Getting Started with KSDK and FreeRTOS OS

Contents

1 Overview

Kinetis Software Development Kit (KSDK) 2.0 provides a comprehensive software package with pre-integrated FreeRTOS OS. Freescale provides FreeRTOS OS additions, such as RTOS driver wrappers, RTOS ready FatFs file system, and the implementation of FreeRTOS tickless mode. This document describes steps required to configure supported development tools used to build run, and debug applications with the FreeRTOS OS targeted for KSDK.

2 FreeRTOS OS role in KSDK

The Kinetis Software Development Kit (KSDK) is a software framework for developing applications on Kinetis MCUs. The software components in the framework include peripheral drivers, middleware and real time operating systems.

This version of KSDK provides FreeRTOS OS version 8.2.3. Detail information about the distribution can be found on www.freertos.org. Framework contains a set of examples which shows basic FreeRTOS OS features. This makes it easy to start a new FreeRTOS project or just begin experimenting with FreeRTOS OS. Selected drivers and middleware are RTOS ready with related FreeRTOS adaptation layer.

3 FreeRTOS source description

Core files Generic FreeRTOS core files are located in <KSDK_DIR>/rtos/freertos_8.2.3/Source. Header files are in subfolder <KSDK_DIR>/rtos/freertos_8.2.3/Source/include.

Port files FreeRTOS porting files are located in <KSDK_DIR>/rtos/freertos_8.2.3/Source/portable. FreeRTOS OS in KSDK supports ARM Cortex®-M0+ and Cortex®-M4 architectures.

Configuration files Configuration files are designed as application specific and its location is dependent on the application location. Usually it is in the example's root folder (for freertos_sem: <KSDK_DIR>/boards/<board>/rtos_examples/ freertos_sem). The main configuration file is FreeRTOSConfig.h.

4 FreeRTOS package integration in KSDK

- Removed files unrelated to KSDK.
    - Remove extensions to the FreeRTOS OS (CLI, FAT_SL, and UDP).
    - Reduce FreeRTOS folder structure (remove demo folder, remove nested folders).
- Add SystemCoreClock global variable to FreeRTOS port.c files
- Enabled tickless mode. For more information see: www.freescale.com/freertos

5 FreeRTOS drivers

Selected drivers provide FreeRTOS support in form of an additional layer. This solution enables simple driver integration in RTOS-based applications.

Drivers with FreeRTOS layers are:

- UART / LPUART / LPSCI
- I2C / LPI2C
- SPI / LPSPI

The drivers for FreeRTOS OS is a layer built on top of standard KSDK peripheral drivers to achieve multithread (RTOS) awareness. The wrappers provide an API which blocks the calling task until the I/O operation completes and allows other tasks to run in the background. This is achieved by using the asynchronous API of the underlying driver along with RTOS task synchronization objects. Underlying drivers require enabled interrupts for proper operation.

It is recommended to use the FreeRTOS drivers instead of KSDK bare metal drivers. The UART class drivers are demonstrated in the FreeRTOS UART examples. The example shows how to use UART class driver with the FreeRTOS OS for standard communication.

6 FreeRTOS Example Applications

KSDK provides a set of FreeRTOS OS-related applications. The examples are written to demonstrate basic FreeRTOS features and the interaction between peripheral drivers and the RTOS.

6.1   List of examples

Table 1.   List of examples

FreeRTOS examples	Driver examples
freertos_hello	freertos_uart
freertos_event	freertos_lpuart
freertos_sem	freertos_i2c
freertos_mutex	freertos_lpi2c
freertos_queue	freertos_spi
freertos_swtimer	freertos_dspi
freertos_generic	sdcard_freertos
freertos_tickless	mmcccard_freertos

6.2   Building a FreeRTOS example application

The FreeRTOS OS in KSDK is provided in the form of source files directly linked to the projects.

- <KSDK_DIR> is the directory where the Kinetis SDK package is installed on the hardware.
- <FREERTOS_DIR> is the directory where FreeRTOS OS is located within KSDK, specifically <KSDK_DIR>/rtos/freertos_8.2.3.
- <board> replaces the name of the board (e.g., TWR-K64F120M)
- <mcu> replaces the name of the processor (e.g., MK64F120M)
- <tool> replaces the name of the toolchain (e.g., IAR)
- <target> replaces the name of the project target (e.g., Debug)

6.3   Build a first FreeRTOS application

- Open the workspace file with the related FreeRTOS example located in <KSDK_DIR>/boards/<board>/rtos_examples /<example>/<tool>/<example>.
- Build the required targets (e.g., Debug) in all projects contained in the workspace.
- Run the application.

7   Revision History

This table summarizes revisions to this document.

Table 2. Revision history

Revision number	Date	Substantive changes
0	01/2016	Initial release

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support