Document Number: KSDK200MKS22FN256RN
Rev. 0, 12/2015

Kinetis SDK v.2.0.0 Release Notes Supporting the MKS22FN12 Devices

Contents

1 Overview

The Kinetis Software Development Kit (KSDK) 2.0.0 is a collection of software enablement for Kinetis Microcontrollers that includes peripheral drivers, high-level stacks including USB and LWIP, integration with WolfSSL and mbed TLS cryptography libraries, other middleware packages (multicore support, fatfs), and integrated RTOS support for FreeRTOS and uCOS. In addition to the base enablement, the KSDK is augmented with demo applications and driver example projects, and API documentation to help our customers quickly leverage the support of the Kinetis SDK.

2 What is New

KSDK 2.0 is the evolution of KSDK 1.x into a more optimized software solution. KSDK 2.0 eliminates the need for separate HAL and Peripheral drivers, replacing these two layers with a single driver for each peripheral. The single driver provides both the low-level functionality of the HAL and the non-blocking interrupt-based functionality of the Peripheral Driver, enabling customers to select the right level of abstraction for their solution. Peripheral drivers in KSDK 2.0 also eliminate external software dependencies; the Operating System Abstraction, Power Manager, and Clock Manager are no longer required by the KSDK 2.0 drivers.

At the middleware level, RTCS and MFS have been removed, and the USB stack has been replaced with a BSD licensed solution. KSDK 2.0 has also aligned with ARM® through the integration of mbed TLS with our accelerated cryptography drivers. This integration ensures the highest level of performance from our on-chip security peripherals.

Existing MQX RTOS support has been deprecated to focus on support of FreeRTOS and uCOs II/III.

Device family support

- KS22F12

Peripheral support

- ADC
- Clock
- CMP
- CRC
- DAC
- DMAMUX
- DSPI
- eDMA
- EWM
- Flash
- FlexCAN
- FlexIO, FlexIO-UART, FlexIO-I2C, FlexIO-SPI, and FlexIO-I2S
- GPIO, LLWU
- I2S/SAI
- LPI2C
- LPTMR
- LPUART, MCG
- OSC
- PIT
- PDB, PMC
- PORT, RCM
- RNGA
- RTC, SDSPI, SIM, SMARTCARD, SMC
- TPM
- UART
- WDOG

USB

- Enabled HID, CDC, MSD, Audio, and PHDC on both the device and host, and Video only on the device.

3 Development Tools

The Kinetis SDK 2.0.0 was compiled and tested with the following toolchains:

- Kinetis Design Studio (KDS) IDE v3.0
- IAR Embedded Workbench for ARM® v7.50.0
- MDK-ARM Microcontroller Development Kit (Keil)® v5.17 with keil.kinetis_KSxx_DFP.1.0.0.pack download from https://www.keil.com/dd2/pack/
- Makefiles support with GCC revision 4.9-2015-q3-update from ARM Embedded
- Atollic® TrueSTUDIO v5.4.0
- Segger® J-Link v5.10 download from http://www.segger.com/jlink-software.html

4 Supported Development Systems

Table 1 shows the boards and devices supported by this release. Boards and devices in bold face fonts were tested in this release.

Table 1. Supported MCU devices and development boards

Development boards	Kinetis MCU devices
MAPS-KS22 (Rev. A)	**MKS22FN256VLL12**, MKS22FN256VLH12, MKS22FN128VLL12, MKS22FN128VLH12

5 Release Contents

The following table provides an overview of the KSDK release package contents and locations.

Table 2. Release contents

Deliverable	Location
Boards	<install_dir>/boards/...
Demo applications	<install_dir>/boards/<mapsks22>/demo_apps/…
USB Demo applications	<install_dir>/boards/<mapsks22>/usb/…
Driver examples	<install_dir>/boards/<mapsks22>/driver_examples/...
RTOS examples	<install_dir>/boards/<mapsks22>/rtos_examples/…
Documentation	<install_dir>/doc/...
Middleware	<install_dir>/middleware/...
FATFS File System	<install_dir>/middleware/fatfs/...
Driver library, startup code and utilities	<install_dir>/devices/MKS22F12/...
Cortex Microcontroller Software Interface Standard (CMSIS) ARM Cortex®-M header files, DSP library source	<install_dir>/CMSIS/…
Linker control files for each supported tool chain	<install_dir>/devices/MKS22F12/linker/...
System-on-Chip (SoC) header files, Extension header files, and feature header files	<install_dir>/devices/MKS22F12/...
CMSIS-compliant startup code	<install_dir>/devices/MKS22F12/[arm,gcc,iar]/...
Peripheral Drivers	<install_dir>/devices/MKS22F12/drivers/...
Utilities such as debug console	<install_dir>/devices/MKS22F12/utilities
RTOS Kernel Code, RTOS abstraction implementations, and RTOS kernel folders	<install_dir>/rtos/...
Toolchain files	<install_dir>/tools
USB stack	<install_dir>/middleware/usb/...

6 Kinetis SDK Release Overview

The KSDK 2.0 release package contents are aligned with the silicon subfamily it supports. This includes the boards, CMSIS, devices, documentation, middleware, and RTOS support.

The device folder contains all of the available software enablement for that SoC subfamily. This folder includes clock specific implementation, device register header files, device register feature header files, a CMSIS derived device SVD, and the system configuration source files. Included with the standard SoC support are folders containing peripheral drivers, toolchain support, and a simple debug console.

The device-specific header files provide direct access to the Kinetis MCU peripheral registers. The device header file provides an overall SoC memory-mapped register definition. In addition to the overall device memory-mapped header file, the Kinetis SDK also includes the feature header file for each peripheral instantiated on the SoC.

The toolchain folder contains the startup code and linker files for each supported toolchain. The startup code is a simple CMSIS-compliant startup that efficiently transfers the code execution to the main() function.

6.1 Kinetis MCU platform support

The device folder contains all of the available software enablement for that SoC subfamily. This folder includes clock specific implementation, a device register header file, a device register feature header file, CMSIS derived device SVD, and the system configuration source files. Included with the standard SoC support are folders containing peripheral drivers, toolchain support, and a simple debug console.

The device-specific header files provide direct access to the Kinetis MCU peripheral registers. The device header file provides an overall SoC memory-mapped register definition. In addition to the overall device memory-mapped header file, the Kinetis SDK also includes the feature header file for each peripheral instantiated on the SoC.

The toolchain folder contains the startup code and linker files for each supported toolchain. The startup code is a simple CMSIS-compliant startup that efficiently transfers the code execution to the main() function.

6.2 Kinetis board support

The boards folder provides the board specific demo applications, driver examples, RTOS, and middleware examples.

6.2.1 Startup code

The Kinetis SDK includes simple CMSIS-compliant startup code for the supported Kinetis MCUs which efficiently deliver the code execution to the main() function. An application can either include the startup code directly in the project build environment or include a prebuilt startup code library for a cleaner project build environment.

6.2.2 Driver library

The Kinetis SDK provides a set of drivers for the Kinetis MCU product family on-chip peripherals. The drivers are designed and implemented around the peripheral hardware blocks rather than for a specific Kinetis MCU.

The Peripheral Drivers provide a set of easy-to-use interfaces that handle high-level data and stateful transactions. They are designed for the most common use cases identified for the underlying hardware block and are reasonably efficient in terms of memory and performance. They are written in C language and can be easily ported from product to product as they are designed to be initialized at runtime based on the driver configuration passed in by the user. In most cases, the Peripheral Drivers can be used as is. However, if the Peripheral Driver does not address a particular target use case, it can either be modified/enhanced or completely rewritten to meet the target functionality and other requirements. In this case, the existing Peripheral Driver can be used as a reference to build a custom driver based on the CMSIS register macro.

Detailed implementation of hardware peripheral functionality is implemented in stages. The features which are missing from the current driver versions may be implemented in future releases.

6.2.3 Header files

The Kinetis SDK devices directory contains device-specific header files which provide direct access to the Kinetis MCU peripheral registers. Each supported Kinetis MCU device in the Kinetis SDK has an overall SoC memory-mapped header file. In addition to the overall SoC memory-mapped header file, the feature header files for each peripheral are instantiated on the Kinetis MCU. Along with the SoC header files and feature header files, the Kinetis SDK CMSIS directory includes common CMSIS header files for the ARM Cortex®-M core and DSP library from the ARM CMSIS version 4.5 release.

6.2.4 Linker files

The Kinetis SDK devices directory contains linker control files (or simply linker files) for each supported tool chain and Kinetis MCU device.

6.2.5 Utilities

The platform/utilities directory contains useful software utilities such as a debug console.

6.2.6 Demo applications and driver examples

The demo applications demonstrate the usage of the peripheral drivers to achieve a system level solution. Each demo application contains a readme that specifies the operation of the demo and required setup steps.

The driver examples demonstrate the capabilities of the peripheral drivers. Each example implements a common use-case to help demonstrate driver functionality.

The RTOS and middleware folders each contain examples demonstrating the use of the included source.

6.3 Middleware

6.3.1 USB stack

A Freescale USB stack is integrated with the Kinetis SDK and supprts both bare metal and RTOS setups. For more details about the Freescale USB stack, see the Integration of the USB Stack and Kinetis SDK (document USBKSDKUG).

6.3.1.1 Tested USB devices

- HUB
- Flash driver
- Card reader
- Mouse
- Keyboard

6.3.2 File system

A FAT file system is integrated with Kinetis SDK and can be used to access the USB memory stick when the USB Mass Storage Device class implementation is used.

6.4 RTOS

The Kinetis SDK is pre-integrated with FreeRTOS, µC/OS-II, and µC/OS-III.

6.5 CMSIS

The Kinetis SDK is shipped with the standard CMSIS development pack, including the pre-built libraries.

6.6 Documentation

7 Known issues

7.1 Maximum file path length in Windows® 7 Operating System

Windows 7 operating system imposes a 260 character maximum length for file paths. When installing the Kinetis SDK, place it in a directory close to the root to prevent file paths from exceeding the maximum character length specified by the Windows operating system. The recommended location is the `C:\Freescale` folder.

7.2 Update shell script in Linux OS

After unzipping the release package in Linux OS, the shell script under <demo_name>/armgcc requires run access mode and updates to the unix file format using the command "chmod +x *.sh; dos2unix *.sh".

7.3 LPUART parity check limitation

When the LPUART parity check is enabled, it only supports 7 bit data transfer because the MSB bit (bit7) is used for a parity check. After reading the data, the user needs to discard the bit7 like the following: LPUART_ReadBlocking(LPUART0, &ch, 1); ch = ch&0x7F.

8 Change log

9 Revision History

This table summarizes revisions to this document.

Table 3. Revision History

Revision number	Date	Substantive changes
0	12/2015	Initial release

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number KSDK200MKS22FN256RN
Revision 0, 12/2015

ARM® POWERED

freescale™