NXP Semiconductors
User's Guide

Document Number: KTSFSCIBQSUG
Rev. 4, 12/2016

Kinetis Thread Stack and FSCI Bootloader Quick Start Guide

1. Introduction

Contents

1. Introduction 1
2. Folder Locations 2
3. Deploying Standalone Bootloader Firmware 3
4. Building Thread Applications Binaries with Bootloader Offset with IAR IDE 4
5. Building Thread Applications Binaries with Bootloader Offset with KDS 6
6. Entering FSCI Bootloader Mode 8
7. Using the Host SDK Sample with the FSCI Bootloader 9
8. Revision History 10

Kinetis Thread Stack integrates with an Over-the-Wire (serial bus) bootloader with support for UART and SPI which uses the FSCI bus protocol. This process allows host devices to use the same serial bus protocol to drive both the bootloader and the THCI (Thread Host Controlled Interface) used by the Host Controlled Device examples.

2. Folder Locations

The following folder tree locations relative to the Thread package installation path contain artifacts showcasing the use of the FSCI bootloader:

- **\boards\frdmkw24\wireless_examples\thread\host_controlled_device** contains IAR and KDS example projects for the Host Controlled Device running on the FRDM-KW24D
- **\tools\wireless\binaries** contains binary image files for the bootloader variants of the FRDM-KW24D: FSCI and OTA update bootloaders
- **\boards\frdmkw24\wireless_examples\framework\bootloader_fsci** contains a common source tree and an IAR project for the bare-metal FSCI bootloader
- **\tools\wireless\host_sdk\hsdk-python\src\com\nxp\wireless_connectivity\test\bootloader** contains a Python language example script integrated with the Host SDK Linux® OS or Windows® OS which shows how to drive the FSCI bootloader from a host system
- **\tools\wireless\host_sdk\hsdk\demo** contains a C language example program integrated with the Host SDK Linux® OS which shows how to drive the FSCI bootloader from a host system
- **\docs\wireless\Common\Kinetis FSCI Host Application Programming Interface.pdf** contains a reference to deploying the host SDK for using samples with a FRDM-KW24D development board running the Thread Host Controlled Device Application firmware

3. Deploying Standalone Bootloader Firmware

To deploy the standalone FSCI bootloader to the FRDM-KW24D, choose one of the following options:

- From IAR® EWARM, deploy the Debug or Release configuration of the following workspace: **\boards\frdmkw24\wireless_examples\framework\bootloader_fsci\bm\iar\ bootloader_fsci_bm.eww** to the board using OpenSDA or J-Link pod debuggers. Make sure to select the appropriate driver under Options -> Debugger -> Setup -> Driver.
- Copy the **\tools\wireless\binaries\bootloader_fsci_frdmkw24.bin** file to the FRDM-KW24D mass storage device emulated via OpenSDA.
- Using a J-Link pod and a tool such as SEGGER J-Link Tools, IAR EWARM or NXP Connectivity Test Tool for Kinetis Firmware Loader, erase the MCU and then load the **\tools\wireless\binaries\bootloader_fsci_frdmkw24.bin** file to the device flash.

If the standalone bootloader is loaded to the first flash sector and there is an application firmware following in the flash, by default the bootloader will load the application, instead of waiting for new firmware. In order to return to the FSCI bootloader mode, one must use one of the options presented in Section 6 *Entering FSCI Bootloader Mode*.

To configure a standalone bootloader to automatically wait for new firmware, one must rebuild the bootloader binary file in IAR EWARM after setting the **gBootFlags** attribute to value 0x00UL in the file: **\boards\frdmkw24\wireless_examples\framework\bootloader_fsci\src\bootloader \main.c.**

Default value of **gBootFlags**:

```
#pragma location = "BootFlags"
__root volatile const bootFlags_t gBootFlags =
{
    0xFFFFFFFFFFFFFFFF
};
```

should be updated to

```
#pragma location = "BootFlags"
__root volatile const bootFlags_t gBootFlags =
{
    0x00UL
};
```

4. Building Thread Applications Binaries with Bootloader Offset with IAR IDE

To build a Thread application in IAR® IDE, which is offset to load starting at second sector of the flash, thus making the binary compatible with the FSCI bootloader, ensure the IAR Project Options for the Linker configuration match the settings below.

To build a bootloader-compatible application, for the Host Controlled Device, start with the workspace in: **\boards\frdmkw24\wireless_examples\thread\host_controlled_device\freertos\iar** and ensure gUseBootloaderLink_d is set to 1.

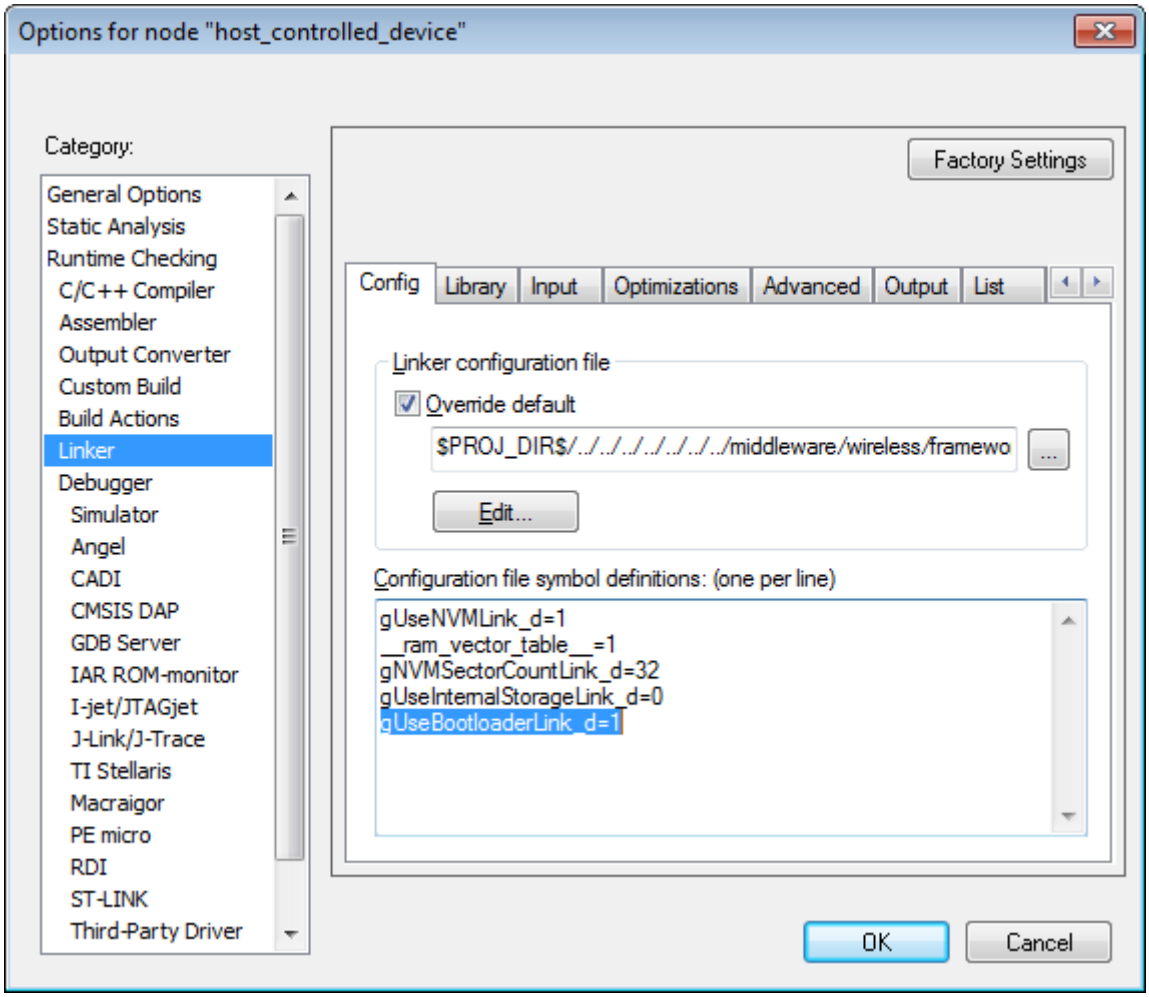

Figure 1. IAR Linker Configuration Options

- In the **Project Options → Output Converter** switch the **Output format** to **binary**:

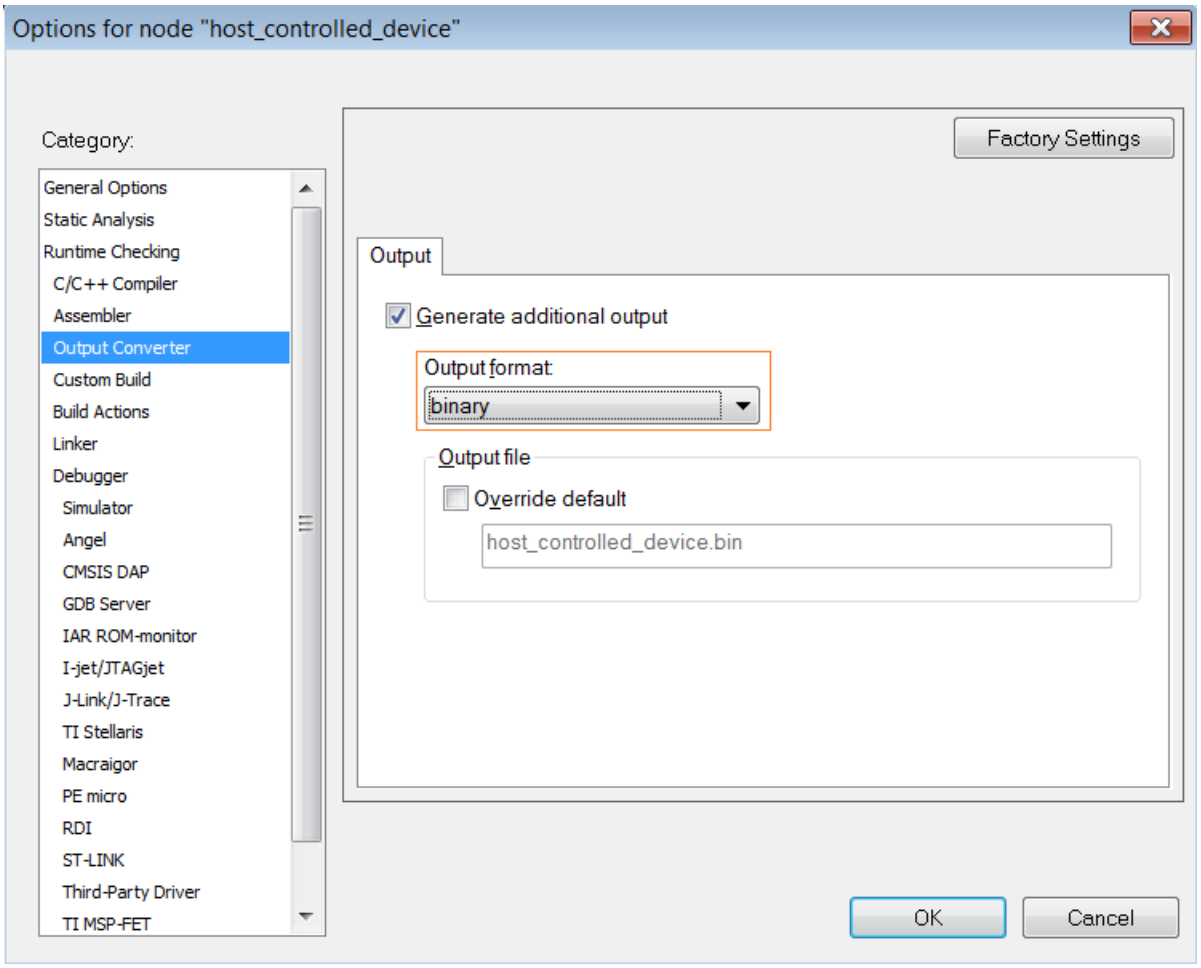

Figure 2. Project Options → Output Converter

- Build the project, the bootloader compatible *.bin file is generated in the output folder, e.g., at: **\boards\frdmkw24\wireless_examples\thread\host_controlled_device\freertos\iar\debug**

5. Building Thread Applications Binaries with Bootloader Offset with KDS

To build a Thread application in KDS, which loads starting at the second sector of the flash, making the binary compatible with the FSCI bootloader to ensure the KDS Project Options for the Linker configuration match the settings below.

To build a bootloader-compatible application, for the Host Controlled Device, start with the workspace in: **\boards\frdmkw24\wireless_examples\thread\host_controlled_device\freertos\kds** and ensure gUseBootloaderLink_d is set to 1.

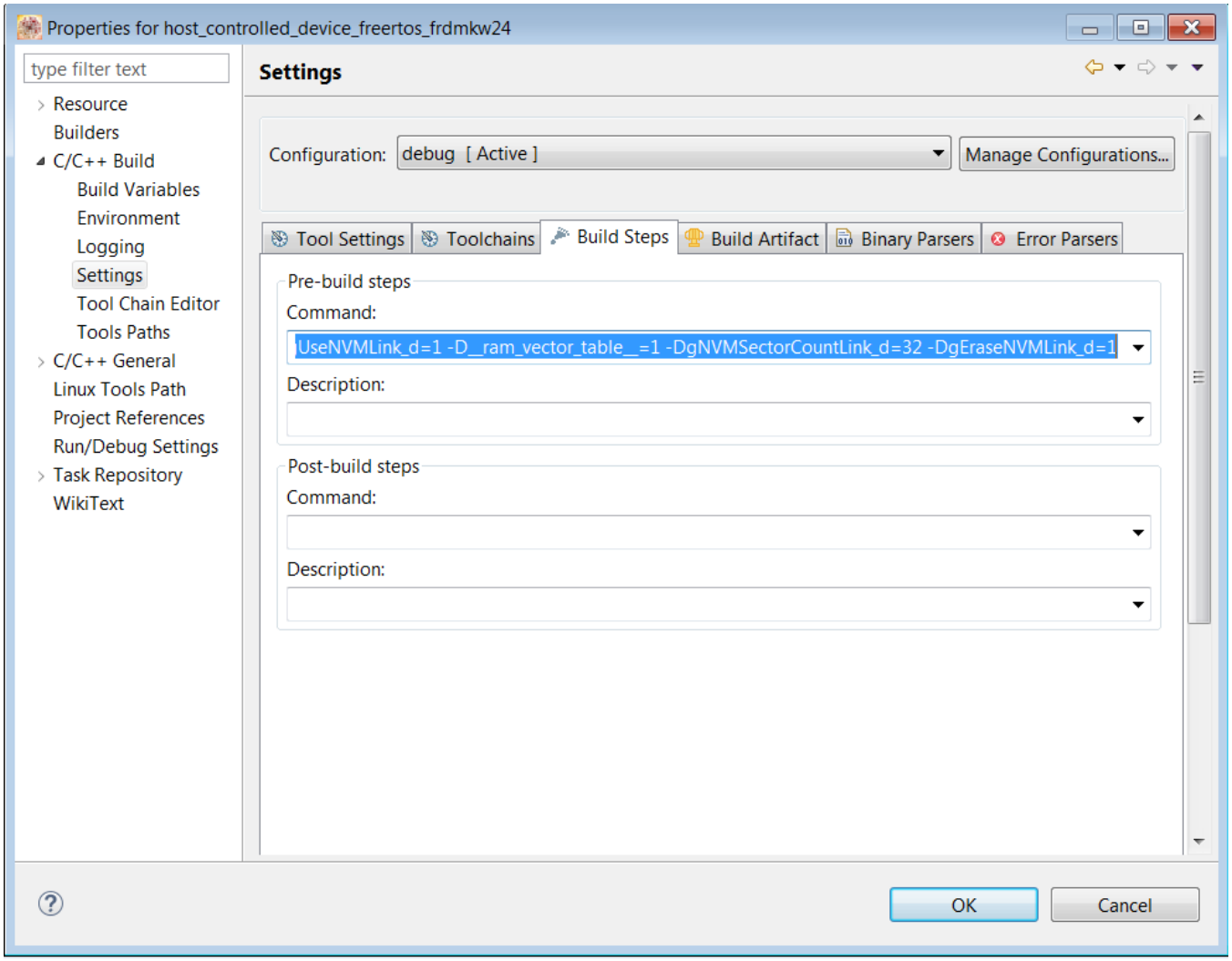

Figure 3. KDS Build Steps

- In the **Properties → C/C++ Build → Settings → Cross ARM GNU Create Flash Image → GENERAL Tab** switch the **Output file format (-O)** to **Raw binary**:

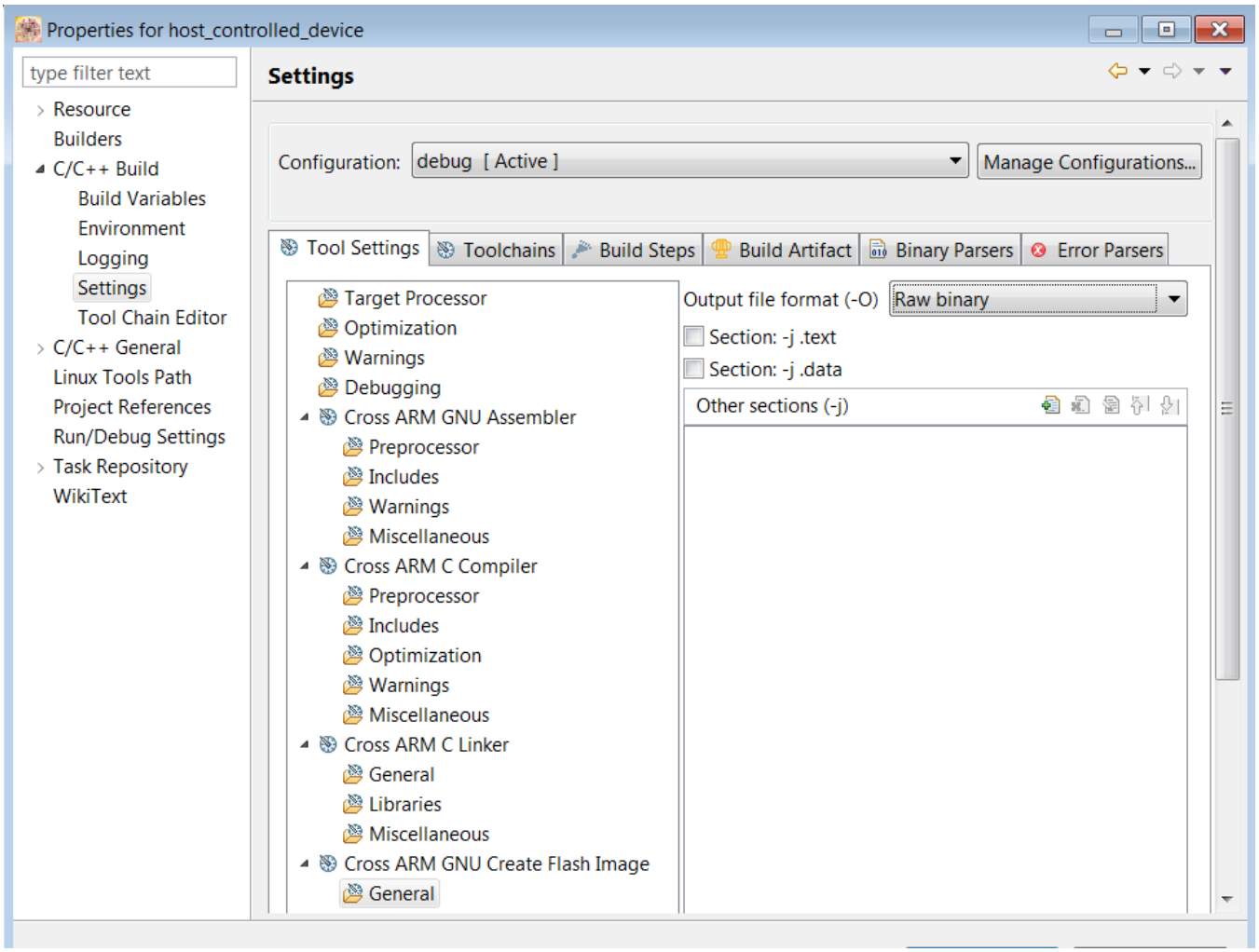

Figure 4. Cross ARM GNU Create Flash Image

- Build the project, the bootloader compatible *.bin file is generated in the output folder, for example, at:
  \boards\frdmkw24\wireless_examples\thread\host_controlled_device\freertos\kds\debug

6. Entering FSCI Bootloader Mode

After a valid application is programmed starting with the second flash sector, the bootloader jumps to the application automatically and the bootloader mode must be triggered specifically.

To trigger bootloader mode using the FRDM-KW24D, do the following:

- Hold RST and press SW1, then release RST first and SW1 second.
- If the THCI is active in the Thread firmware, send the command to enter bootloader mode over the THCI/FSCI serial bus; this command has:
    - THCI/FSCI OpCode Group: `0xA3`
    - THCI/FSCI OpCode: `0xCF`

  When receiving the command, the device auto-resets and remains in bootloader mode.

7. Using the Host SDK Sample with the FSCI Bootloader

To deploy the Host SDK to a host Linux system OS that works with the Thread THCI (Host Control Interface/Device), follow the deployment procedure in **\docs\wireless\Common\Kinetis FSCI Host Application Programming Interface.pdf**, then run the script at: **\tools\wireless\host_sdk\hsdk-python\src\com\nxp\wireless_connectivity\test\bootloader fsci_bootloader.py** providing as command line arguments the device serial port and a binary firmware file compatible with the bootloader which has been built as shown in the section above. For example,

```
export PYTHONPATH=$PYTHONPATH:<hsdk-path>/hsdk-python/src/
python fsci_bootloader.py /dev/ttyACM0 host_controlled_device_freertos.bin
```

The script does the following:

- Sends the THCI command for the device to reset, then enter and remain in bootloader mode.
- Sends the command to cancel an image as a safety check and to verify the bootloader is responsive.
- Sends the command to start firmware update for a new image.
- Pushes chunks of the firmware images file sequentially until the full firmware is programmed and display intermediate progress as percent of binary file content loaded.
- Sets the flags to commit the image as valid.
- Resets the device to ensure it boots to the new firmware. Additionally, one may use the C mirror of the script, located at **hsdk/demo/FsciBootloader.c**. The command line arguments are the same used for the Python script.

8. Revision History

This table summarizes revisions to this document.

Table 1. **Revision history**

Revision number	Date	Substantive changes
0	03/2016	Initial release
1	04/2016	Updates for the Thread KW41 Alpha Release
2	08/2016	Updates for the Thread KW41 Beta Release
3	09/2016	Updates for the Thread KW41 GA Release
4	12/2016	Updates for the Thread KW24D GA Release

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: KTSFSCIBQSUG
Rev. 4
12/2016