Freescale Semiconductor
User's Guide

Document Number: MQXKDSUG
Rev. 01, 04/2015

Getting Started with Kinetis Design Studio IDE and Freescale MQX™ RTOS

How to Reach Us:

Home Page:
www.freescale.com

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

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

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale 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 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 does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Table of Contents

1 Read Me First

This document describes how to use the Kinetis Design Studio (KDS) IDE for the MQX™ RTOS basic development tasks. See *Getting Started with Freescale MQX RTOS* (document MQXGSRTOS) and other user documentation included in the latest Freescale MQX RTOS installation for more information that is not specifically related to the KDS IDE tools.

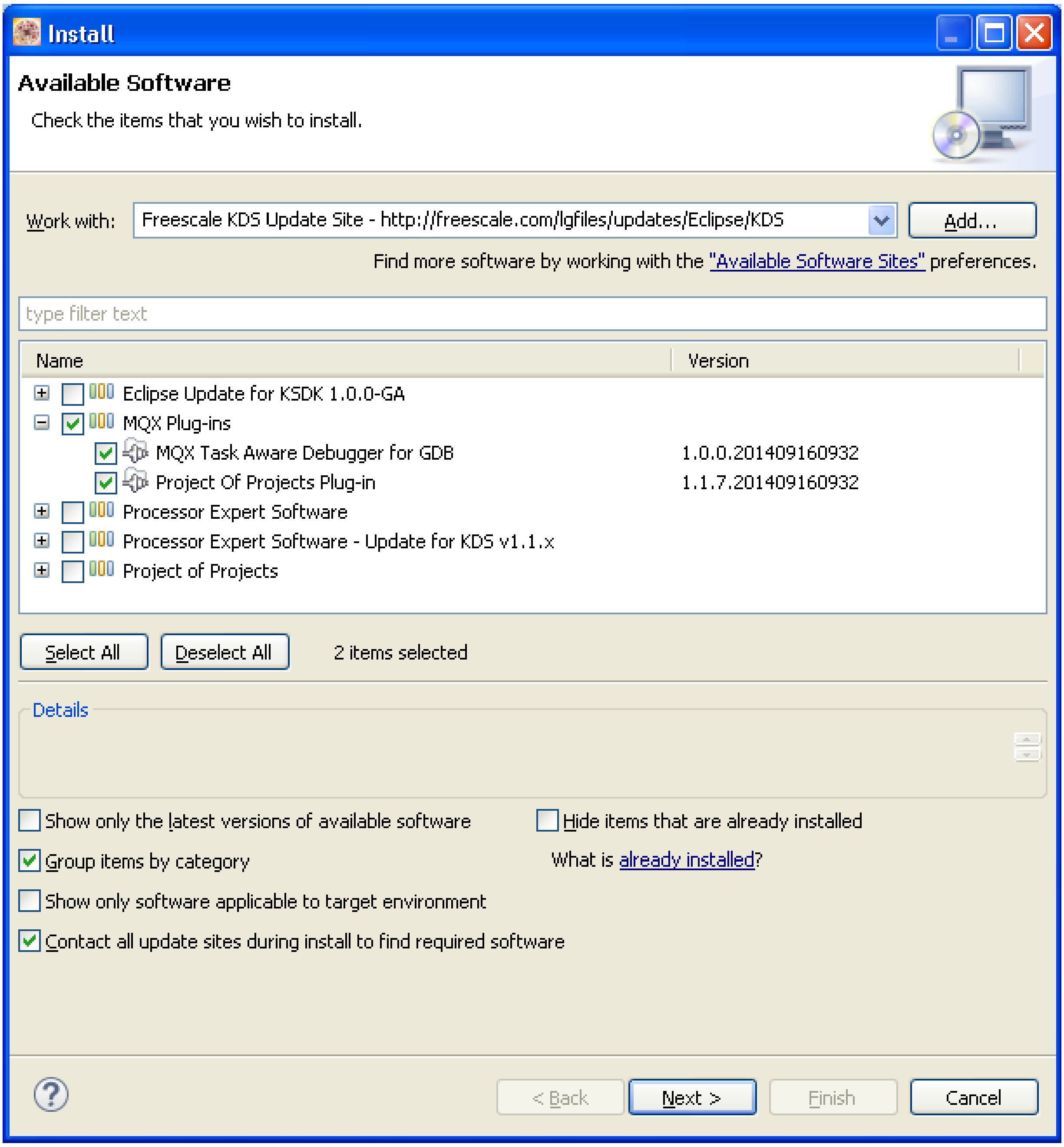
Use the latest Freescale MQX RTOS available at freescale.com/mqx.

2 MQX RTOS Build – initial steps

The MQX RTOS release provides the KDS IDE native projects to more conveniently build MQX RTOS libraries and applications.

This chapter concentrates on KDS IDE-specific steps only. For details about the generic build process and compile time configuration, see Chapter 2 of the *Getting Started with Freescale MQX™ RTOS* (document MQXGSRTOS).

Install the MQX RTOS KDS IDE plug-ins using ***Help | Install New Software*** menu. Chose the ***Freescale Update Site*** from the ***Work with*** menu, and select the checkboxes next to **MQX Plug-ins** items.



Available Software

Check the items that you wish to install.

Work with: Freescale KDS Update Site - http://freescale.com/lgfiles/updates/Eclipse/KDS

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
⊞ □ Eclipse Update for KSDK 1.0.0-GA	
⊟ ☑ MQX Plug-ins	
☑ MQX Task Aware Debugger for GDB	1.0.0.201409160932
☑ Project Of Projects Plug-in	1.1.7.201409160932
⊞ □ Processor Expert Software	
⊞ □ Processor Expert Software - Update for KDS v1.1.x	
⊞ □ Project of Projects	

2 items selected

Details

□ Show only the latest versions of available software

□ Hide items that are already installed

☑ Group items by category

What is already installed?

□ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

Figure 1: Install KDS IDE plug-ins

Install GDB server for Kinetis Devices application from P&E Micro web site or alternative GDB server software. For a step-by-step guide on J-Link GDB server debugging see the `<mqx_install_dir>/doc/tools/gnu/MQX_GNU_Getting_Started.pdf` document.

3 Building MQX RTOS library and application project files

Every application/example in MQX RTOS has one associated working set description file which includes the path to the example project file and the dependent MQX RTOS library project file. Simply import that file into KDS working space. For example with Hello World example of MQX RTOS for FRDM-K64F120M BSP import file

`<mqx_install_dir>/mqx/examples/hello/build/kds/hello_frdmk64f/hello_frdmk64f.wsd`

using ***File\Import\MQX\Import Working Sets*** menu. The MQX RTOS library projects and Hello World example project will be imported into KDS IDE working space together with build configurations settings.

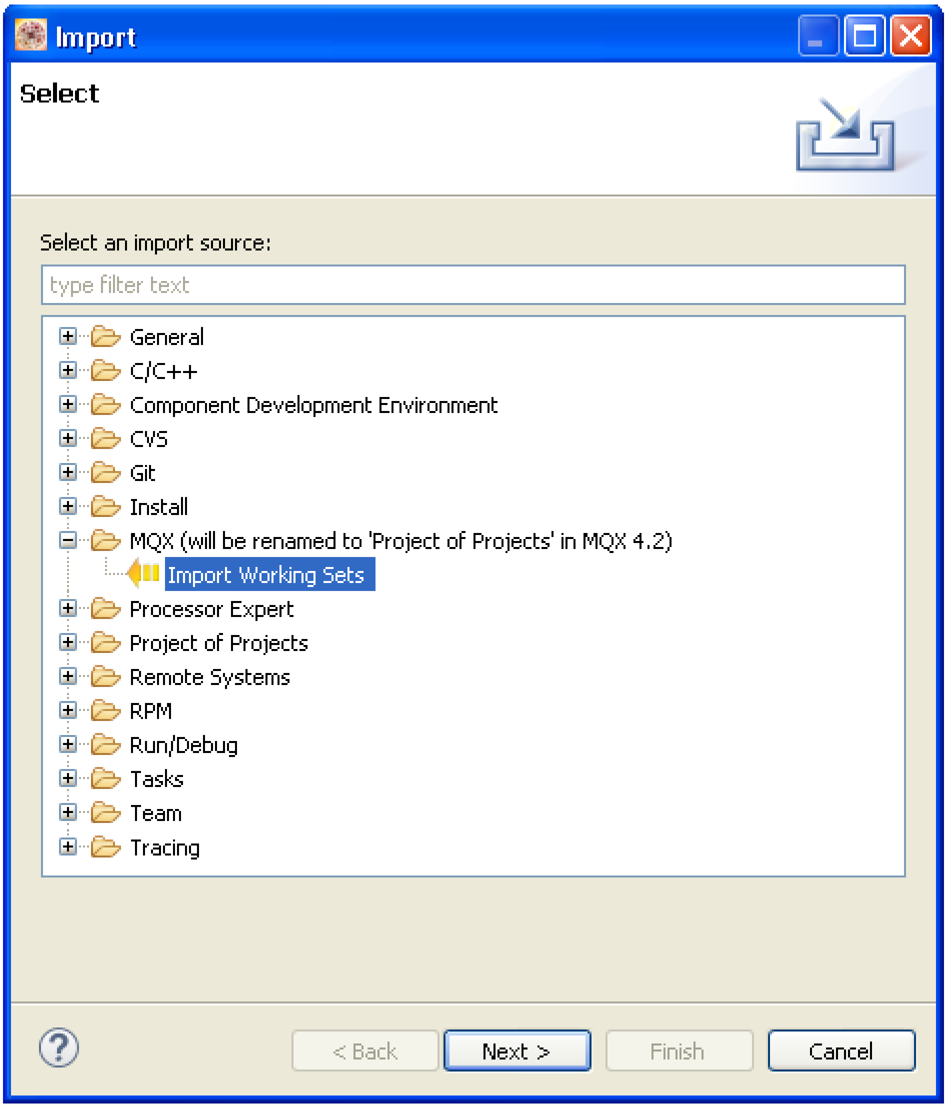


Figure 2: Import MQX RTOS library

First, build MQX RTOS libraries by left-clicking the project file in the Project Explorer tab view and click the hammer icon (in red) and then select the target to build.

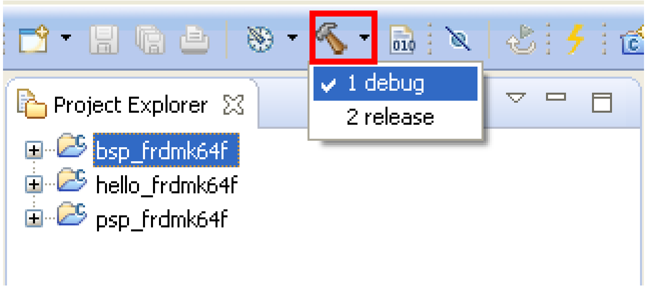


Figure 3: Build library

After the user is finished building the MQX RTOS libraries (bsp and psp in this case), select the hello project and build different target with different memory configuration.

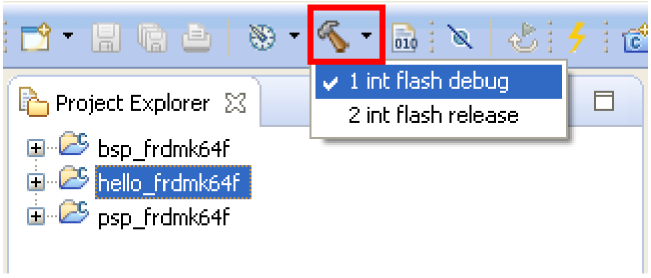


Figure 4: Build example application

4 Running and Debugging MQX RTOS applications

This description is provided for the Kinetis FRD-MK64F BSP and Hello World example applications. The same procedure applies for all other BSPs and example applications distributed in the MQX RTOS release package.

4.1 Debugging MQX RTOS Hello World program

Connect a USB cable to the OpenSDA debug connector. Set the communication speed to 115200 in the terminal program.

Click the arrow next to the bug icon (in red) and select **Debug Configurations**.

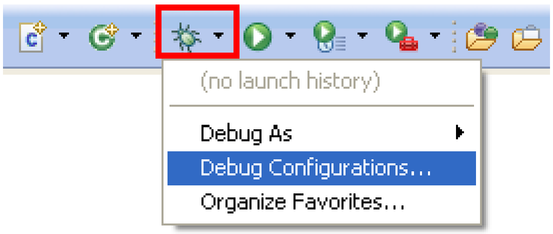
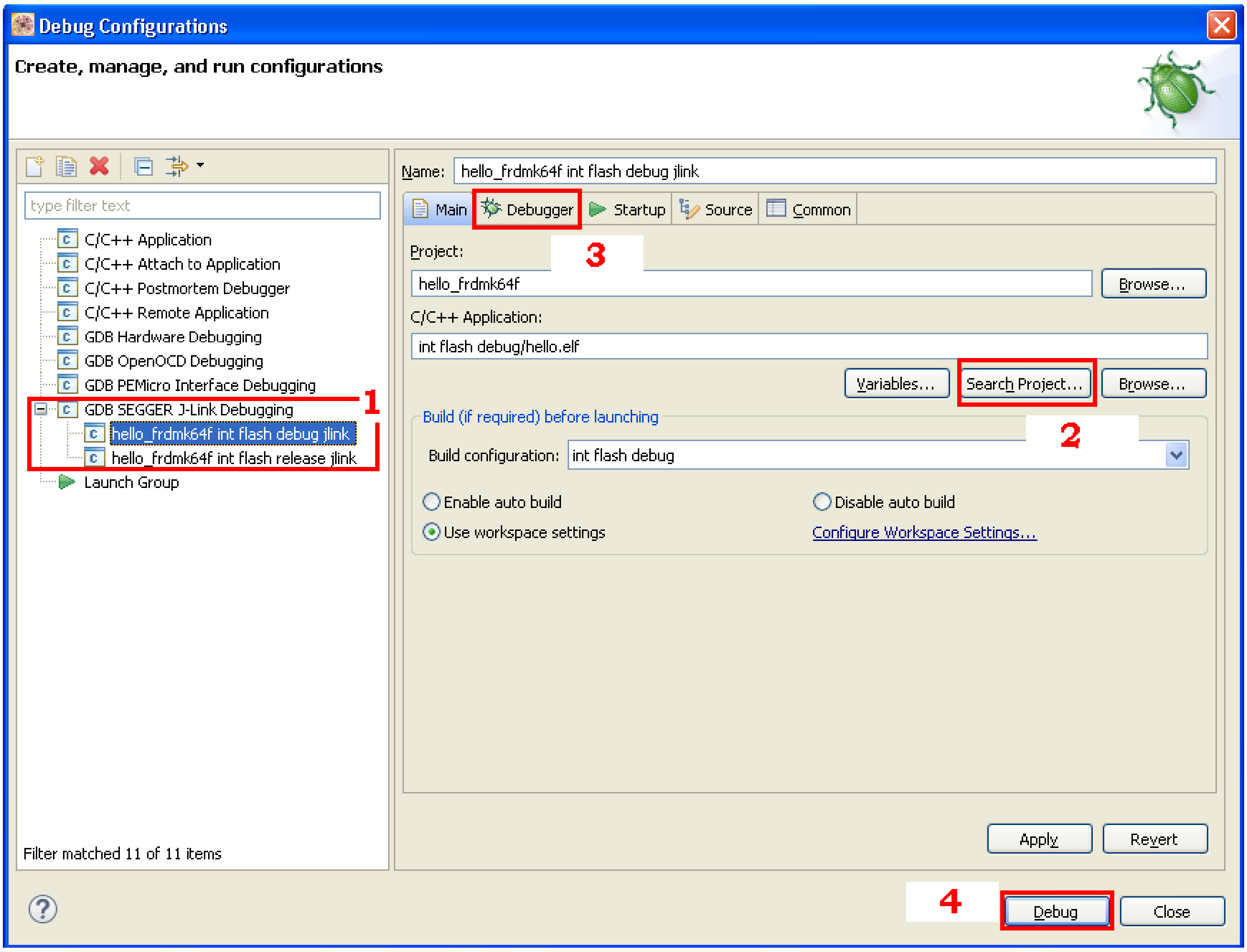


Figure 5: Open debug configuration

In the Debug Configurations window, select the debugger type, and then configure the project to debug (see Figure 6). The example application image is loaded into the internal Flash memory. The application is executed and runs to the main function (see Figure 7).



Debug Configurations

Create, manage, and run configurations

Filter matched 11 of 11 items

Name: hello_frdmk64f int flash debug jlink

Project: hello_frdmk64f

C/C++ Application: int flash debug/hello.elf

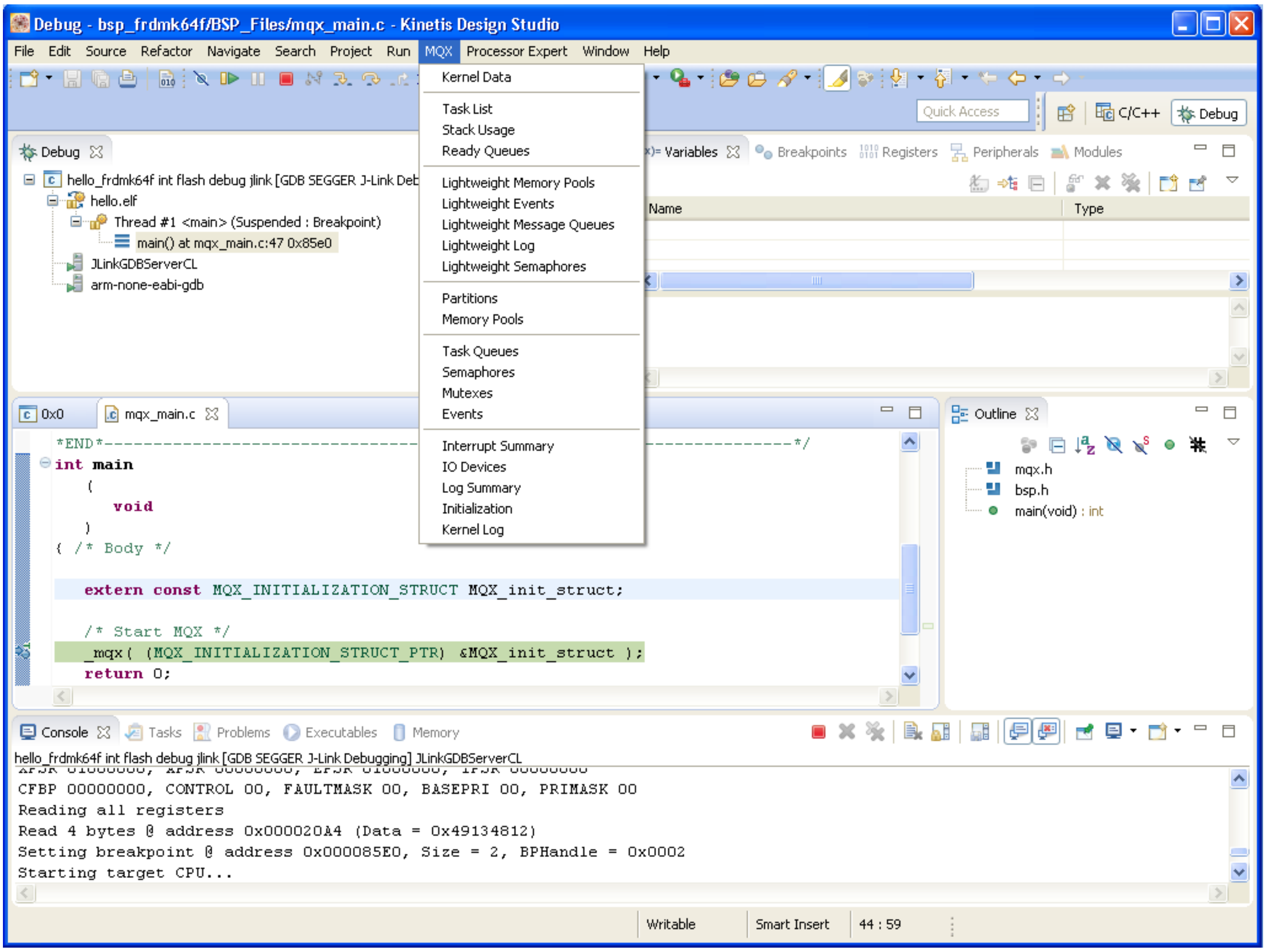
Build (if required) before launching

Build configuration: int flash debug

○ Enable auto build ○ Disable auto build

● Use workspace settings Configure Workspace Settings...

Figure 6: Configure the debugger



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

   extern const MQX_INITIALIZATION_STRUCT MQX_init_struct;

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

hello_frdmk64f int flash debug jlink [GDB SEGGER J-Link Debugging] JLinkGDBServerCL

```
CFBP 00000000, CONTROL 00, FAULTMASK 00, BASEPRI 00, PRIMASK 00
Reading all registers
Read 4 bytes @ address 0x000020A4 (Data = 0x49134812)
Setting breakpoint @ address 0x000085E0, Size = 2, BPHandle = 0x0002
Starting target CPU...
```

Figure 7: Debugging an application program

4.2 MQX RTOS Task Aware Debugging

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

4.2.1 Using MQX RTOS TAD Screens

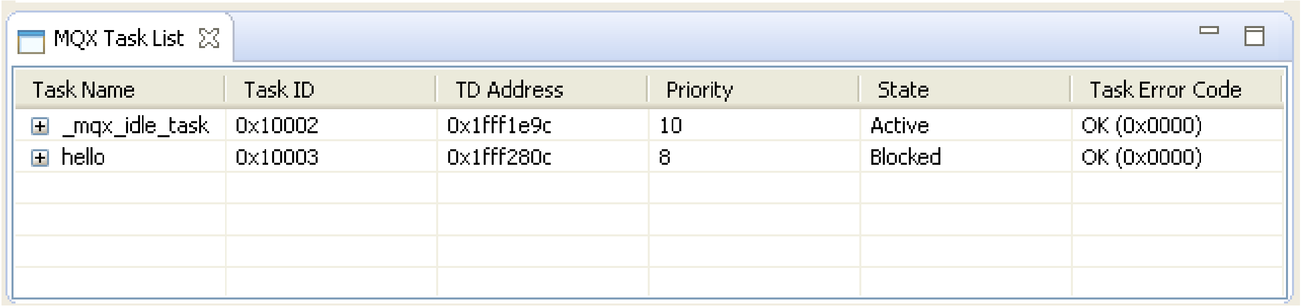
MQX RTOS TAD Screens are accessible from the MQX RTOS menu which is displayed during the debug session.

Resume (F8 or Run/Resume) the debug session and then suspend (Run/Suspend) it again to initialize MQX RTOS structures needed for the MQX RTOS TAD.

For the Hello World example of MQX RTOS for the FRD-MK64F platform, the following components of MQX RTOS are extracted and displayed thanks to the MQX RTOS TAD plug-in in KDS.

The most helpful and frequently used screens are shown in these figures:

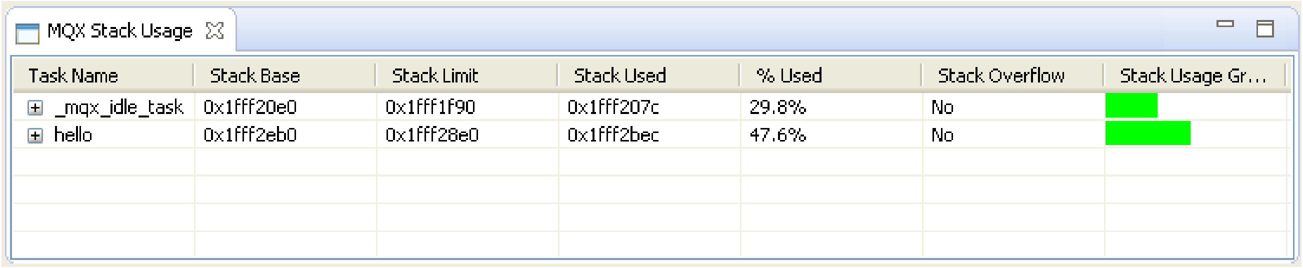
Figure 8, Task List – overview about all tasks created in the MQX RTOS application



Task Name	Task ID	TD Address	Priority	State	Task Error Code
⊞ _mqx_idle_task	0x10002	0x1fff1e9c	10	Active	OK (0x0000)
⊞ hello	0x10003	0x1fff280c	8	Blocked	OK (0x0000)

Figure 8: Task list

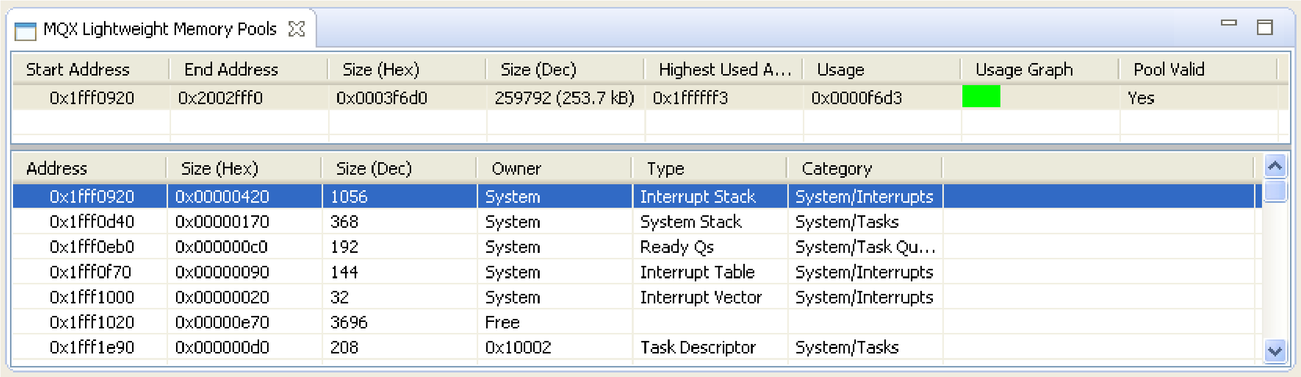
Figure 9, Stack Usage – displays information about interrupt and task stacks. Typically, a stack overflow is the root cause for vast majority of problems in MQX RTOS user applications.



Task Name	Stack Base	Stack Limit	Stack Used	% Used	Stack Overflow	Stack Usage Gr...
⊞ _mqx_idle_task	0x1fff20e0	0x1fff1f90	0x1fff207c	29.8%	No	🟩
⊞ hello	0x1fff2eb0	0x1fff28e0	0x1fff2bec	47.6%	No	🟩

Figure 9: Stack usage

Figure 10, Memory Pools (or Lightweight Memory Pools) – displays address, size, and type information about each memory block allocated in the selected memory pool by the MQX RTOS system or applications.

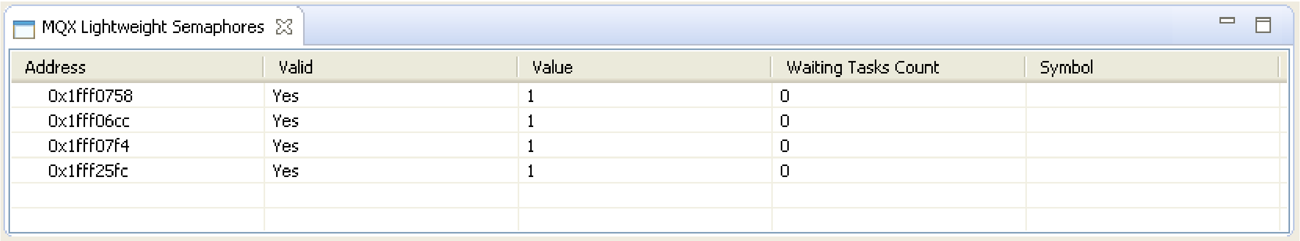


Start Address	End Address	Size (Hex)	Size (Dec)	Highest Used A...	Usage	Usage Graph	Pool Valid
0x1fff0920	0x2002fff0	0x0003f6d0	259792 (253.7 kB)	0x1ffffff3	0x0000f6d3	🟩	Yes

Address	Size (Hex)	Size (Dec)	Owner	Type	Category
0x1fff0920	0x00000420	1056	System	Interrupt Stack	System/Interrupts
0x1fff0d40	0x00000170	368	System	System Stack	System/Tasks
0x1fff0eb0	0x000000c0	192	System	Ready Qs	System/Task Qu...
0x1fff0f70	0x00000090	144	System	Interrupt Table	System/Interrupts
0x1fff1000	0x00000020	32	System	Interrupt Vector	System/Interrupts
0x1fff1020	0x00000e70	3696	Free		
0x1fff1e90	0x000000d0	208	0x10002	Task Descriptor	System/Tasks

Figure 10: LW-Memory pools

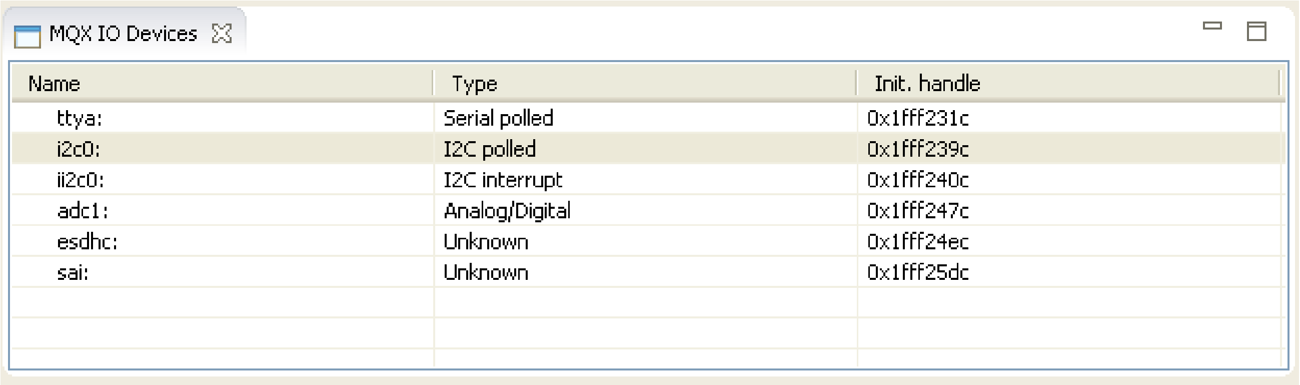
Figure 11, Semaphores, Events, Mutexes (or Lightweight Semaphores, Lightweight Events) - display address and status of synchronization objects created by the MQX RTOS system or application. When a synchronization object is allocated either as a global or static variable in the system, or as an array element or as a structure member allocated as global or static variable, the TAD plug-in also displays the symbolic name of the object.



Address	Valid	Value	Waiting Tasks Count	Symbol
0x1fff0758	Yes	1	0	
0x1fff06cc	Yes	1	0	
0x1fff07f4	Yes	1	0	
0x1fff25fc	Yes	1	0	

Figure 11: LW-Semaphores

Figure 12, I/O Devices – displays name, type and address of I/O Devices used by MQX RTOS application.



Name	Type	Init. handle
ttya:	Serial polled	0x1fff231c
i2c0:	I2C polled	0x1fff239c
ii2c0:	I2C interrupt	0x1fff240c
adc1:	Analog/Digital	0x1fff247c
esdhc:	Unknown	0x1fff24ec
sai:	Unknown	0x1fff25dc

Figure 12: I/O devices