Document Number: KBTLDR200RN
Rev. 0, 04/2016

Kinetis Bootloader v2.0.0 Release Notes

Contents

1 Overview

These are the release notes for the Kinetis bootloader v2.0.0. For more information and getting started instructions, see the Getting Started section of this document.

The Kinetis bootloader is an application that you program into the internal flash memory of a Kinetis device. The bootloader is designed to detect communication traffic on one of the supported peripherals (USB-HID, USB-MSC, UART, SPI, I2C, and CAN), download a user application, and write the application to internal flash. The bootloader stays resident on flash along with your application.

This release includes the PC-hosted Kinetis Flash Tool application. This application allows to you choose a device application image and send it to the bootloader over USB-HID or UART.

2 Development tools

The Kinetis bootloader 2.0 was compiled and tested with these development tools.

Firmware projects:

- Kinetis Design Studio (KDS) IDE v.3.2.0
- IAR Embedded Workbench for ARM® v7.50.1

NOTE

The IAR tool binary path must be added to the system environment path. For example, C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.5\arm\bin.

Host projects:

- Microsoft Visual Studio® Professional 2015 for Windows® OS Desktop
- Microsoft NET Framework v4.5 (included in Windows OS 8)
- Microsoft Visual Studio C++ Redistributable for Visual Studio 2013 (vcredist_x86.exe)
- Python v2.7 (www.python.org)
- Keil MDK v5.18 with corresponding packs

NOTE

The Python path must be added to the system environment path. For example, C:\Python27.

- Apple Xcode® v7.3 (for the blhost and elftosb tools)
- Linux® OS GNU Compiler (GCC) v4.8.1 (for the blhost tool), libstdc++6, libudev-dev, libc6, and libgcc1 (for the blhost tool) for the Linux build. Running blhost on Linux OS requires the following libraries to be installed: libstdc++.so.6, libudev.so.1, libc.so.6, libgcc_s.so.1, and libpthread.so.0
- Linux OS tools have been tested on Ubuntu 14.04 LTS
- Apple Mac® OS host tools have been tested on Mac OS 10.11.4

3 System requirements

System requirements are based on the requirements for the development tools and the Kinetis Flash Tool application.

The recommended PC configuration is 2 GHz processor, 2 GB RAM, and 2 GB free disk space.

Windows OS applications like QCBGenerator.exe require installation of Visual C++ redistributable 2013 or greater.

.Net framework 4.5

- JP version: www.microsoft.com/ja-JP/download/details.aspx?id=30653
- US version: www.microsoft.com/en-US/download/details.aspx?id=30653

VS++ redistributable 2013

- JP version: www.microsoft.com/ja-JP/download/details.aspx?id=40784
- US version: www.microsoft.com/en-us/download/details.aspx?id=40784

4 Target requirements

This release of the Kinetis bootloader supports the following platforms:

- FRDM-K64F Freedom Development platform
- FRDM-KL25Z Freedom Development platform
- FRDM-KV31F Freedom Development platform
- FRDM-K82F Freedom Development platform
- FRDM-KL82Z Freedom Development platform
- FRDM-K66F Freedom Development platform
- FRDM-KL28T Freedom Development platform
- MAPS-KS22F256 MAPS Development platform

- TWR-K22F120M Tower System module
- TWR-K64F120M Tower System module
- TWR-KV46F150M Tower System module
- TWR-KV31F120M Tower System module
- TWR-K24F120M Tower System module
- TWR-K65F180M Tower System module
- TWR-KV11Z75M Tower System module
- TWR-K80F150M Tower System module
- TWR-KL82Z72M Tower System module
- TWR-KL28Z72M Tower System module
- TWR-KV58F220M Tower System module
- TWR-KL25Z48M Tower System module

There are no special requirements for the hardware other than what the board requires to operate.

5 Release contents

This table describes the release contents.

Table 1. Release contents

Deliverable	Location
Host flash tool and demo applications	<install_dir>/apps/...
Host binaries and utilities	<install_dir>/bin/...
Documentation	<install_dir>/doc/...
Bootloader and host tools source code	<install_dir>/src/...
Tool chain build projects	<install_dir>/targets/...
Host validation tools build projects	<install_dir>/validation/...

6 Getting started

See the *Kinetis Bootloader Demo Applications User's Guide* (document KBTLDRDEMOUG) for a description of how to use the Kinetis bootloader to load a user application on the Kinetis MCUs.

See the *Getting Started with Kinetis ROM Bootloader* (document KBTLDRUG) for Kinetis ROM-specific information.

See the *Getting Started with Kinetis Flashloader* (document KFLLDRUG) for Kinetis Flashloader-specific information.

For porting information, see Chapter 10, "Kinetis bootloader porting" in the *Kinetis Bootloader v2.0.0 Reference Manual* (document KBTLDR200RM).

For customization, see Chapter 11, "Create a custom flash-resident bootloader" in the *Kinetis Bootloader v2.0.0 Reference Manual* (document KBTLDR200RM).

7 Features

The bootloader release contains source code and toolchain projects for building flash-resident bootloaders and flashloaders for the supported platforms (see Section 4, “Target Requirements”). A flash-resident bootloader stays resident in flash along with the user application. The flash-resident bootloader can be used to download and program an initial application image into a blank area on the flash, and to later update the application. In contrast, a flashloader gets replaced in flash by the user application and thus is a one-time programming aid.

The Kinetis bootloader supports the following communication interfaces for downloading an application. Not all interfaces are supported on all platforms. See the individual platform reference manual for supported interfaces.

- USB-HID
- USB-MSD
- UART
- I2C
- SPI
- CAN

Usually, USB-HID, USB-MSC, and UART connections are made directly to a PC, whereas I2C, SPI, and CAN require additional hardware. The bootloader, running on the target platform, acts as a communication slave. The bootloader can automatically detect which peripheral is being used to download the application and, for UART and CAN, automatically detect the baud rate.

The application image is downloaded to the target through a series of command and data packets sent from a host PC or embedded host platform.

8 Host tools

The bootloader release contains source code and build projects for the following PC-based host tools:

- blhost - command line debug tool to send individual commands to the bootloader.
- KinetisFlashTool - GUI application to download and flash an application image.
- Elftosb - command line tool to convert ELF formatted application image to SB format.
- Mfgtool2 - GUI application to be used in factory production.

For more information, see the *Kinetis blhost User’s Guide* (document KBLHOSTUG) and *Kinetis Flash Tool User’s Guide* (document KFLASHTOOLUG), the *Kinetis Elftosb User's Guide* (document KBLEFLTOSBUG) , and the *Manufacturing Tool v2.0.0 (Mfgtool2) for Kinetis Bootloader User's Guide* (document KBLMFGTOOLUG).

9 New features

The following new features were introduced in this release:

- Addition of new supported platforms as indicated in Section 4, “Target Requirements”.
- Support for the latest version of KDS toolchain.
- Addition of embedded_host example.
- The Kinetis Updater host tool has been replaced by the Kinetis Flash Tool.
- The Elftosb host tool has been added.
- The Embedded Host example has been added.
- Support for the Manufacturing Tool has been added.
- Support for the KEIL toolchain has been added.
- The QSPI demo and QCBGenerator tool have been added.
- Support has been added for a Reliable Update feature.
- The Receive-sb-file bootloader command is now fully supported.

10 Fixed issues

The following issues were fixed since the previous release:

- Compiler warnings for the KDS tool chain have been fixed.

11 Known issues

The following are known issues with this release:

- The KDS IDE tool chain does not support debugging or downloading the application image via CMSIS-DAP yet. Instead, use J-Link, replace the CMSIS-DAP firmware with J-Link-CDC firmware on the target platform.
- When using USB-MSC (drag-and-drop to an attached target), the USB connection may abort with a timeout if the SB file contains a command to erase QSPI. Use a different peripheral interface to erase the QSPI, such as USB-HID.

12 Tool notes

- When changing an IAR project to generate additional output (Options->Output Converter->Generate additional output), the output filename extension (Linker->Output->Output filename) must be changed to '.out'.
- The blhost tool accepts any speed setting in the "--buspal" option for the SPI and I2C peripherals. However, the maximum effective speed settings when using the BusPal example are approximately 300 kHz for I2C and 8000 kHz for SPI.
- When running blhost on Mac and Linux OS, it may be required to run as super user so that the tool have access to the device ports, i.e., 'sudo blhost'.
- When running blhost on Linux OS, it may be necessary to press the reset button on the target platform after the CDC device has been enumerated by the OS. This restarts the bootloader's UART autobaud operation.
- When running on Mac OS, open the "cu" device instead of the "tty" device. This prevents the OS from trying to initialize the target as a modem, i.e., 'blhost -p /dev/cu.usbmodem*'
- When using KDS, load projects using File->Import->Existing Projects into Workspace. Do not use File->Import->Projects of Projects.

Each bootloader target supports one or more of the following project types:

1. tower bootloader - bootloader designed to execute from target flash memory on the Tower platform.
2. freedom bootloader - bootloader designed to execute from target flash memory on the Freedom platform.
3. maps bootloader – bootloader designed to execute from target flash memory on the MAPS platform.
4. flashloader – bootloader designed to execute from target RAM memory on either the Freedom, Tower, or MAPS platform.
5. flashloader_loader – bootstrap loader designed to execute from flash memory on either the Freedom or Tower platform. This loader copies an image of the flashloader into RAM, then executes the flashloader from RAM.

The flashloader_loader project uses the output of the flashloader build to create the flashloader image to load into RAM. For this reason, the flashloader project must be built before building the flashloader_loader project.

When debugging flashloader with ARM Keil MDK compiler, there is no need to use the load button. The "Start/Stop Debug Session" button correctly downloads the flashloader image into the target RAM address and starts executing it. However, the flashloader_loader and flash-resident bootloader are required to use the load button to properly program the internal flash.

There is an intermittent issue with ARM Keil MDK when it uses J-Link to load the image. An error message pops up indicating flash download failed. The workaround to this issue is to load the Keil-built image separately, using J-Link, and not through Keil.

13 Revision history

The following table contains a history of changes made to this document.

Table 2. Revision history

Revision number	Date	Substantive changes
0	04/2016	Kinetis bootloader v2.0.0 initial release

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: KBTLDR200RN
Rev. 0
04/2016