Document Number: FREERTOSKSDKUG

FreeRTOS and Kinetis SDK Integration Notes

Freescale Semiconductor, Inc.

The Kinetis SDK (KSDK) release includes a pre-integrated FreeRTOS zip package. Section 1 of this document describes the modifications that have been done by Freescale to the official FreeRTOS 7.5 release to create the pre-integrated FreeRTOS package.

To install the FreeRTOS kernel to the correct place in the KSDK directory structure, follow the instructions provided in the readme file included in the package. When the FreeRTOS kernel is installed, you can build and run the demo "I2C Demo with RTOS" to verify it is functioning correctly. See the *Kinetis SDK Demo Applications User's Guide* (document KSDKDEMOUG) for details about the demo. Section 2 of this document provides a step-by-step guide to use the IAR tool chain to build a new project using the KSDK with FreeRTOS.

Contents

1 Modifications to the official FreeRTOS Release

The steps in this section describe the process Freescale followed to integrate the FreeRTOS to KSDK. If you use the pre-integrated version included with the KSDK release package, these steps are not required and you may skip to Section 2.

1. Download the official FreeRTOS 7.5.0 release and extract it.
2. Copy these files to the folder <KSDK_ROOT>\rtos\FreeRTOS\src.

   FreeRTOSV7.5.0\FreeRTOS\Source\croutine.c
   FreeRTOSV7.5.0\FreeRTOS\Source\list.c
   FreeRTOSV7.5.0\FreeRTOS\Source\queue.c
   FreeRTOSV7.5.0\FreeRTOS\Source\tasks.c
   FreeRTOSV7.5.0\FreeRTOS\Source\timers.c
   FreeRTOSV7.5.0\FreeRTOS\Source\portable\MemMang\heap_1.c
   FreeRTOSV7.5.0\FreeRTOS\Source\portable\MemMang\heap_2.c
   FreeRTOSV7.5.0\FreeRTOS\Source\portable\MemMang\heap_3.c
   FreeRTOSV7.5.0\FreeRTOS\Source\portable\MemMang\heap_4.c
3. Copy the folder FreeRTOSV7.5.0\FreeRTOS\Source\include to the folder <KSDK_ROOT>\rtos\FreeRTOS, which contains these:

Figure-1 <KSDK_ROOT>\rtos\FreeRTOS folder structure

4. Modify FreeRTOS code for Kinetis SDK integration:
   - Include task.h for taskENTER_CRITICAL/taskEXIT_CRITICAL in the FreeRTOS OSA.

```
File: FreeRTOS/include/FreeRTOS.h
#include "portable.h"
+#include "task.h"
```

   - Add macros to divide the different heap schemes.

```
File: FreeRTOS/src/heap_1.c
 #include <stdlib.h>
+#include "FreeRTOSConfig.h"
+#if configFRTOS_MEMORY_SCHEME==1
…
```

```
+#endif // Add this at the end of this file.

File: FreeRTOS/src/heap_2.c
#include <stdlib.h>
+#include "FreeRTOSConfig.h"
+#if configFRTOS_MEMORY_SCHEME==2
…
+#endif // Add this at the end of this file.

File: FreeRTOS/src/heap_3.c
#include <stdlib.h>
+#include "FreeRTOSConfig.h"
+#if configFRTOS_MEMORY_SCHEME==3
…
+#endif // Add this at the end of this file.

File: FreeRTOS/src/heap_4.c
 #include <stdlib.h>
+#include "FreeRTOSConfig.h"
+#if configFRTOS_MEMORY_SCHEME==4
…
+#endif // Add this at the end of this file.
```

- Suppress the IAR Pa082 warning.

```
File: FreeRTOS/src/tasks.c
 #include "StackMacros.h"
+#if (configCOMPILER == configCOMPILER_ARM_IAR)
+#pragma diag_suppress=pa082
+#endif
```

5. Download the pre-integrated FreeRTOS zip package from Freescale, unzip it and copy the folders “config”, “port”, and “app” to the folder <KSDK_ROOT>\FreeRTOS. The folders “config” and “port” contain configuration files and port codes for Kinetis MCUs. The folder “app” contains an example, which demonstrates using FreeRTOS with Kinetis SDK.

2 Create an IAR project using KSDK with FreeRTOS

This section is for users who wish to create a project from scratch using KSDK with FreeRTOS. The demo application I2C Demo with RTOS provides a FreeRTOS project. If you wish to run the demo, you can disregard this section.

1. Create an empty IAR project as shown in these figures:

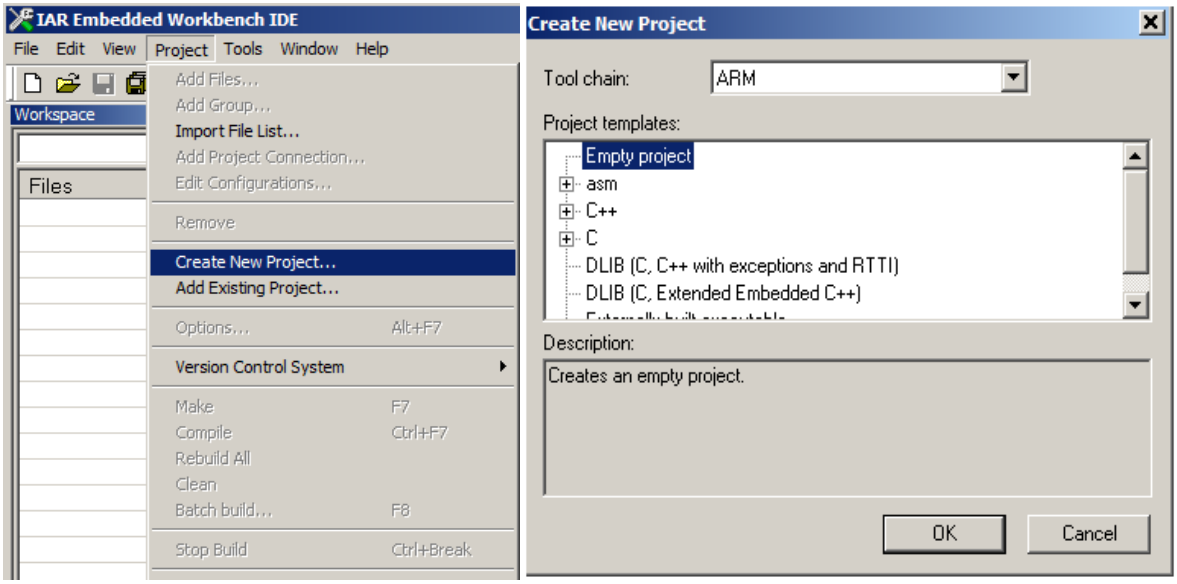

Figure-2 Creating a new project in IAR IDE

2. Choose the device type.

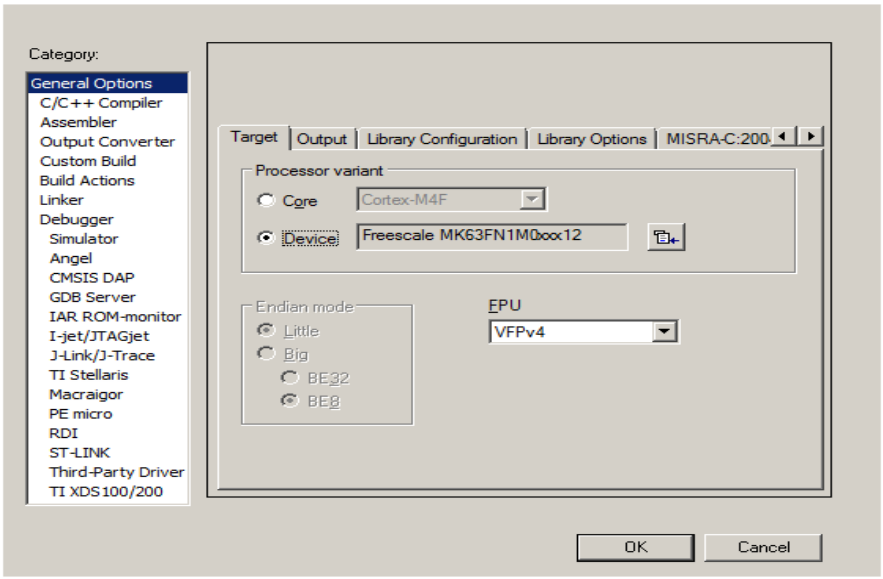

Figure-3 choosing a device type

3. Add FreeRTOS source code to the project. FreeRTOS source code is in the folder <KSDK_ROOT>\rtos\FreeRTOS. Note that the source code in **src** and **port** folders must be added.

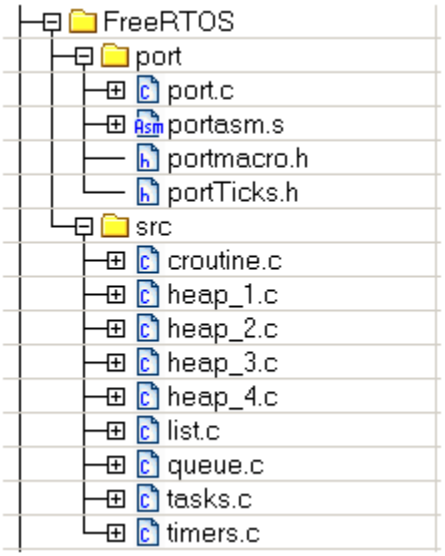

Figure-4 FreeRTOS source code

4. Add startup code to the project, which is in the <KSDK_ROOT>\platform\startup, according to the hardware you are using.

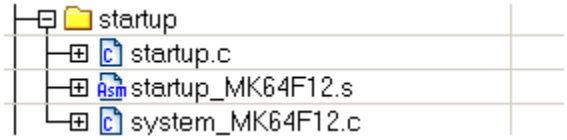

Figure-5 Adding startup code

5. Add the BSP files, which are in the folder <KSDK_ROOT>\boards\<BOARD>.

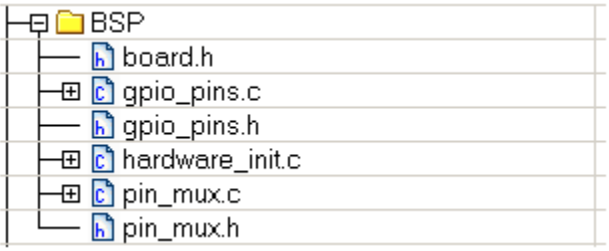

Figure-6 Adding BSP files

6. Add your application code. You can use the <KSDK_ROOT>\rtos\FreeRTOS\app\app.c as a template. To use OSA, include the file fsl_rtos_abstraction.h and add the <KSDK_ROOT>\rtos to the include path.
7. Add the include paths to the project. Specifically, add the assembler include path.

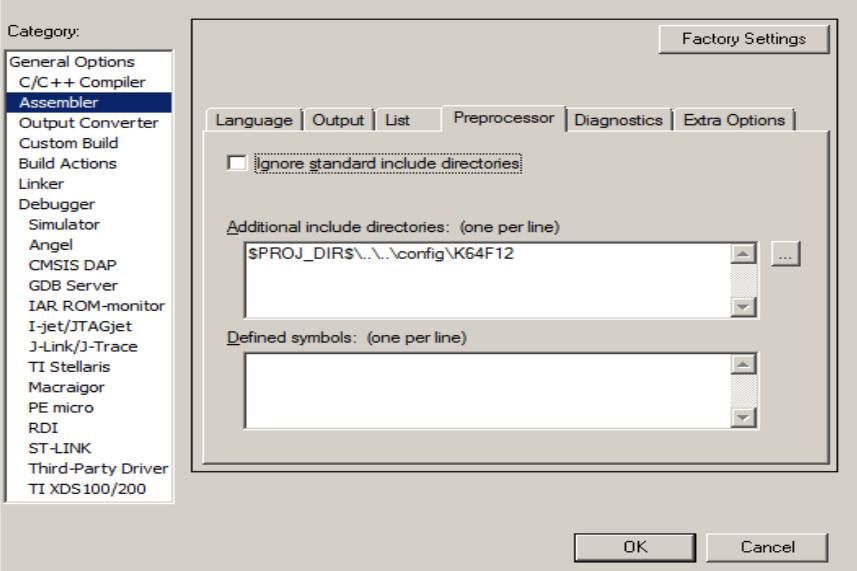

Figure-7 Adding include paths

8. Add predefined symbol FSL_RTOS_FREE_RTOS and other necessary predefined symbols to the box Defined symbols.

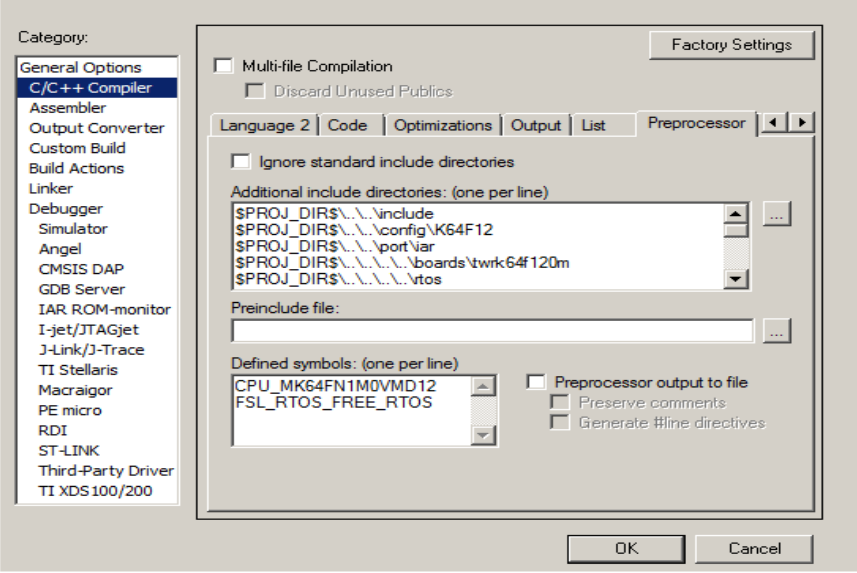

Figure-8 Adding predefined symbols

9. Add the file platform_freertos_lib.a as a link library. The library is in the folder <KSDK_ROOT>\lib\iar\<CHIP>\output\<Debug or Release>\.

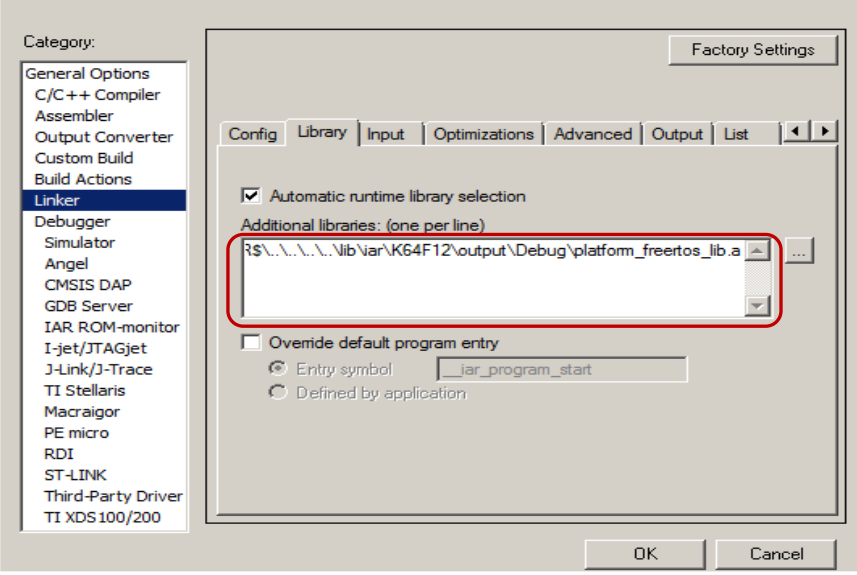

Figure-9 Adding platform_freertos_lib.a

10. Change to use the linker script in the folder <KSDK_ROOT>\platform\linker\iar\<CHIP>\.

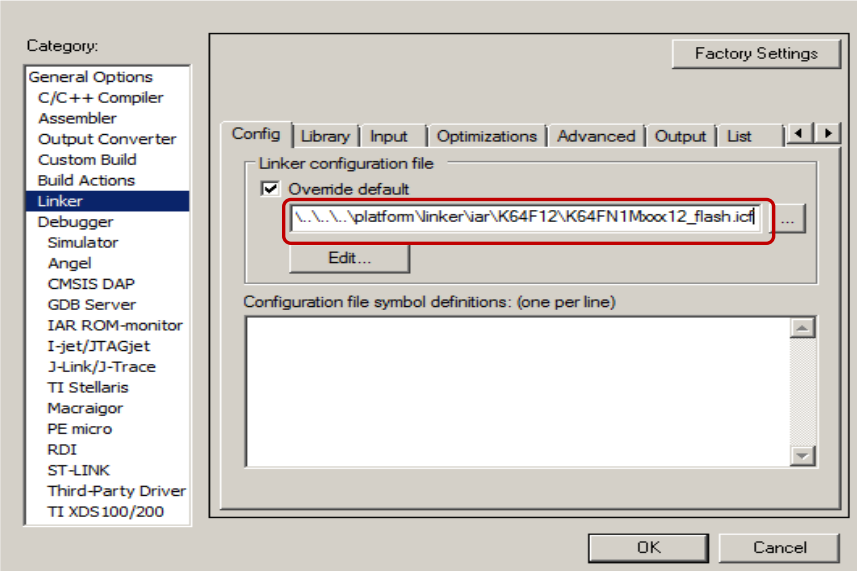

Figure-10 Using link script

11. Set up the debugger.

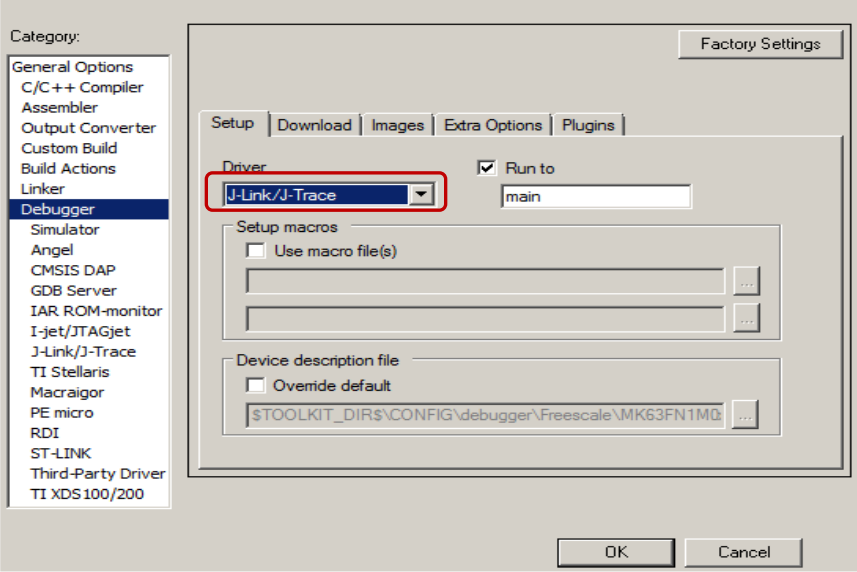

Figure-11 choosing the debugger

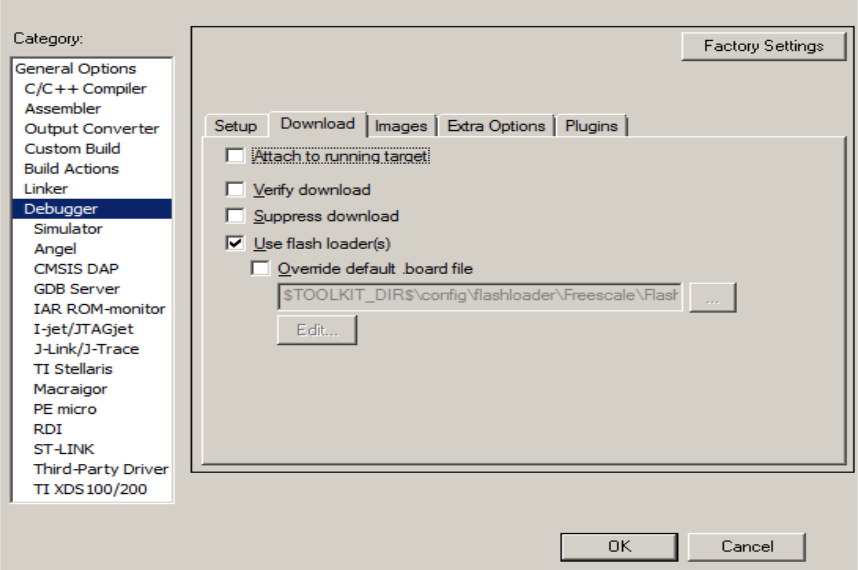

Figure-12 setting up the debugger

12. Drag the project <KSDK_ROOT>\lib\iar\<CHIP>\platform_freertos_lib.ewp to the IAR workspace.

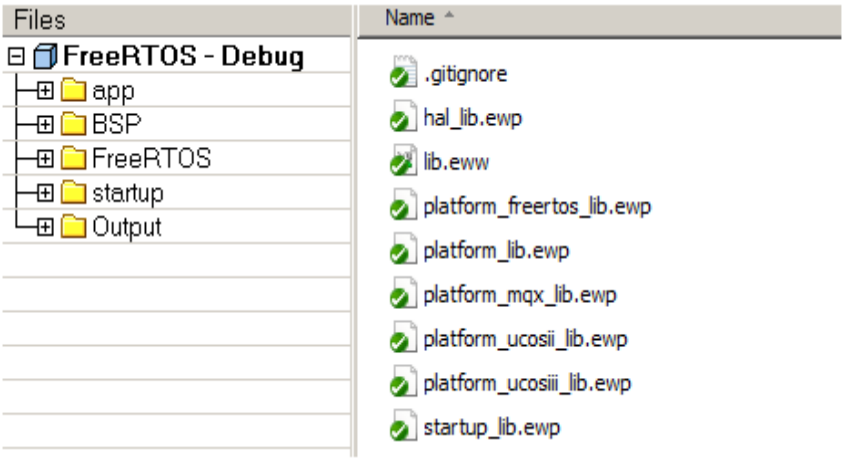

Figure-13 dragging the project into the IAR workspace

The workspace manages the platform_freertos_lib and the newly created project.

Figure-14 Platform_freertos_lib and new project managed in workspace

13. Rebuild the platform_freertos_lib.
14. Build your application project.

3 Revision history

This table summarizes revisions to this document.

Revision History	
Location	**Change description**
Entire document	Initial release – 1.0.0-beta

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: FREERTOSKSDKUG
Rev. 1.0.0-beta
03/2014