Freescale MQX™ RTOS 4.1.0 FRDM-K64F Release Notes

1 Introduction

These are the Release Notes for the FRDM-K64F standalone package for Freescale MQX™ RTOS 4.1.0. Freescale K64F belongs to the Kinetis K series processor family of 32-bit microcontrollers. The software is built based on the MQX version 4.1.0. It includes the full set of RTOS services and a standard set of peripheral drivers. Prior installation of MQX 4.1.0 is not required to install this package.

For more information, see *Freescale MQX™ RTOS 4.1.0 Release Notes* (document MQXRN) and *Getting Started with Freescale MQX™ RTOS* (document MQXGSRTOS).

Contents

1.1 Development tools

The FRDM-K64F release was tested with the following development tools:

- CodeWarrior Development Studio for Microcontrollers Version 10.5
  - Support available for Kinetis K64 Ethernet USB 120 MHz MCUs
  - See build projects in `cw10gcc` subdirectories
  - Makefiles build option: TOOL = cw10gccc
- IAR Embedded Workbench® for ARM® Version 6.70.1
  - See build projects in `iar` subdirectories
- MDK-ARM^{TM} – Keil µVision® version 5.05
  - See build projects in `uv4` subdirectories
- GNU Tools for ARM Embedded Processors version 4.7-2013-q3
  - Makefiles build option: TOOL = gcc_arm
- Makefile support (mingw32-make version 3.8.2)
  - Library makefiles are located in `<MQX_root_dir>/build/<board>/make`
  - Application makefiles are located in `<example_dir>/build/make/<board>`

1.2 System requirements

The system requirements are defined by the development tool requirements. There are no special host system requirements for the Freescale MQX RTOS distribution.

- Minimum PC configuration:

  As required by Development and Build Tools
- Recommended PC configuration:

  GHz processor – 2 GB RAM – 2 GB free disk space
- Software requirement:

  OS: Windows XP or later

1.3 Target requirements

The FRDM-K64F package was tested with the following hardware configuration:

FRDM-K64F Freescale Freedom development platform Rev. C with an MK64FN1M0VLL12 processor

2 Features

2.1 Key features

This package provides support for the FRDM-K64F Freedom development platform and a standard set of features and example applications.

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

- Core clock: 120 MHz (High Speed Run mode, default)
- Bus clock: 60 MHz (High Speed Run mode, default)
- BSP Timer: SysTick
- Default console: ttya (debugger USB_CDC port, J4 micro USB connector)

The package supports these features:

- PSP support for the Kinetis K64F
- BSP for the FRDM-K64F development board
- Standard set of I/O drivers supporting the K64F peripherals including:
    - LWGPI/O driver
    - Serial interrupt and polled driver
    - SPI driver
    - I2C interrupt and polled driver
    - LWADC
    - Flash Driver
    - RTC Driver
    - PIT Timer
    - LPT Timer
    - WDOG
    - Low-power Mode
    - SAI
    - DMA
    - SD Card
    - Ethernet driver

- USB Host and Device drivers and stacks

- Example and demo applications demonstrating MQX, RTCS, USB and MFS usage

2.2 Limitations

This release does not support this feature:

CodeWarrior v10.5 New project wizard

2.3 Example applications

This package contains applications demonstrating kernel, peripheral, Ethernet and USB functionality on the FRDM-K64F development board. The applications can be found at the following locations:

- `<install_dir>/mqx/examples`: standard set of examples for kernel features and basic peripheral drivers

- `<install_dir>/mfs/examples`: example applications demonstrating the MFS file system features

- `<install_dir>/rtcs/examples`: example applications demonstrating the Ethernet system features

- `<install_dir>/usb/host(device)/examples`: examples demonstrating USB stack features and class drivers

- `<install_dir>/demo`: various demo applications showing more complex examples

2.4 Release content

This section provides an overview of the release content.

Deliverable	Location
MQX PSP Source Code and Examples	\<install_dir>/mqx/...
MQX PSP source code for Kinetis	.../mqx/source/psp/cortex_m
MQX PSP build projects	.../mqx/build/\<compiler>/psp_frdmk64f
MQX example applications	.../mqx/examples/...
MQX BSP Source Code	\<install_dir>/mqx/...
MQX BSP source code for FRDM-K64F board	.../mqx/source/bsp/frdmk64f
MQX BSP build projects	../mqx/build/\<compiler>/bsp_frdmk64f
RTCS Source Code and Examples	\<install_dir>/rtcs/...
RTCS source code	.../rtcs/source
RTCS build projects	.../rtcs/build/\<compiler>/rtcs_frdmk64f

RTCS example applications	.../rtcs/examples
USB Host Drivers Source Code and Examples	<install_dir>/usb/host/...
USB Host source code and class drivers	.../usb/host/source
HUB Class Driver	.../usb/host/source/classes/hub
Human Interface Device (HID) Class Driver	.../usb/host/source/classes/hid
Mass Storage (MSD) Class Driver	.../usb/host/source/classes/msd
CDC Class Driver	.../usb/host/source/classes/cdc
USB Host build projects	.../usb/host/build/<compiler>/usb_hdk_frdmk64f
USB Host example applications	.../usb/host/examples
USB Device Drivers Source Code and Examples	<install_dir>/usb/device/...
USB Device source code	.../usb/device/source
USB Device build projects	.../usb/device/build/<compiler>/usb_ddk_frdmk64f
USB Device example applications	.../usb/device/examples
Shell Library Source Code	<install_dir>/shell/...
Shell source code	.../shell/source
Shell build projects	.../shell/build/<compiler>/shell_frdmk64f
PC Host Tools	<install_dir>/tools
TFS Make Utility	…/tools/mktfs.exe
Check for Latest Version tool	…/tools/webchk.exe
Documentation	<install_dir>/doc
User Guides and Reference Manuals for MQX RTOS, RTCS, MFS, IO Drivers, USB etc.	…/doc

Note: <compiler> can be iar, uv4, make or cw10gcc.

3   Installation Instructions

3.1   Installation guide

Run the FRDM-K64F package installer and proceed according to instructions. This package can be used independently of the current MQX 4.1.0 installation folder.

3.1.1   Building procedure

For build procedures, see *Getting Started with Freescale MQX™ RTOS* (document MQXGSRTOS).

3.1.2   Default jumper settings

These are the default jumper settings for FRDM-K64F Rev. C operation:

- J25 on position 1-2

- J14, J17, J18, and J20 open

- J5, J7, J8, J11, J12, J16, J19, J21, and J23 closed

3.1.3   Important jumper settings

For basic operations, ensure that the following jumper settings are applied:

- For USB Device and USB Host mode, use the onboard J22 micro USB connector and the default jumper settings.

- When using the J-Link debugger, populate the JTAG/SWD CONNECTOR J9, and cut off the J8 and J12 jumpers.

3.1.4   Board-specific build targets

Internal Flash (Debug and Release): These targets enable building applications suitable for booting the system from the Internal Flash memory. After reset, the code is executed from the Internal Flash.

4 Other Notes

4.1 IAR patch

Because IAR EW version 6.70.1 does not support K64F120M, a patch is needed for this device.

- Decompress the patch "`Patch_for_IAR_to_include_K24,_K63,_and_K64.zip`" located at `<install_dir>\tools\iar_extensions` to a temporary folder.

- Copy the three overwritten folders inside the patch "`config`", "`inc`", "`src`" to the "`arm`" folder of the IAR default installation.

    For example: `C:\Program Files\IAR Systems\Embedded Workbench 6.70\arm`

- If required, create a backup copy of the current files.

CMSIS-DAP is set as the default debugger for all the IAR projects. This release has been tested with the CMSIS-DAP firmware version 2.1 (build date: Feb 25, 2014).

4.2 Keil (Uv4) patch

Because the Keil version 5.05 does not support the K64F120M device, a patch is required for this device. Install the patch by following the installer instructions.

The Patch for Keil can be found at:
`<install_dir>\tools\keil_extensions\uVision4\`

The file name is: `MDK-MRM_AddOn_K24F_K63F_K64F.exe`.

CMSIS-DAP is set as the default debugger in the Keil projects. Before debugging an application, verify that the Single Wire option is set and the correct flash algorithm is selected. This release has been tested with the CMSIS-DAP firmware version 2.1 (build date: Feb 25, 2014).
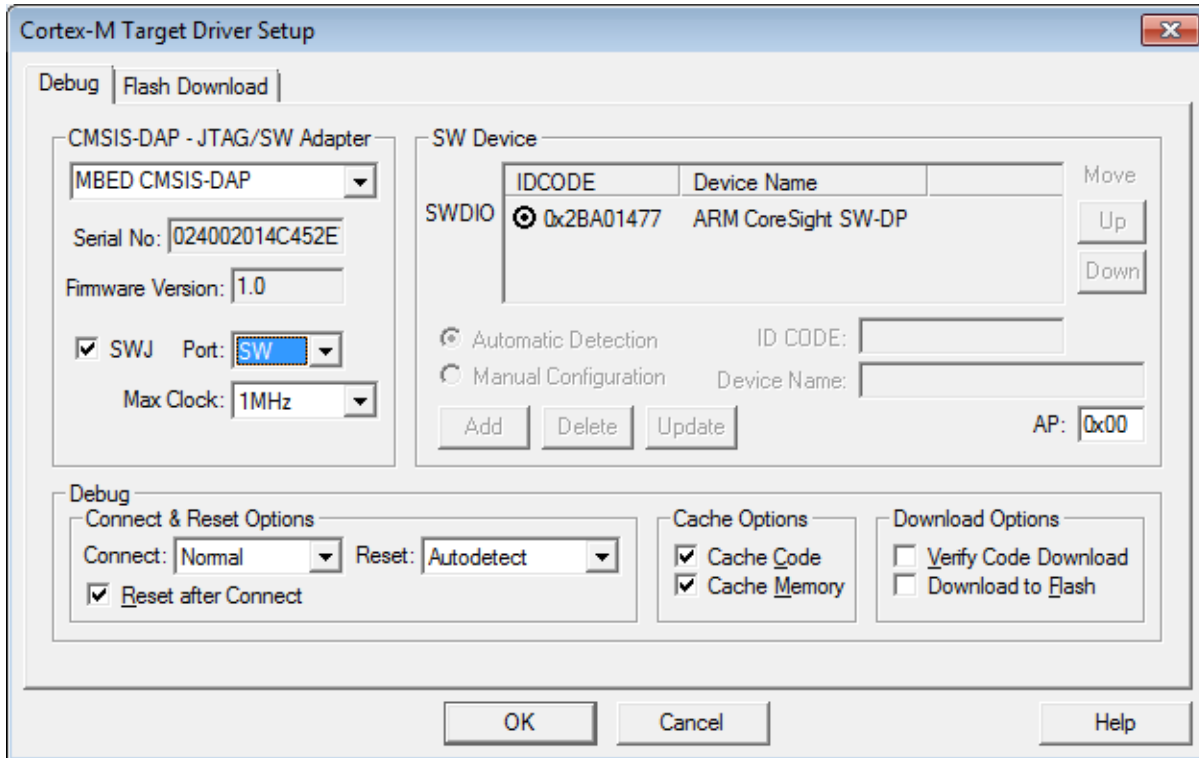
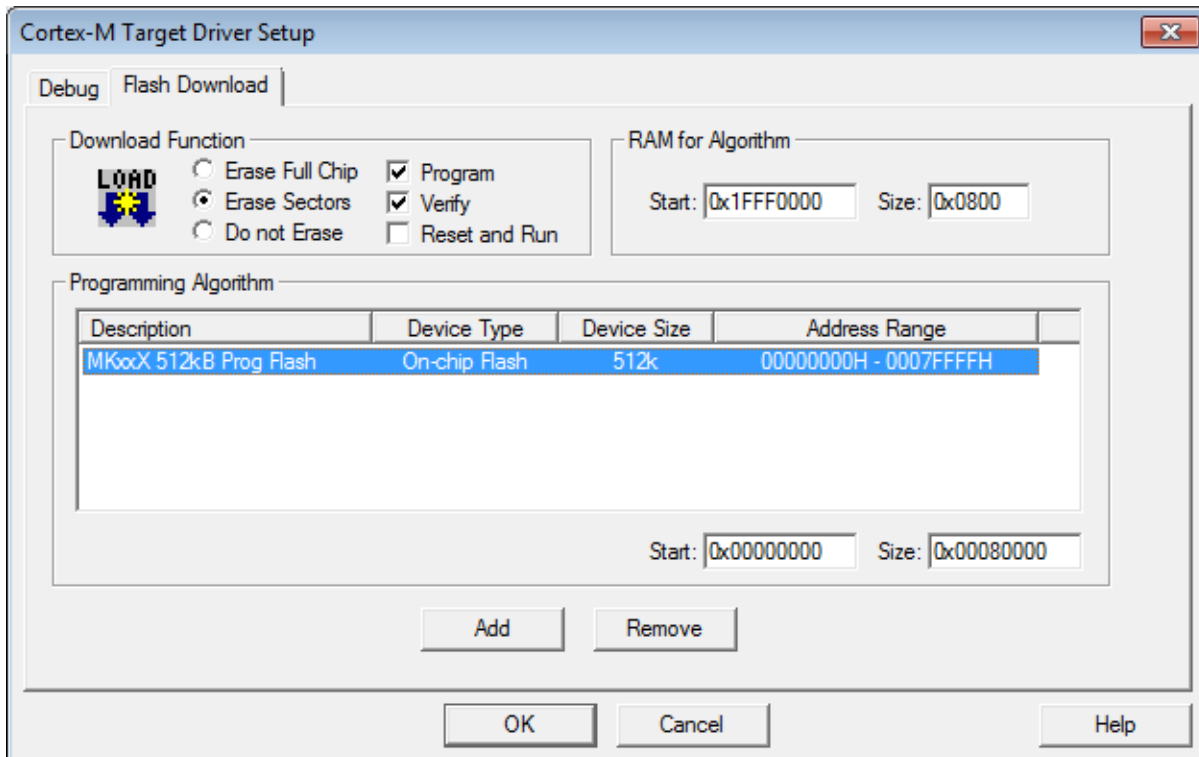**Figure 1 Cortex-M Target Driver Setup dialog box (1)**

Figure 2 Cortex-M Target Driver Setup dialog box (2)

4.3 CodeWarrior v10.5

- Install all the available updates for CodeWarrior v10.5 through the update site.

- Install the "CW MCU v10.5 Kinetis K24_K63_K64_120MHz Service Pack (Win)" service pack version 1.0.2 for K64 devices available at the Freescale website.

  Search for "CW-MCU10: CodeWarrior for MCUs" on freescale.com.

- J-Link is set as the default debugger for all the CodeWarrior projects. For the proper functionality of all the applications, update your J-Link drivers or firmware to the latest available version. It is also recommended to update the J-Link ARM dll in <CW 10.5 installation folder>\MCU\bin\plugins\support\ARM\gdi\JLinkARM.dll to the latest version from SEGGER.

5 Known Issues

- When running a USB host-based application, such as mfs_usb or web_hvac, the application console (debugger USB_CDC) stops working when an USB device is attached to the J22 micro USB connector. The application itself still works fine. To avoid this board issue, plug-in the USB device before the application is started.

- The `CW MCU v10.5 Kinetis K24_K63_K64_120MHz Service Pack (Win)` service pack also contains the Processor Expert support for Kinetis K64 devices. However, the generated K64 Processor Expert files are not consistent with the way these files are generated for other Kinetis platforms supported in the MQX 4.1.0 release. To get the MQX pe_demo working, perform the following steps when the FRDM-K64F BSP project is built:

1. Replace "`${ProcessorExpertPath}/lib/Kinetis/pdd_100331/inc`" by "`${ProcessorExpertPath}/lib/Kinetis/pdd_100308/inc`" in respective project settings.

    Because of the 1.0.2 version of the "`CW MCU v10.5 K24F_K63F_K64F Service Pack`", it is necessary to change the path to the correct Processor Expert PDD headers in the BSP project as well as in all the application projects (including the pe_demo application).

2. Once the Processor Expert files are generated (BSP build project), in the "Properties for Vectors.c" dialog box, select "Exclude resource from build" to disable the generated Vectors.c file.
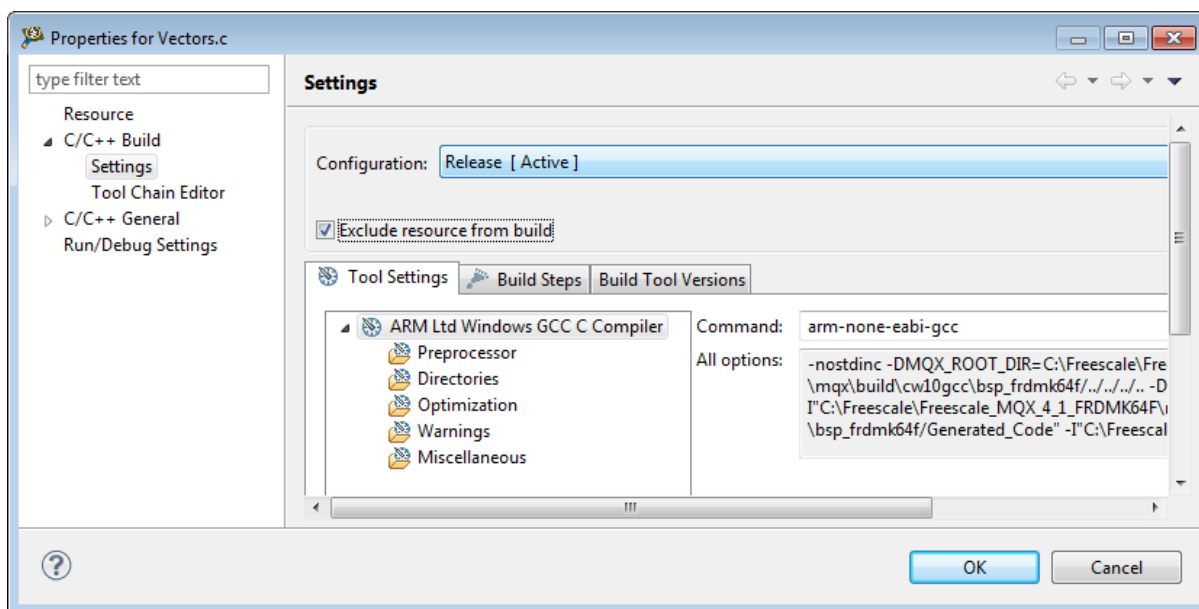

Figure 3 Properties for Vectors.c dialog box

3. Manually change the C-code of the `CPU_Init.c/.h` files.

    Rename the `_init_hardware()` function to `_pe_initialize_hardware()` and the delete the configuration of the SCB_VTOR register from this function in `CPU_Init.c`.

```
/*
** ===================================================================
```

```
**      Method      :  __init_hardware (component MK64FN1M0LQ12)
**
**      Description :
**          This method is called from the _startup() function and
**          provides necessary system initialization such as PLL, and
**          external bus.
**          This method is internal. It is used by Processor Expert only.
**  ==================================================================
*/
#if defined(__IAR_SYSTEMS_ICC__)
int __low_level_init()
#elif defined(__CWCC__) | defined(__GNUC__) | defined(__CC_ARM)
void __pe_initialize_hardware()
#endif
{
  /* Interrupt vector placement initialization */
  #if defined(__IAR_SYSTEMS_ICC__)
  SCB_VTOR = (uint32_t)(&__vector_table); /* Set the interrupt vector table position */
  #elif defined(__CWCC__) | defined(__GNUC__) | defined(__CC_ARM)
  SCB_VTOR = (uint32_t)(&__vect_table); /* Set the interrupt vector table position */
  #endif
  /* Disable the WDOG module */
  #if STARTUP_WDOG
  WDOG_UNLOCK = (uint16_t)STARTUP_WDOG_KEY_1; /* Key 1 */
  WDOG_UNLOCK = (uint16_t)STARTUP_WDOG_KEY_2; /* Key 2 */
  WDOG_STCTRLH = (uint16_t)STARTUP_WDOG_STCTRLH_VALUE; /* Disable WDOG */
  #endif /* STARTUP_WDOG */
```