to the user application the USB DDK library enables the application to act as a USB device supporting one or more of the following classes:

- HID (mouse functionality demonstrated)
- MSD (internal RAM area accessed as mass storage device)
- CDC COM (virtual serial line implementation)
- CDC NIC (virtual network interface card implementation)

- PHDC (medical applications)

4.7 MQX Shell

The shell and command-line handling code is implemented as a separate library called Shell.

4.8 Changing the MQX Source Files

The Freescale MQX RTOS is distributed in the source code form. However, unless you are advanced in C programming and MQX kernel knowledge, it is recommended NOT to modify any of the source files other than compile-time configuration files. This recommendation applies to all files under “source” and “build” sub-directories in all MQX, RTCS, MFS, USB and other core components folders.

Only in case of creating custom board support packages or adding additional I/O drivers, there are two directories where the new files and subdirectories may need to be added:

```
<install_dir>/mqx/source/bsp
<install_dir>/mqx/source/io
```

4.9 Building the MQX Libraries

For more details about building MQX libraries and applications with the CodeWarrior tools see “MQX Getting Started” document.

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:

4.10 Example Applications

There are example „Lab“ applications in the directory:

```
<install_dir>/demo
```

The examples are accompanied with Lab guide documents – describing step-by-step how to run them on the target board. The examples were written to demonstrate the most frequently used features of the Freescale MQX™ RTOS.

In addition to these demo applications, there are simpler example applications available in MQX, RTCS, MFS and USB directories.

The following tables summarize all demo and example applications provided in this release.

MQX Example Applications
`mqx/examples/...`

Name	Description
adc	Shows usage of the ADC driver, sampling analog values from two ADC channels.
can/flexcan	Shows usage of FlexCAN API functions to transmit and receive CAN frames.
can/mscan	Shows usage of msCAN API functions to transmit and receive CAN frames.
can/mscan_loopback	The msCAN API driver demonstration in loopback mode.
demo	Shows MQX multitasking and inter-process communication using standard objects like semaphores, events or messages. See lwdemo for the same example using the lightweight objects.

`event`	Simple demonstration of MQX events.
`flashx`	Demonstration of FlashX driver functionality.
`gpio`	Shows usage of GPIO driver to control on-board LEDs and switches.
`hello`	A trivial Hello World application using a single task.
`hello2`	A trivial Hello World application spread across two tasks.
`i2c`	Shows how to read/write data from/to external SPI EEPROM. Additional HW setup is needed.
`io`	Shows using an alternate UART port as a console output.
`ipc`	UART-based interprocessor communication demonstration.
`isr`	Shows how to install interrupt service routine and how to chain it with the previous handler.
`klog`	Shows kernel events being logged and later the log entries dumped on console.
`log`	Shows application-specific logging feature.
`lwdemo`	Same as the "demo" application, but implemented using lightweight components only.
`lwevent`	Simple demonstration of MQX lightweight events.
`lwlog`	Simple demonstration of MQX lightweight log feature.
`lwmsgq`	Simple demonstration of MQX lightweight inter-process messaging.
`lwsem`	Simple demonstration of MQX task synchronization using the lightweight semaphore object.
`msg`	Simple demonstration of MQX inter-process message passing.
`mutex`	Simple demonstration of MQX task synchronization using the mutex object.
`nill`	Yet simpler than Hello World. A void application which may be used for copy/paste to start custom application.
`rtc`	Shows the Real Time Clock module API. Demonstrates how to synchronize RTC and MQX time and how to use RTC alarm interrupts.
`sem`	Simple demonstration of MQX task synchronization using the semaphore object.
`spi`	Shows how to read/write data from/to external SPI EEPROM. Additional HW setup is needed.
`taskat`	Shows how task can be created within statically allocated memory buffer (avoid heap allocation for task stack and context).
`taskq`	Shows custom task queue and how the queue can be suspended and resumed.
`test`	Shows the self-testing feature of each MQX component.
`tfs`	Shows the usage of ROM-based Trivial File System in an MQX application.
`timer`	Simple demonstration of MQX timer component.
`watchdog`	Simple demonstration of MQX task timeout detection using the watchdog component.

RTCS Example Applications

`rtcs/examples/...`

Name	Description
`httpsrv`	Simple web server with cgi scripts and few web pages stored in internal flash.
`ipc_udp`	UDP-based interprocessor communication demonstration.
`shell`	Shell command line providing commands for network management.
`snmp`	SNMP protocol example providing microprocessor state information.

MFS Example Applications

`mfs/examples/...`

Name	Description
`mfs_ftp`	RTCS FTP demo accessing MFS filesystem mounted on the USB mass storage. For FTP example without USB functionality, refer to RTCS Shell demo.
`mfs_usb`	Console shell-based example showing how to access MFS filesystem mounted on the USB mass storage.
`cfcard`	Console shell-based example showing the MFS filesystem used with and CFCard storage.
`ramdisk`	Shows use of MFS accessing the external RAM (or MRAM).

`sdcard`	Shows use of MFS accessing the SPI-connect SD Card.

USB Host Example Applications

`usb/host/examples/...`

Name	Description
`hid/keyboard`	This application echoes keys pressed on the USB keyboard onto serial console.
`hid/mouse`	Displays USB mouse events on serial console.
`hid/keyboard+mouse`	Keyboard and mouse demos combined in a single application.
`msd/msd_commands`	Executes the standard "mass storage device" commands to the USB disk and shows the response on the serial console (see mfs demos for USB filesystem access).

USB Device Example Applications

`usb/device/examples/...`

Name	Description
`msd/disk`	Implements small storage device in internal RAM memory.
`hid/mouse`	Creates virtual mouse which keeps moving in square loop, 100 pixels in size.
`cdc/virtual_com`	Implements a virtual serial line loopback.
`cdc/virtual_nic`	Implements a virtual network interface cards.
`phdc/bridge`	Ethernet to PHDC bridge. Receives Medical Data Exchange packets or APDU's from the Continua Agent Device on the Ethernet link as passes it to the USB Continua Manager. See http://www.continuaalliance.org.
`phdc/weighscale`	Implementation of IEEE11073 Data Exchange protocol with Weigh Scale specialization device.

Lab Tutorial Demos

`demo/...`

Name	Description
`hvac`	Simple implementation of console-based HVAC, with optional USB logging and FTP access.
`hvac_error`	Intentional error injected to the HVAC demo code to demonstrate power of TAD plug-in.
`web_hvac`	The HVAC demo with HTTP server implementing the GUI. Ajax-based pages demonstrate an advanced use of HTTP server.
`telnet_to_serial`	Simple character passing between UART console and telnet session. Shows custom "lightweight" telnet server.
`security_email`	Small-footprint application originally developed for MCF51CN. Example of simple SMTP client, able to send email message upon activity detected on GPIO input pins.
`security_telnet`	Telnet-based security monitor, displaying status of GPIO and ADC values.
`security_webserver`	WEB sever-based security monitor, displaying status of GPIO and ADC values.
`digital_sign`	Display driver demonstration with built-in FTP and WEB sever functionality.

5 What is Missing?

The MQX development teams in Freescale are working on continuous improvement of the MQX code, bringing new features as well as enabling and validating legacy MQX features with the latest supported devices.

This section provides a list of legacy MQX features which were removed and are not available in the current release of Freescale MQX™ RTOS. Features may either be removed temporarily, until the code is re-tested and validated with the latest supported devices; or permanently when the feature is considered obsolete.

In MQX PSP

- MMU and Virtual Memory Handling - Memory Management Unit and Virtual Memory are not available on devices currently supported by Freescale MQX™ RTOS.

In USB Support

- OTG Support - The USB OTG Drivers which were supported by ARC are not included in the current release of Freescale MQX™ RTOS.

PC Host Tools

- Legacy Host Tools - PC Host tools known from previous MQX releases (Design tool, EDS client tool, Performance tool) are not included in the current release of Freescale MQX™ RTOS.
  New in CodeWarrior version 10, there is a MQX Performance Data analyzer plug-in which brings some of features of the legacy MQX Performance tool. This plug-in is able to display kernel log dump and task execution history graph.

6 Known Issues and Limitations

6.1 Performance of Code Running in MRAM

The runtime performance of the MRAM based targets is approximately 8x degraded comparing to Flash-based execution. MRAM is an external memory device connected to the ColdFire core by 8-bit data bus and having one wait-state generated for each access. In order to fetch one 32-bit value (instruction) from the MRAM memory, four accesses need to occur – each inserting one wait-state clock.

This behavior is normal by design and applies to other processors using the external memory to store executable code.

6.2 Network Communication Performance and Delayed ACKs

Significant differences in performance of host to target TCP communication can be observed, depending on which client operating system is used. These differences are caused by implementation of the TCP delayed ACK algorithm.

Linux is more aggressive in acknowledging packets, and temporarily disables the delayed ACK algorithm on connection startup. As a result, there is no performance issue observed with Linux-based browsers (Mozilla Firefox tested).

Microsoft Windows based operating systems take a common approach to implementing the delayed ACK algorithm. As a result, web server performance can be significantly worse when it is accessed from Microsoft Windows-based browsers (Internet Explorer and Mozilla Firefox tested). This issue and a workaround are described at http://support.microsoft.com/kb/328890.

This issue is not directly related to the RTCS stack, however limited RAM resources (e.g. 64KB on MCF52259) imply lower buffer and data window sizes, making it more difficult to work around the issue in RTCS software.

6.3 Default Kernel Configuration of Small-RAM Devices

The default kernel configuration of small-RAM devices is optimized in order to run the demonstration applications located in the `/demo` folder. To meet tight RAM constraints, some of MQX or RTCS features are disabled by default. Also some of the I/O drivers not used by the main demo applications are disabled.

You typically get a compile-time error message if you try to run an example application while the required kernel feature or I/O driver is missing in the library code. In order to execute some of the example applications, you first need to enable the required features in the `/config/<board>/user_config.h` file and recompile all MQX libraries.

6.4 USB Host HUB Examples

HUB class support is enabled in HID example applications. The applications run correctly with the USB device attached both directly, or through the hub. However, the example code is still ready to handle a single device only. Combined Mouse+Keyboard demo is able to handle one mouse and one keyboard simultaneously. Using multiple devices of the same kind attached through the hub is not supported by the example applications.

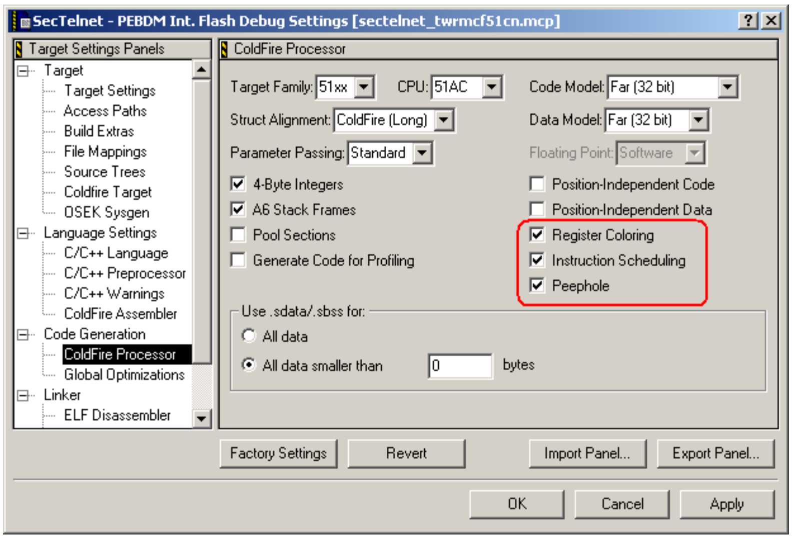
6.5 USB Host HUB Functionality with MCF5222x

The HUB interfacing is problematic with the older MCF52223 device. The issue is addressed by chip errata document.

6.6 ColdFire V1 Code Footprint in Debug Build Targets

This release of Freescale MQX™ RTOS brings the first version of the RTCS TCP/IP stack for ColdFire V1 devices. Even with the smallest device supported by this MQX release (MCF51CN128), the applications are provided to demonstrate HTTP, DHCP, ICMP and SNTP protocols over the TCP and UDP. With some of these applications, the size of executable compiled without size optimizations exceeds the Flash memory available (128KB on MCF51CN128).

As a workaround, size optimizations were enabled in the “Debug” build targets of the MQX, RTCS and other libraries for the ColdFire V1 processors. All optimizations remain disabled in “Debug” build target of the application projects itself so the application code debugging is not affected.



Target Family: 51xx
CPU: 51AC
Code Model: Far (32 bit)
Struct Alignment: ColdFire (Long)
Data Model: Far (32 bit)
Parameter Passing: Standard
Floating Point: Software

☑ 4-Byte Integers
☑ A6 Stack Frames
☐ Pool Sections
☐ Generate Code for Profiling
☐ Position-Independent Code
☐ Position-Independent Data
☑ Register Coloring
☑ Instruction Scheduling
☑ Peephole

Use .sdata/.sbss for:
○ All data
● All data smaller than 0 bytes

6.7 OSBDM / OSJTAG Firmware Compatibility

The latest versions of the CodeWarrior and IAR tools bring a new version of the OSBDM / OSJTAG Debugger interface with improved performance and stability. This new interface requires the firmware (the HCS08JM60 code) to be updated to the latest version. Refer to instructions provided in the Release Notes documents coming along with the development tools you are using.

6.8 Supporting “Hot Device Uninstall” in MQX I/O Subsystem

With today’s implementation of MQX I/O subsystem, special attention is needed to uninstall a device driver while there is one or more device files open at the application layer. In other words, the application is responsible for dealing with application tasks that have opened file handles when a device driver is being uninstalled.

A typical demonstration of the problem is USB mass storage handling:

- When USB attach event is detected, an application installs the MFS partition manager and MFS filesystem ‘device’ on top of the USB driver.

- The application runs tasks (e.g. shell) which open and access files provided by the MFS filesystem device.
- When user unplugs the USB mass storage device, the application has only a limited way to detect an opened file exists before uninstalling the MFS filesystem device.
- The file I/O functions begin to report errors when accessing the device after it is physically detached. The application code should be designed in a way that the tasks close all files affected by the detach event before the MFS filesystem driver can be uninstalled.
- When MFS filesystem device is uninstalled while open files are still accessed from other tasks, an unhandled exception may occur.

We recognize that this implementation may add additional application overhead. We are enhancing MQX I/O subsystem so the file operations will safely return error states even after the underlying device driver is uninstalled. This enhancement will simplify the application code error recovery.

6.9 Spurious Interrupts on ColdFire Platforms

MQX users reported accidental occurrence of spurious interrupts. Spurious interrupt occurs when interrupt source is being disabled by software while the interrupt flag is being set by hardware. As a consequence, interrupt controller looses the information about what was the source of the interrupt.

ColdFire processor documentation reccomends to temporarily disable all interrupts each time any interrupt source is being masked in interrupt controller or in peripheral module register.

User application or custom user drivers need to follow this procedure to avoid spurious interrupts to occur. The interrupt-driven I2C I/O driver was updated accordingly in MQX 3.6. Updating other I/O drivers remains a task for future MQX versions.

6.10 CodeWarrior version 10.1

The MQX 3.7 newly includes support for CodeWarrior Development Studio version 10.1. Several issues were observed during MQX porting and will be addressed in follow-up releases of both MQX and CodeWarrior tools:

- The CW10.1 ARM compiler and ColdFire compilers patches are required. (MQX RTOS is unstable if patches are not used – especially USB and RTCS features are affected)
- OSBDM / OSJTAG debugger looses connection with target when CPU executes the “stop” instruction. This happens in MQX when all tasks are blocked waiting for event occurrence or time interval to elapse. When kernel dispatcher does not find any task to be set active, it executes the “stop” instruction and suspends the CPU until next interrupt. With OSBDM-connected boards, avoid this situation by enabling the idle task. Set MQX_USE_IDLE_TASK to 1 in the user_config.h file.

6.11 TWR-MEM Compact Flash interface issues

Some Compact Flash cards does not work correctly with TWR-MEM and MQX CF card driver. The reasons could be following:

- 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.
- In some cases MQX driver incorrectly detects the card in the slot. First check if all CF related jumpers are set correctly according to the Getting Started document. If you still experience

incorrect behavior, connect two pull-up resistors between card detect pins (CF_CD1, CF_CD2) and 3.3V VCC.

6.12 Kinetis Processor Support Package issues

This MQX version brings the support for the Freescale ARM® CortexM Kinetis platform and Freescale MK40 and MK60 microcontrollers. Some of the MQX PSP and BSP features are unavailable in the current release. This limitation is subject to change for future versions.

- The MQX kernel cannot operate without an idle task. The MQX_USE_IDLE_TASK configuration options must be set to 1.
- The CF Card driver (to be used with TWR-MEM card) is not implemented for Kinetis processors yet
- TWR-K60N512 REV C board issue: 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.

6.13 IAR Embedded Workbench limitations

There is known limitation related to debugging comfort in the Embedded Workbench. The call-stack information is not displayed for any code which has assembler-coded routines in its call history. And because of the MQX kernel code itself is all coded in assembler, all tasks running in the MQX OS are affected by this limitation.

To overcome this issue, the assembler-coded routines need to be annotated by special CFI directives. This remains a priority for the future MQX releases.

6.14 Prinf floating point (%f) does not work on IAR and Kinetis

The problem appears in IAR EWARM 6.10. The IAR compiler requires the Stack Pointer to be aligned at 8 bytes. Stack alignment is not ensured in current MQX. The behavior will be corrected in future MQX versions.

6.15 USB EHCI stack cannot handle cached memory

User application buffers used by USB EHCI Class drivers have to reside in un-cached memory space. This issue will be addressed in future MQX version by introducing double buffering scheme.

6.16 FlashX driver cannot read data across different flash areas

The FlashX driver allows to handle several independent Flash memory block by one instance of FlashX driver. All areas are connected into the one compact logical address space. However reading data across multiple blocks is not possible. A workaround is to read from within single memory block, then seek to another and read the rest of required data. The seek() and write() functionality works correctly.

7 Change Log

Version 3.0.0 (December 5th 2008)

- This is initial release supporting the MCF52259 processor, M52259EVB and M52259DEMO boards.

Version 3.0.1 (January 22nd 2009)

- Small enhancements throughout the whole code base of MQX kernel and other components
    - Memory block “type” information added to all system memory allocations. TAD is now able to give detailed information about each memory block allocated by kernel or system component.
    - Dedicated memory allocation routines in RTCS, MFS and USB simplify the memory pool usage.
- IPCFG ethernet link monitoring features and automatic IP address binding functionality was added to RTCS. The “ipconfig” shell command replaces the old “bind”, “ifbind” and “dhcp” shell commands.
- The HTTP server in RTCS was re-written to enable multiple sessions to be served by a single task. This feature brings more reliable HTTP server while maintaining low memory requirements.
- USB Host HUB class was added.
- MQX User Guide and several other documents are included in the setup package.

Version 3.1.0 (April 3rd 2009)

- USB Device low-level driver has been implemented for MCF522xx family and example applications were created for M52259EVB and M52223EVB.
- The “usb” subdirectory of the /lib output folder was split to “host” and “device” parts.
- Added PSP, BSPs and other support files for MCF52223 and MCF52235 evaluation boards:
    - USB Host and Device libraries were ported to M52223EVB
    - RTCS library was ported to M52235EVB
    - MFS and Shell libraries ported to both new boards
- USB Host HUB issue resolved with MCF52259 (excessive number of errors observed when USB devices were accessed through USB HUB). The issue was solved by implementation of SOF frame scheduler in the USB Host low-level driver. An issue with a similar impact remains on MCF5222x implementation (silicon problem, no workaround known).
- Error codes naming and numbering convention has been made more consistent across the MQX, RTCS and other libraries. Old error code names remain implemented for backward compatibility (numeric values changed).
- Kernel Data and other internal structures were optimized for size. Parts of the structures were made conditionally compiled based on the user configuration.
- TAD (CodeWarrior debugger plugin) was updated to handle renaming in internal structures.

- Memory type information was added also to lightweight memory structures. TAD is now able to display the type information for memory blocks allocated using lightweight memory allocator.
- Lightweight memory allocator has been made the default option for all BSPs.
- The `/config/<board>/user_config.cw` files were eliminated. Such files have contained a subset of configuration options for assembler-coded files in the MQX kernel. All kernel assembler-coded files are now preprocessed using C preprocessor and make use of standard C header macros. The C-language-specific content of the header files is conditionally excluded during assembler compilation by using `__ASM__` macro.
- The assembler `vectors.cw` file in each BSP was re-coded to C syntax. A new user configuration macro (`MQX_ROM_VECTORS`) may be set non-zero to avoid vector table being copied to RAM.
- The BSP, PSP, RTCS and other library build projects were changed to be easier to understand (virtual folders inside each CodeWarrior project was updated).
- SNMPv2 code was changed to enable ROM-based MIB structures. Example application is provided demonstrating custom MIB nodes and user traps.
- Several source code files were renamed to better reflect the content, especially in the I/O driver directories.
- Added “root directory” concept to the FTP server, “rename” command added.
- SPI I/O driver added (Master mode only). Example application is available.
- Initial ADC I/O driver was added. This driver (including its API) is subject to improvement in future releases. ADC device driver usage is demonstrated in ADC example application.
- RTC API updated, example application added.
- The new IPCFG API updated and documented
- The new HTTP server API documented
- MQX I/O Driver User Guide added

Version 3.2.0 (May 15th 2009)

- Added PSP, BSPs and other support files for TWR-MCF51CN-KIT Tower Kit with ColdFire V1 processor MCF51CN128.
    - RTCS library was also ported to MCF51CN128 device.
- Ethernet driver significantly re-written and generalized to cover both ColdFire V1 and V2 processors. Application is now able to change some of the driver parameters dynamically, without need to recompile the driver code.
- Most of the features missing in the previous release of the ADC driver were implemented. The driver documentation is still on the to-do list.
- ADC driver partially re-written and low-level layer ported to ADC module of MCF51CN device. There are implementation differences between MCF51CNxx and MCF522xx drivers which are still to be addressed by future releases.
    - MCF51CN driver uses internal lightweight timer component
    - MCF522xx driver still uses PIT timer for internal timing. Implementation with lightweight timer (lower performance) is planned for future versions

- New GPIO callback-on-interrupt feature was added to GPIO driver of MCF51CN. This feature will be ported to other supported devices in future releases.
- Three new “security” demo applications were created for MCF51CN device demonstrating the ADC and GPIO drivers, network protocols and low-power mode of operation. The applications will be ported to other devices in future.

Version 3.2.1 (May 28th 2009)

- Sparse interrupt table implemented and made a default option for small-RAM devices. This feature saves up to 1k of RAM. The TAD CodeWarrior plug-in was updated to support this feature.
- The legacy MQX Flash driver (called FlashX) was ported to support internal Flash memory of all supported devices (MCF51CNxxx, MCF522xx). The Flash driver requires Flash parameters to be passed from the application or linker command file. The linker command files were updated in all BSP. Sample application is provided.
- The GPIO driver callback-on-interrupt feature was ported to MCF52xx devices. This feature is still a subject to change in future versions. The MCF51xx driver was optimized for code size.
- Minor bug fixes in Serial driver (fflush system call implemented).
- MCF51CNxx SPI example enhanced to support both SPI EEPROM and SPI Flash devices. The example applications may be used with Memory Storey board.
- Fixed known issue with MQX Stationery as it was described in the last release notes. The MQX_PATH named source tree is no longer used for referencing the debugger initialization file (this was causing Flash Programmer crash). Copies of debugger initialization and memory configuration files were made part of all stationery projects and are referenced using relative path only in the debugger settings.
- I/O Driver User Guide updated. The GPIO, ADC, SPI and FlashX driver description added.

Version 3.3.0 (Aug 7th 2009)

- Added BSP for MCF52259 Tower Kit.
  - CodeWarrior projects, demos and example applications were ported from M52259EVB to this BSP.
  - CodeWarrior projects and the BSP also support debugging targets and running code from external MRAM memory. This is prepared for future support of Memory Storey board for the Tower Kit.
- Added PSP, BSPs and other support files for M54455EVB ColdFire V4 system. This is the first V4 device supported by Freescale MQX operating system. Few new features were added to PSP:
  - Cache control support added. Non-cached memory pool allocation API added.
  - Flash-to-RAM code copying enabled in startup.
  - Startup code is now part of PSP. Two files (`CF_startup.c` and `ROMCopy.c`) were reused from CodeWarrior runtime library. These files were modified to support ROM-to-RAM code copying.
  - Flash Programmer config files available in `tools/flash_programmer_scripts/config` directory.

    - SPI driver for M54455EVB created (the DSPI module), example supports onboard SPI Flash
- Ethernet driver and RTCS were significantly re-written.
    - Device-independent part of the driver was re-written to support multiple Ethernet MAC devices of the same or even different kinds. A support for memory-optimized handling of small frames was also added.
    - Device-dependent part of the Ethernet driver was re-written to support ColdFire V1-V4 Fast Ethernet Controller module.
    - Several RTCS features were re-tested and fixed to support multiple physical controllers.
- RTCS updated
    - The rtcs_shell example now uses the MFS lib (with RAM-disk) and supports all FTP commands (conditional compilation)
    - Minor changes in IPCFG API to support multiple devices, potentially of different kind than ENET.
- TAD updated
    - Strings displayed in TAD moved to separate text files available inside MQX installation.
    - New Ethernet driver screen added.
    - Symbolic reader enhanced some objects (LWsem, LWevent, ...) are displayed not only with address information, but also with symbolic names.
- Several key TAD-like screen dumps implemented also on embedded application side in PSP. These functions may be used to print out the TAD-like debugging data by the application.
- Shell library updated
    - RTCS support commands changed (ipconfig, netstat).
    - MFS format command was made available (also see `mfs/examples/ramdisk` application)
    - Shell interface to TAD-like screens dumps added.
- Workaround for MCF5223x Ethernet PHY auto-negotiation issue implemented according to chip errata.
- FlashX driver updated:
    - Support of external Flash devices made available. Tested with external flash memory devices of M54455EVB.
    - FlashX example application modified to support external flash1 device on M54455EVB.
    - Write protect ioctl command implemented for external Flash devices.
- USB Host updated:
    - Code refactored and generally updated (MASS storage class renamed to MSD).
    - CDC class added. Example application shows how to forward characters between standard UART TTY and a virtual USB TTY device.
- I2C driver refactored and updated, eeprom example changed.

- CodeWarrior “Stationery” templates for creating new MQX projects were changed:
    - All kinds of stationery projects now have full set of libraries added to project (BSP, PSP, MFS, RTCS, USB Host, USB Device, Shell). The standard CodeWarrior linker will optimize out the unreferenced code. Feel free to remove the unused libraries from the project.
    - MQX-Only stationery is a simple “hello-world” like example.
    - MQX+MFS stationery is simple shell example working with RAM disk.
    - MQX+RTCS stationery is simple shell example with basic networking commands and telnet server.
    - MQX+RTCS+MFS stationery is a union of the two examples above.
    - USB Host is a simple application able to detect devices attached to the host port or devices attached through USB hub. You still need to refer to the USB host example applications to see how to interface attached devices.
    - The USB Device stationery is not available in this release. Please refer to the USB Device example applications.
- Device-specific sections added to this release notes document. Please read the required jumper settings for each evaluation board supported.

Version 3.4.0 (Sep 25th 2009)

- Support for Register ABI (register parameter passing) was implemented
    - RegABI build targets (for both Release and Debug configurations) added in all library build projects. The binary libraries compiled with RegABI configuration get the “_regabi” postfix.
    - Former build targets which use the Standard ABI are still maintained in the library build projects for backward compatibility. The targets were renamed to “StdABI”.
    - All example and stationery applications were reconfigured to use Register ABI and RegABI MQX libraries.
- M52277EVB BSP added
- M52233DEMO BSP added
- USB EHCI Host Support implemented
    - USB Host functionality enabled on MCF52277 and MCF54455.
- USB Device Stack reworked and enhanced
    - The USB Device Stack code has been partially rewritten to be consistent with the similar bare-metal stack available for Freescale HCS08 platform (released separately).
    - CDC class implementation examples added (virtual serial line and virtual network interface card).
    - PHDC medical class implementation examples added.
- SPI driver reworked to support all kinds of ColdFire SPI modules (SPI, QSPI and DSPI)
    - Former QSPI driver was removed from the release.
- IPC inter-processor communication files made available in the release.

    - IPC Example applications provided.
    - SCI-based IPC tested only.
- SPI-based SD Card Driver added
    - The driver was tested with Memory storey for Tower Kit only (MCF51CN and MCF52259 devices). Not tested with M52277EVB board.
- Bug fixed: Wrong SCI baud rate divisors calculation fixed.
- Bug fixed: Shell “dir” command file attribute filter is now applied correctly.
- Bug fixed: MFS read and write calls correctly return negative value when physical device returns access error. Please see also another known issue described in section 6.8 “Supporting “Hot Device Uninstall” in MQX I/O Subsystem”.

Version 3.5.0 (Jan 26th 2010)

- CodeWarrior build projects location was changed. The `build/codewarrior` subdirectory was replaced by several directories depending on type and version of the CodeWarrior tool.
    - `build/cwmcu62` contains build projects for CodeWarrior Development Studio for Microcontrollers version 6.2 (ColdFire V1 projects from MQX 3.4 and earlier)
    - `build/cwmcu63` contains build projects for CodeWarrior Development Studio for Microcontrollers version 6.3 (New versions of ColdFire V1 projects)
    - `build/cwcf71` contains build projects for CodeWarrior Development Studio for ColdFire Architectures version 7.1 (ColdFire V2-V4 projects from MQX 3.4 and earlier)
    - `build/cwcf72` now contains build projects for CodeWarrior Development Studio for ColdFire Architectures version 7.2 (New versions of ColdFire V2-V4 projects)
    - The post-linker batch files were separated to `build/bat` out of the build folders to be able to reuse them between different build project versions.
    - Mass-build projects of all libraries were also moved from `config/<board>` folder to an appropriate subdirectory.
    - Build projects for the later CodeWarrior versions (`cwcf72` and `cwmcu63)` no longer support standard (on-stack) parameter passing. StdABI targets are not available in build projects.
- The code and folder structure was updated to make it ready for IAR toolset support
    - CodeWarrior-specific C and Assembler syntax was changed to be compatible with IAR compilers. Only few parts of code made conditionally compiled depending on the compiler.
    - CodeWarrior-specific portions of BSP (start-up code, memory initialization code, debugger configuration files) were moved to `cw` subfolder. The `iar` subfolder will be added by the IAR support patch in future.
- Changes in PSP
    - CPU numbering scheme was changed. The CPU is still defined in the user_config.h file, but uses constants pre-defined in `psp_cpudef.h` file.
    - Several new code-size compile-time configuration options were added. See updated MQX User‘s Guide documentation.

- Changes in MFS
  - Dynamic read-only status checking of the physical device was added to MFS write functions as optional feature (MFSCFG_READ_ONLY_CHECK). This option is useful for example with SD Card storage with write-protect switch.
- RTCS stack updated
  - Set of RTCS iwcfg_*xxx*() functions was implemented to enable application-level control of any future WiFi functionality.
  - ARP resend and expiration timeout values were made configurable through the user_config.h file. Especially the “expire incomplete” time needs attention as it affects the total time a send call takes when target IP address does not exist on the local subnet.
  - Default setting for DHCP DISCOVER message was changed. Broadcast flag is set to FALSE to request unicast DHCP responses by default.
  - Internal structure names were renamed to keep consistency with C typedef names. The structure tag names are now always the lowercase variant of the C typedef name. This change enables proper ELF file format parsing by Task Aware Debugger plugin.
  - Bug fixed – RTCS_selectset function now behaves as in RTCS versions prior to 3.0. The socket array is not altered by this function.
  - RTCS_select call enhanced to support more than one socket in the array.
- Shell library was extended by new “iwconfig” command to enable the iwcfg_*xxx*() control from the Shell environment
- BSP and support for MCF51EM256-based DEMOEM board was added
  - SCI, SPI, I2C and other standard drivers were ported to this platform
  - ADC, RTC, FlashX were enhanced to support additional features the MCF51EM offers
- ENET Driver was updated
  - Generic WiFi support was added into driver code for future WiFi support.
  - ENET MAC interface structure was extended to support both generic and device-specific control commands (so-called media control commands).
  - Set of WiFi-related media control commands was defined (setting the ESSID, encryption, …)
  - ENET driver initialization structure was extended by new device-dependent initialization parameter.
  - ENET driver concept of device context locking was enhanced (from global interrupt disable to lwsem-based locking).
  - HVAC, WEB_HVAC, HTTPSRV and RTCS Shell example applications were made ready for WiFi functionality.
- SPI driver was updated
  - Support for SPI16 module of MCF51EM was added.
  - Driver code for 8-bit ColdFire V1 SPI module was internally renamed to SPI8.
  - The SPI driver now provides uniform I/O API to SPI8, SPI16, QSPI and DSPI modules.

- Bug fixed – QSPI chip select was incorrectly asserted during driver initialization making a glitch on this signal.

- GPIO driver was extended
    - Open flags were added to control polarity of edge which triggers interrupt callback routine to be called (rising/falling edge).
    - I/O control commands were added to enable/disable interrupt callback function to be called.
- ADC driver now supports the PDB trigger module and sixteen-bit ADC module available at MCF51EM.
    - ADC driver install function was changed to accept additional device-specific initialization structure.
- SD Card driver now supports the SD Card interface available on M52277EVB. SD Card Example application was added for this board.
- Example applications
    - SD Card example application now decodes GPIO inputs for card presence and read-only switch.
    - Code-size benchmark application was added for MCF52259EVB. The application makes artificial reference to MQX kernel API functions so the kernel code-size can be analyzed from map files. A tool automating the build, map file parsing and HTML report generation is also available.
    - Timing benchmark application was added for MCF51CN, MCF52259 and MCF54455 devices. This application measures key timing parameters of the MQX kernel. HTML report generator tool is also available.
    - RTC example application enhanced to support new features available in MCF51EM.
    - SPI example application extended from EEPROM testing to general SPI-based memory device (EEPROM, Flash or serial MRAM).

Version 3.5.1 (Feb 19th 2010)

- Fixed bugs:
    - MCF5225x ADC driver - pin initialization was not performed properly; problem appears with “Release” targets.
    - FlashX driver was not working correctly with internal Flash memory of MCF52xx processors.
- Example applications updated
    - MFS SD Card example application demonstrates use of Partition Manager now.
    - Code-size benchmark applications and results are available for M52259EVB, M54455EVB, TWR-MCF51CN.

Version 3.6.0 (Jun 21st 2010)

- The support for older CodeWarrior for ColdFire 7.1 and CodeWarrior for Microcontrollers 6.2 was removed. Build projects are available only for versions 7.2 and 6.3.

- Eclipse-based CodeWarrior 10 support is added. All library and application build projects are also available in the native CW10 format.
- MCF51AC-based DEMOAC Board Support Package added with basic set of peripheral drivers: ADC, FLASHX, GPIO, I2C, INT_CTRL, SERIAL, SPI.
- MCF51JM-based EVB51JM128 Board Support Package added with basic set of peripheral drivers: ADC, FLASHX, GPIO, I2C, INT_CTRL, SERIAL, SPI. A support for USB Host/Device stack is also available.
- MPC5125 BSP added. This is the first PowerPC port supported by Freescale MQX. This version includes the code released recently as MPC5125 patch to MQX 3.5.1.
    - Basic set of I/O drivers supporting the MPC5125 peripherals is included: PSC UART, Fast Ethernet controller, DIU, msCAN and RTC.
    - Example applications available: Standard set of MQX and RTCS examples, the “Digital Sign” demo application demonstrating the graphical DIU module of MPC5125 and the HDMI interface.
- The MQX_SUPPRESS_FILE_DEF and MQX_SUPPRESS_STDIO_MACROS may now be defined as 1 in user_config.h to avoid declaration of generic FILE, FILE_PTR types and <stdio.h> functions like read, write, printf etc.
    - The MQX code was refactored to use the new MQX_FILE and MQX_FILE_PTR types and make aliases to backward compatible FILE and FILE_PTR types only if not suppressed.
    - The key changes were implemented in the file fio.h in mqx/source/include.
    - Default values of both configuration options are defined such that backward compatibility is not affected.
- The user_config.h files and shared configuration files in /config/common directory were re-formatted and updated:
    - All BSPCFG options related to given board are now available in individual user_config.h files.
    - MQX_TASK_DESTRUCTION and MQX_EXIT_ENABLED options are now set to 0 in small_ram_config.h file.
    - MQX_COMPONENT_DESTRUCTION option was removed from the verif_enabled_config.h file. The default value of 1 defined in mqx_cnfg.h is used. The ColdFire V1 processors override this option to 0 in user_config.h to save code size.
- A new _lwevent_get_signalled() API call was added to enable detecting which lwevent bits caused the active task to return from _lwevent_wait() call.
- Inline assembler code was eliminated where possible to simplify migration to alternative C compilers.
    - MCF52xx and MCF51xx FlashX flashing routines were rewritten in C.
    - The “nop” instruction primitive _ASM_NOP() added to compiler-specific header file and used instead of asm(nop) inline assembler statement.
    - The “stop” instruction primitive _ASM_STOP(x) added to compiler-specific header file and used instead of asm(stop #(x)) inline assembler statement.
- Support for Freescale eGUI software and TWR-LCD display board has been added. See eGUI MQX demos within the Freescale eGUI package. The eGUI comes with set of low-level drivers for non-OS bare-metal operation. The drivers in MQX 3.6 enable the eGUI

package to be configured for hardware-independent “MQX” mode where all board-dependent code resides in the MQX BSP.

  - o Resistive Touch Screen driver added to TWR-MCF52259 and TWR-MCF51CN BSPs.
  - o LCD memory regions added to TWR-MCF52259 and TWR-MCF51CN linker command files in BSPs.
  - o GPIO pins, ADC channels and SPI channel specific for LCD and touch screen control are defined in TWR-MCF52259 and TWR-MCF51CN BSP header files.

- Inter-processor communication (IPC) over the UDP protocol was added together with an example application in RTCS.
- RTCS ipconfig shell command enhanced to be able to print immediate values of low-level PHY registers.
- Serial driver supports new open mode-flag (IO_SERIAL_NON_BLOCKING) to enable accessing the serial driver in non-blocking mode. Supported by both polled and interrupt serial drivers.
- New I2C ioctl command (IO_IOCTL_I2C_GET_BUS_AVAILABILITY) implemented to detect idle/busy condition on the bus.
- Register I/O map structures were updated to cover all supported parts and changes in I/O drivers. In addition to changes in processor specific header files, the following shared I/O map files have been changed:
  - o mcf5xxx_spi16.h ... SPI16 no-FIFO module added.
  - o mcf5xxx_usbotg.h ... ULPI control registers added.
  - o mcf522x_usbotg.h ... registers added
  - o mcf51xx_i2c.h ... I2C with/without SMB support defined
  - o mcf51xx_ftsr.h ... registers added
- Register I/O map structures of MCF54455 were expanded to cover peripherals not supported by MQX drivers (DMA timers, EPORT, PCI, RNG, SSI.
- The entire code was “untabified” (TAB characters replaced by spaces).
- Readme files created for selected example applications (GPIO, I2C, etc.).
- IPC demo application code generalized for various TTY ports availability.
- Codesize MAP-file analysis script now prints code-size statistics for RTCS, MFS and other system libraries.
- MQX I/O User Guide updated:
  - o SPI, I2C, ADC, RTC and SD Card driver sections updated.
  - o ESDHC, FlexCAN, DAC Driver and NAND Flash Driver sections added.
- Board-specific information related to MQX release has been moved from Release Notes into a separate document (see “MQX Getting Started” document).

Bugfixes in 3.6.0

- MQX Kernel:

- The _io_fclose() no longer locks the kernel's IO_LWSEM semaphore before calling the device driver's close function. This could cause a race condition with layered drivers.
- The EWL library linking is now disabled in all MQX library projects (PSP, BSP, MFS, RTCS, ...). Having this option enabled in "Librarian" projects settings panel was causing all EWL code to be included in each MQX library. This was effectively disabling EWL function overrides from MQX code. For example the malloc() code was taken from EWL instead of from MQX and was causing linking issues (malloc is also part of C++ *new* operator implementation).
- Handling of priority messages was fixed. In some cases a priority message could have been received from a wrong message queue.
- Format specifier parameter is declared as const in the sscanf prototype. This avoids compiler warning messages when passing constant string as the scan format.
- The MQX watchdog component (do not confuse this with the on-chip hardware watchdog) was fixed in _watchdog_isr() to expire at proper time on Coldfire V1 platforms. Potentially this also solves problems in case the watchdog uses other timer than the main BSP tick timer.

- BSPs and I/O drivers:
  - EPORT interrupt level macros in MCF522xx board-specific headers in BSP are now set correctly according to the levels hardwired in the controller device. Be aware that EPORT_EPF7 level is hardwired to non-maskable interrupt so no MQX calls may occur in the interrupt routine.
  - GPIO button pin identifiers (BSP_SW1 and BSP_SW2) were fixed in M52233DEMO BSP header.
  - ADC driver functions now return more detailed error codes instead of generic IO_ERROR value.
  - MCF522xx interrupt initialization and unmasking functions fixed. Problem was that the code was also inadvertently unmasking other peripheral interrupt sources in the interrupt controller.
  - TWR-MCF51CN tick timer interrupt handler was fixed in BSP. Due to wrong timer interrupt acknowledge code, the MQX system tick time was one half of the configured value.
  - MCF51EM-specific code was removed from generic MCF5xxx RTC driver code.
  - I2C clock source changed from BSP_BUS_CLOCK to BSP_SYSTEM_CLOCK on all MCF52259-based boards. This fixes the problem of I2C bitrate which was 2x higher than configured.
  - Uncached memory region definitions were fixed in M54455EVB linker command files in BSP.
  - Default PLL system clocks setup was changed in M52277EVB BSP startup code. The MCF52277 processor now runs at system clock 160MHz (80MHz bus clock, 60MHz USB clock).
  - USB I/O pin initialization routine was fixed in M52277EVB BSP. The problem with pin settings made USB Host functionality unusable on M52277EVB.
  - SPI chip select control optimized for performance.
  - MCF5xxx FEC driver initialization of 2nd Ethernet interface (ipconfig 1 init) is no longer dependent on having 1st interface initialized.

    - Compiling MCF5xxx FEC code generates error message if un-cached memory support is disabled in MQX configuration.
- MFS Filesystem:
    - The CLUSTER_SIZE_BYTES member of MFS_DRIVE_STRUCT is now extended to 32 bit to avoid size overflow in case of 65k cluster size (solves some USB memory stick problems).
    - Hidden attribute checking was fixed in MFS_Attribute_match() function.
    - MFS write/append performance problem was solved by avoiding unnecessary sector reading.
- SHELL Library:
    - Memory violation error fixed in the shell “help” command implementation.
- USB Host Stack:
    - USB Host HUB class enhanced to support asynchronous status inquires.
    - USB CDC Host driver now waits for control transactions to ACM interface to complete. This fixes a problem of invalid memory access with some CDC devices.
    - KHCI USB bus control is deactivated after device is detached. This avoids problems where noise on detached bus was evaluated as a valid communication.
    - KHCI USB data bus weak pull-down resistors enabled to improve detection of reset/detach event.
    - EHCI fixed to toggle data tokens correctly when a transfer has odd number of transactions.
- Example applications:
    - Digital sign demo running on TWR-MPC5125 now runs the display task with higher priority than the network task. This fixes a problem where images were displayed after 3 minutes ARP timeout when board was not connected to network.
    - Bar-graph CGI code in HTTP Server demo now reads ADC channels properly.
    - msCAN example application fixed on TWR-MPC5125.
- Task Aware Debugger plug-in for CodeWarrior
    - The value of SR register is now displayed correctly in context of any tasks being examined.
- Other:
    - New Version Checker tool was fixed so it always runs in a single instance. Each of the TAD DLLs (current version and all backward-compatibility versions) starts an instance of the web checker tool upon CodeWarrior start. Running the tool in multiple instances was causing problems with loading the *webchk.wcp* configuration file or resulted in multiple “New Version Available” notifications being displayed.

Version 3.6.1 (Aug 31st 2010)

- MCF51MM-based TWR-MCF51MM Board Support Package was added
    - Standard set of peripheral drivers is enabled: FLASHX, GPIO, I2C, INT_CTRL, UART, SPI, RTC, CF card, ADC, DAC, SD card
    - New driver was added: DAC (API driver).

  - o USB Host and Device stacks are available. The host (KHCI) driver can be configured to include software workaround for early silicon issue of the MCF51MM device. This workaround is disabled by default.

- MCF53015-based M53015EVB Board Support Package was added
  - o Standard set of peripheral drivers is enabled FLASHX, GPIO, I2C, INT_CTRL, SERIAL, SPI, SDHC, CF card, SD card.
  - o USB Host and Device stacks are available.
  - o Two Ethernet interfaces are enabled in RTCS TCP/IP.
- New SDHC I/O driver covering the eSDHC peripheral module was created. This driver allows low-level access to various cards such as SD, SDHC, SDIO, SDCOMBO, SDHCCOMBO, MMC and CE-ATA. MQX currently implements upper layer for SD and SDHC card drivers only.
- HTTP server listen address can now be configured in application and may be optionally set to other value than INADDR_ANY. This may help to run the web-server application on just one of several network interfaces.
- New TAD “Memory Blocks Summary” screen implemented for lightweight memory blocks. The screen shows memory allocation grouped by categories.
- The MQX documentation was updated:
  - o MFS User Guide chapter 3.1.9 was updated (search attributes).
  - o The lwevent description was updated in MQX Reference Manual.
  - o The description of _time_diff_xxx() function parameters corrected in MQX Reference Manual.
  - o The _RTCS_mem_pool setting documented in chapter 2.6 of RTCS User Guide.
  - o IO Guide SPI driver description was extended by new IOCTL command.
- Release notes and Getting started documents were reorganized and extended by more detailed description of the MQX build process.
- New communication mode was implemented in SPI driver for the QSPI module. In this mode, the communication is split to smaller chunks (16 bytes) to fully leverage hardware-driven CS signal activation. Also in this mode, the CSIV bit is not controlled by the driver flush() function which is normally causing glitch on the other QSPI CS signals.

Bugfixes

- The MFS functionality IO_IOCTL_FIND_FIRST_FILE was not working. In case the search attribute MFS_SEAR v2CH_NORMAL was used no files were found. Attribute matching algorithm was corrected and user manual was updated.
- MFS file write bug which appeared in MQX 3.6.0 append-file performance optimization was fixed. In case two or more files were opened simultaneously, wrong content could be written as the sector cache was not updated correctly.
- The interrupt enabling code in the GPIO driver was fixed for MCF5223x and MCF5225x devices. GPIO interrupt callbacks now work properly on these platforms.
- When the _task_restart() function was restarting the calling task itself, destroying task stack frame could crash the system. The behavior was fixed.
- Touch screen driver did not work properly for M5225X platforms and internal flash targets built in “Release” configuration. The driver code was fixed.

- Setting I2C baudrate did not selected the best closest achievable rate. Algorithm for baud rate calculation was changed to find closest match instead of closest-lower rate as it was before.
- FlashX driver fixed. The bug was causing writing incorrect data in case fseek() was used to navigate to the last byte of the Flash file and user wanted to overwrite it.
- Precompiled TWR-MPC5125 libraries were not included in MQX 3.6.0 release. The libraries are now part of the MQX 3.6.1 full installation package.
- New MQX projects created for MPC5125 platform were not using the MQX_PATH variable in the access paths. This might cause mismatch in library and include search paths in the newly created projects. This option is now set properly in project stationery.
- USB DATA0/1 bit toggling fixed in the EHCI USB Host driver code. This issue was affecting the PHDC and CDC data transfers.
- USB Device example applications for high-speed data access (EHCI) were updated to use internal data buffers in fast on-chip SRAM memory. This is a temporary solution which will be replaced by generic support of high-speed data transfer in future.
- The internal bind functionality was fixed in RTCS in order to enable the same TCP port being opened on two different network interfaces.
- Task suspended/running indication bug fixed in TAD. Call history is now displayed correctly for all tasks in both classic CodeWarrior and CodeWarrior version 10.
- Default PLL system clocks setup was changed back to MQX 3.5.0 values in M52277EVB BSP startup code. The SD RAM chip setting (160MHz) was unstable and caused sporadic errors during debugging.
- The lwtimer_create_periodic_queue() functionality was fixed. Timer callback was not invoked in some cases.

Version 3.6.2 (Nov 8th 2010)

- EAR v2 version of the Freescale Kinetis ARM® Cortex M4 Platform Support Package was prepared. See known issues related to this version in sections 6.12 and 6.13 below.
- The Kinetis port of the Freescale MQX is supported by the IAR Embedded Workbench for ARM. The support includes:
    - The EWARM workspace files ready to be used to batch-build all MQX libraries.
    - Projects files for standard set of libraries: PSP, BSP, RTCS, MFS, USB Host and Device.
    - Project files for all example applications and demo projects. Projects are tested with on-board P&E Micro OSJTAG interface and external Segger/IAR j-link interface.
    - By default all projects are prepared for EWARM 5.50.6 and are configured for IAR j-link debugging interface. The projects are ready to be converted to EWARM 6.10 format just by opening them in the new environment. In EWARM 6.10, the debugging connection may be changed to P&E Micro OSJTAG.
    - MQX Task Aware Debugging plug-in for both EWARM 5.50 and EWARM 6.10 versions.
- MK40X256-based TWR-K40X256 Board Support Package was added

- Standard set of peripheral drivers is enabled: GPIO, I2C, INT_CTRL, UART, SPI, RTC, ADC, SD card and FlashX.
  - USB Host and Device stacks are available.
- MK60N512-based TWR-K60N512 Board Support Package was added
  - Standard set of peripheral drivers is enabled GPIO, I2C, INT_CTRL, SERIAL, SPI, SDHC, SD card and FlashX.
  - USB Host and Device stacks are available.
  - Ethernet interface is supported by Enet driver and RTCS TCP/IP stack.
- Changes in the generic PSP code
  - Prototypes of functions which may be in conflict between the MQX and compiler-specific <string.h> header files were made conditionally compiled by the MQX_SUPPRESS_STRINGH_MACROS configuration option. This suppression is enabled by default for IAR EWARM compiler version 6.
  - New memory allocation API function defined. The _mem_alloc_at and _lwmem_alloc_at functions enable to allocate dynamic memory block at specific address within the memory pool. This is used to implement memory “barriers” between physical RAM blocks which cannot be accessed as a continuous area. (this is the case for Kinetis platform).
  - Support for MQX exception handling implemented for sparse interrupt tables.
  - Kernel log entries added for new memory functions and low-level ENET driver functions.
- Changes in the I/O drivers code
  - New MAC-NET Ethernet module driver with hardware time-stamping support was added to support Kinetis platform.
  - RTC driver API enhanced to be able to work with MQX DATE_STRUCT data type directly.
  - Minor changes done in ADC and GPIO drivers towards generalizing the code across multiple platforms. No functional change.
- Changes in the USB code
  - KHCI Host code significantly rewritten to support new Register I/O maps unified across ColdFire and Kinetis families. No functional change.
- The MQX documentation was updated:
  - Reference manual was extended by description of _lwevent_get_signalled() and _task_create_at() API functions.
  - Description of interrupt-level task priorities and NMI handling were extended in the User Manual.

Bug Fixes:

- In the PSP code
  - The _task_get_template_ptr() and _task_get_td() now safely returns NULL for invalid input parameter values.
- In the RTCS code

    - Compressed domain names haven’t been handled correctly in the DNS client parser. Some DNS server responses were not handled properly.
    - Memory leak in the DHCP functionality fixed. Memory was not de-allocated properly during address renewal process.
    - DHCP client requests were not sent complete. With some servers the client was not obtaining the subnet information.
    - UDP service fixed in order to receive frames to a proper socket in case the same UDP port is open by multiple network interfaces.
- In the I/O drivers code
    - ADC examples were modified to work properly also on MCF51MM, MCF51AC and MCF51CN platforms.
    - MCF52259 GPIO_PTCPAR_PTCPA macro values were fixed.
    - KBI Interrupt functionality was fixed in the MCF51MM GPIO driver.
    - IO_IOCTL_I2C_GET_BUS_AVAILABILITY command implemented in I2C driver for ColdFire V3 family.
    - IO_IOCTL_SERIAL_SET_DATA_BITS command behavior fixed in serial for ColdFire V1 family.
    - Slave functionality was fixed in DSPI driver for ColdFire families. The slave was sending additional byte after normal data block.
- Other
    - Shell “help” command was fixed to display detailed description properly.

Version 3.7.0 (Apr 8th 2011)

See Section 2 “What is New?” above.