

Getting Started with the Kinetis Flashloader

1 Introduction

This document describes how to interface with the Kinetis flashloader to program a user application image into the on-chip flash. The Kinetis flashloader application is preprogrammed into the Kinetis flash during manufacturing and enables flash programming without the need for a debugger.

2 Overview

This document describes the steps required to program a user application image into the Kinetis on-chip flash memory, utilizing the standardized Kinetis bootloader command interface. From the factory, the device boots from flash memory and loads the Kinetis flashloader into RAM. Running from RAM, the flashloader has access to the entire flash array for placement of the user application. After the user application is programmed into flash memory, the Kinetis flashloader is no longer available.

2.1 Kinetis flashloader

Contents

1	Introduction.....	1
2	Overview.....	1
2.1	Kinetis flashloader	1
2.2	Host utility.....	2
3	The Kinetis flashloader application.....	2
3.1	Connecting to the Kinetis platform.....	2
3.2	The host utility application - blhost.....	3
3.3	Flashing the user application.....	3
4	User application: vector table offset.....	4
5	User application: Flash Configuration Area.....	4
6	Revision History.....	4

The Kinetis flashloader application

The Kinetis bootloader is a standard bootloader for all Kinetis devices. It provides a standard interface to the device using any of the peripherals supported by the bootloader on a given Freescale Kinetis device.

The Kinetis flashloader is a specific implementation of the Kinetis bootloader. For the flashloader implementation, the Kinetis bootloader command interface is packaged as an executable that is loaded from flash and executed from RAM. This configuration allows the user application to be placed at the beginning of the on-chip flash where it is automatically launched upon boot from flash.

Other Kinetis bootloader implementations include a ROM-based bootloader and a flash-resident bootloader. The Kinetis bootloader is available as source code for custom, flash-based implementations. Example applications are provided to demonstrate how to interface with the bootloader.

Using the Kinetis flashloader to program a user application to the beginning of the Kinetis flash makes this implementation of the bootloader a one-time programming aid.

Developers creating a manufacturing flow for their hardware and software implementations may find it necessary to restore the Kinetis flashloader such that the device works as it did from the Freescale factory. To accomplish this, use an external debugger to program the flashloader_loader.bin file included in this package to the Kinetis on-chip flash. The exact method for doing this varies depending on hardware design and available tools.

2.2 Host utility

The blhost utility is an example host program used to interface with devices running the Kinetis bootloader. It can list and request execution of all of the commands supported by a given Kinetis device running the bootloader.

NOTE

The blhost application is released as part of Kinetis bootloader release package available on www.freescale.com/KBOOT. The blhost application is available in the <install_dir>/bin folder of the release package.

3 The Kinetis flashloader application

The Freescale Kinetis platform must be connected to a host computer to interface with the Kinetis flashloader application. After the platform is connected, use the blhost application to program a user application into the Kinetis flash memory.

3.1 Connecting to the Kinetis platform

The Kinetis flashloader supports UART and USB connections to a computer. See the Kinetis Reference Manual for the specific device to determine which peripherals are supported by the flashloader application and how the signals are routed to the pins of the Kinetis platform. After the Kinetis platform is powered up and there is a physical serial/USB connection between the Kinetis platform and host, the Kinetis device should be ready to receive commands.

For this example, a Kinetis device is connected to a serial-to-USB converter that enumerates on a Windows® operating systems PC as a Serial Port on COMxx.

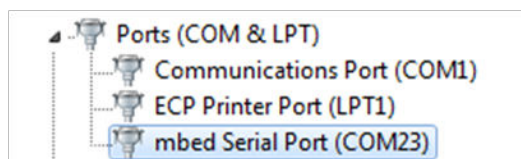


Figure 1. UART connection to Kinetis platform



Figure 2. Alternate UART connection to Kinetis platform

3.2 The host utility application - blhost

This section describes a simple usage of the blhost host utility program to demonstrate communication with the Kinetis bootloader.

- Open a command prompt in the directory containing blhost. For Windows OS, it is <install_dir>/bin/win.
- Type *blhost --help* to see the complete usage of the blhost utility.

For this step, it is important to verify that the Kinetis device is properly connected and is running the flashloader firmware application.

- It is assumed that the Kinetis platform is fresh out of reset.
- Note the COM port that the Kinetis platform is connected to as we did earlier. For this example, the device is connected to COMxx.
- Type *blhost -p COMxx -- get-property 1* to get the flashloader version from the Kinetis flashloader .
- An image similar to this screen shot shows that blhost is successfully communicating with the Kinetis platform.

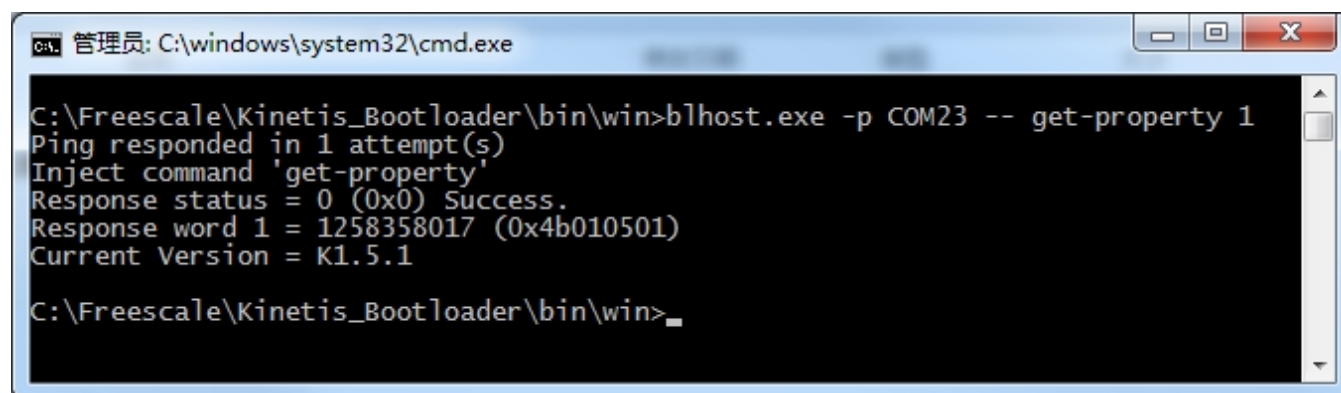


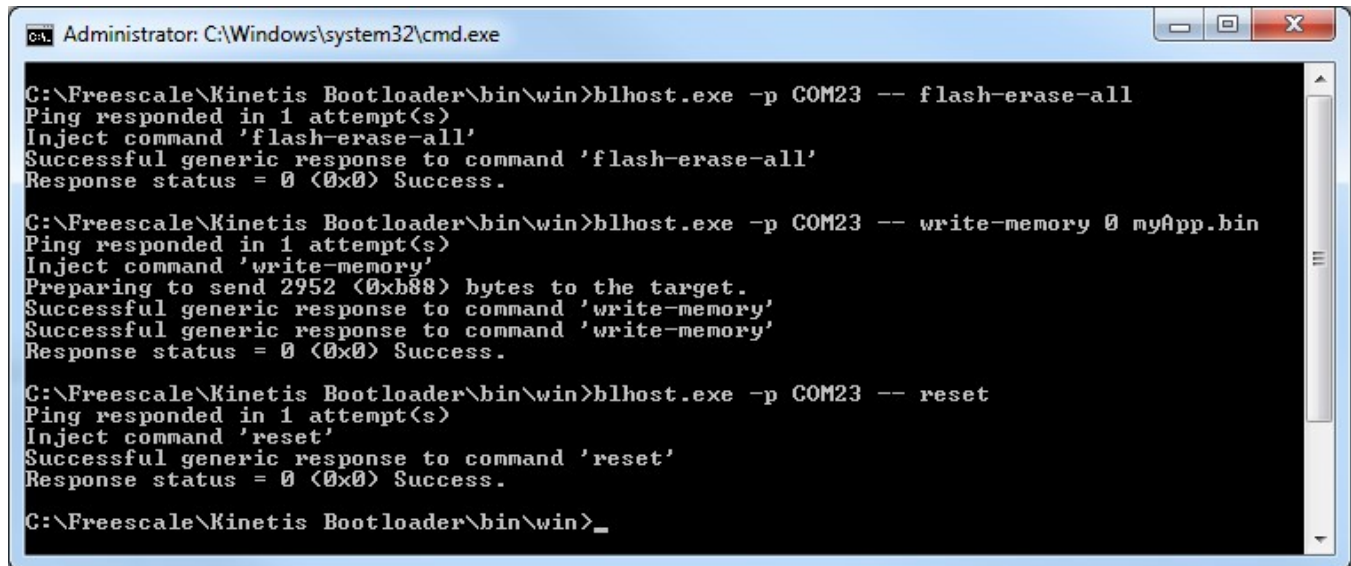
Figure 3. Host communication with Kinetis flashloader

3.3 Flashing the user application

Now that communications have been established between the Kinetis flashloader and the host, issue two commands to program the Kinetis flash memory with a user application.

User application: vector table offset

- `blhost -p COMxx -- flash-erase-all` - Erases the entire flash array.
- `blhost -p COMxx -- write-memory 0 myApp.bin` - Writes the myApp.bin binary image to address 0 of the Kinetis flash memory.
- [Optional] `blhost -p COMxx -- reset` - Resets the Kinetis platform and launches the user application. Note the Kinetis flashloader is no longer running on the device, so further commands issued from the blhost utility fail.
- After issuing the reset command, allow 5 seconds for the user application to start running.
- A screen shot similar to this image shows the successful completion of the above commands.



```
C:\Freescall\Kinetis Bootloader\bin\win>blhost.exe -p COM23 -- flash-erase-all
Ping responded in 1 attempt(s)
Inject command 'flash-erase-all'
Successful generic response to command 'flash-erase-all'
Response status = 0 (0x0) Success.

C:\Freescall\Kinetis Bootloader\bin\win>blhost.exe -p COM23 -- write-memory 0 myApp.bin
Ping responded in 1 attempt(s)
Inject command 'write-memory'
Preparing to send 2952 (0xb88) bytes to the target.
Successful generic response to command 'write-memory'
Successful generic response to command 'write-memory'
Response status = 0 (0x0) Success.

C:\Freescall\Kinetis Bootloader\bin\win>blhost.exe -p COM23 -- reset
Ping responded in 1 attempt(s)
Inject command 'reset'
Successful generic response to command 'reset'
Response status = 0 (0x0) Success.

C:\Freescall\Kinetis Bootloader\bin\win>_
```

Figure 4. Programming a user application using the Kinetis flashloader

4 User application: vector table offset

Section 3 discusses how to program the Kinetis flash memory with the user application, myApp.bin. When creating the user application, the vector table of the application must be located at the beginning address of the flash memory region.

When booting from flash, the Kinetis device considers offset 0 the initial stack pointer and offset 4 the entry point for the application.

5 User application: Flash Configuration Area

The Flash Configuration Area (0x400-0x40F) should be carefully populated with known values according to the Reference Manual for the specific Kinetis platform. In particular, values for the FSEC (0x40C) and FOPT (0x40D) locations may prevent future writes to the Kinetis flash. Extra attention to ensure the correct values in your application image at these offsets is highly recommended. If your code (other than the vector table) is linked to begin at offset 0x410, then the default erased value (0xFF) of these locations makes the device secure, but mass erase is then enabled.

6 Revision History

This table summarizes revisions to this document.

Table 1. Revision History

Revision number	Date	Substantive changes
0	12/2014	Initial release
1	07/2015	<ul style="list-style-type: none">• Added content to Section 3.3• Kinetis Bootloader 1.2.0 updates

How to Reach Us:**Home Page:**freescale.com**Web Support:**freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

Freescale, the Freescale logo, and Kinetis are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. All rights reserved.

© 2015 Freescale Semiconductor, Inc.