

# Getting Started with LPC540xx Flashloader User's Guide



# Contents

**Chapter 1 Introduction..... 3**

**Chapter 2 Overview..... 4**

2.1 The LPC540xx platform..... 4

2.2 DFU Utility usage..... 4

2.3 Bootloader host utility (blhost)..... 4

**Chapter 3 Device running flashloader..... 5**

3.1 Testing flashloader execution using blhost..... 5

3.2 Flashing the user application..... 6

3.2.1 Configuring external Quad SPI flash..... 6

3.2.2 Programming application to Quad SPI memory..... 7

**Chapter 4 Quad SPI configuration structure..... 9**

**Chapter 5 One Time Programmable (OTP) memory..... 11**

5.1 efuse-read-once..... 11

5.2 efuse-program-once..... 12

**Chapter 6 Revision history..... 13**

# Chapter 1

## Introduction

This document describes how to interface with the LPC540xx flashloader to program a user application image into the external SPI flash for the LPC540xx device. The flashloader is the secondary bootloader program that can be loaded into the on-chip RAM of LPC device. The flashloader application provides a programming interface over serial USB peripheral in FS or HS mode. HS mode is enabled by default. A host application can send the commands and data to be programmed on the device's external Quad SPI flash. The flashloader also provides commands to program the internal fuse array (One Time Programmable memory) to configure the device. In this document, flashloader is also referred to as bootloader or secondary bootloader.

## Chapter 2

# Overview

This guide describes the steps required to use the flashloader program to program external Quad SPI memory connected to the LPC device and internal fuse array.

## 2.1 The LPC540xx platform

The NXP LPC540xx platform must be connected to a host computer to load and interface with the flashloader application. On a device with blank external flash, LPC540xx always enumerates on the host device as a USB DFU mode device. This is the default ISP function of the ROM code for the LPC540xx device. For a non-blank device, see Chapter 3, "LPC540xx Boot ROM" in [UM11060 LPC540xx User Manual](#) for instructions on how to boot and connect the device with the host in DFU mode.

## 2.2 DFU Utility usage

The DFU Utility is the host application used to load the flashloader binary into the internal RAM memory of LPC540xx device connected to the host in USB DFU mode. "dfu-util.exe" is an opensource command line application and is available on Windows® OS, Unix, and Mac® OS platforms. It can be downloaded from [dfu-util.sourceforge.net/releases/](https://dfu-util.sourceforge.net/releases/). The following is the command line to load the flashloader.bin:

```
$ dfu-util.exe -D flashloader.bin
```

The DFU Utility prints out messages on standard output to indicate whether flashloader.bin successfully loaded or not.

## 2.3 Bootloader host utility (blhost)

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

## Chapter 3

# Device running flashloader

Once the flashloader binary is downloaded on the device connected in USB DFU mode and starts its execution on the LPC540xx platform and there remains a physical USB connection between the device platform and host, the flashloader will be ready to receive the commands.

For this example, we have an LPC540xx device running flashloader.bin connected over USB that enumerates on a Windows PC as a HID compliant device.

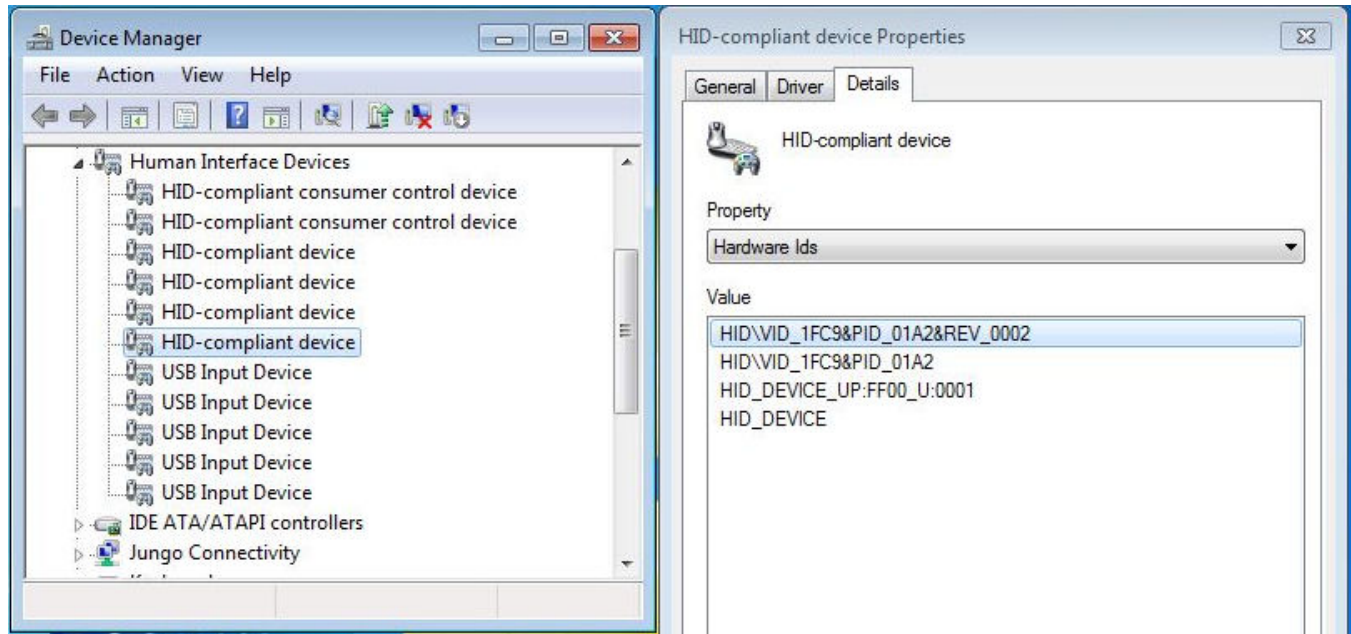


Figure 1. USB connection to LPC540xx platform running flashloader application

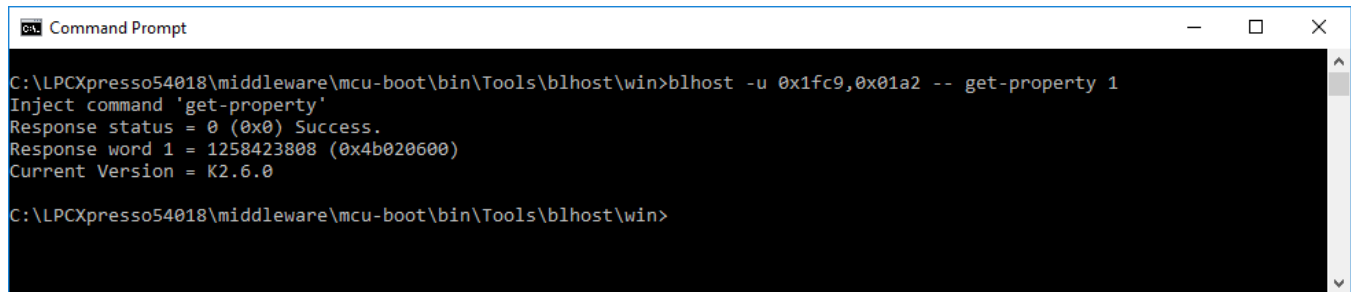
### 3.1 Testing flashloader execution using blhost

This section describes a simple usage of the blhost host utility program to demonstrate communication with the LPC540xx flashloader.

- Open a command prompt in the directory containing blhost. For Windows OS, it is <sdk\_package>/middleware/mcu-boot/bin/Tools/blhost/win.
- Type `blhost --help` to see the complete usage of the blhost utility.

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

- Note the device's USB vendor and product identifiers (VID and PID) as shown in the previous figure. The VID and PID are provided to identify our device with blhost when sending commands to the device.
- Type `blhost -u 0x1fc9,0x01a2 -- get-property 1` to get the bootloader version from the flashloader application.
- You should see something similar to the below figure, which indicates that blhost.exe is successfully communicating with the flashloader.



```

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- get-property 1
Inject command 'get-property'
Response status = 0 (0x0) Success.
Response word 1 = 1258423808 (0x4b020600)
Current Version = K2.6.0

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>

```

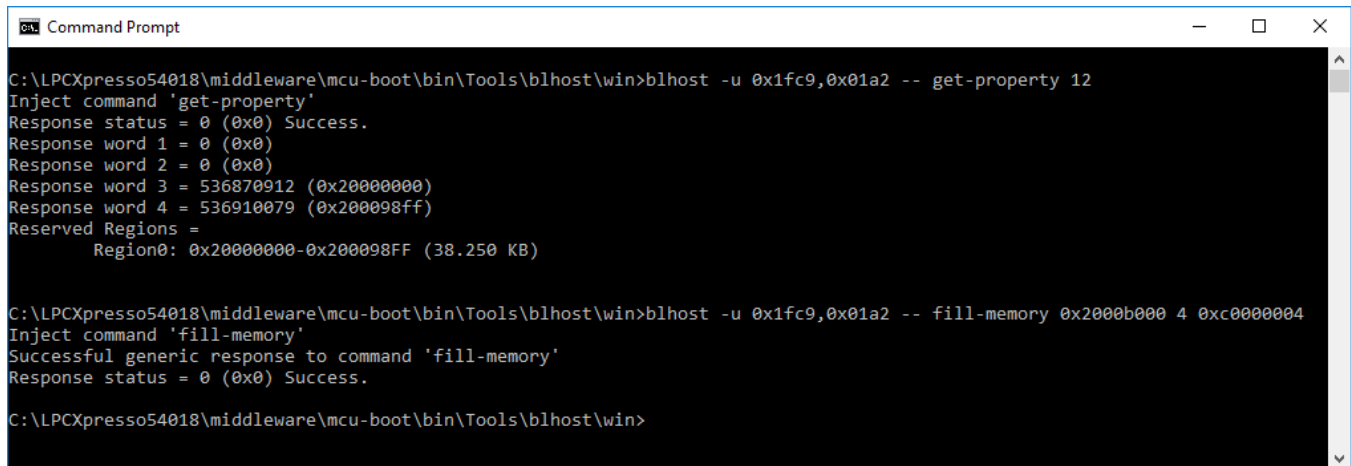
Figure 2. Host communication with ROM bootloader

## 3.2 Flashing the user application

After establishing communication between the flashloader and the host, use blhost commands to configure the external Quad SPI flash memory and program the demo application on the external Quad SPI memory.

### 3.2.1 Configuring external Quad SPI flash

Before accessing the Quad SPI flash memory, the SPIFI peripheral interface needs to be configured to the correct type of external Quad SPI flash present on the device. The flashloader command `configure-memory` is used for configuring, initializing, and preparing the SPIFI peripheral interface. The blhost application can be used to send the `configure-memory` command. The command requires two arguments, `memory-id` and `address-of-spi-nor-config-option-block`. The `memory-id` for Quad SPI flash is `0xA`. The address should point to the appropriate SRAM location where the configuration options block data is written prior to calling the `configure-memory` command. The bootloader commands, `write-memory` or `fill-memory`, can be used to set the configuration data for the Quad SPI at the SRAM address. The following image shows the `fill-memory` command being used to fill the 4 bytes at address `0x2000b000` with configuration data `0xc0000004`. The configuration structure is described in [Chapter QuadSPI configuration structure](#).



```

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- get-property 12
Inject command 'get-property'
Response status = 0 (0x0) Success.
Response word 1 = 0 (0x0)
Response word 2 = 0 (0x0)
Response word 3 = 536870912 (0x20000000)
Response word 4 = 536910079 (0x200098ff)
Reserved Regions =
    Region0: 0x20000000-0x200098ff (38.250 KB)

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- fill-memory 0x2000b000 4 0xc0000004
Inject command 'fill-memory'
Successful generic response to command 'fill-memory'
Response status = 0 (0x0) Success.

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>

```

Figure 3. fill-memory command

The following example shows the call to `configure-memory` command:

```

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- configure-memory 0xa 0x2000b000
Inject command 'configure-memory'
Successful generic response to command 'configure-memory'
Response status = 0 (0x0) Success.

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>

```

Figure 4. Call configure-memory command

Once the QSPI is successfully configured, the Quad SPI memory can then be access for further operations to erase, read, write, and so on.

Bootloader uses its get-property command to read the external memory attributes for the configured QSPI memory. The following image shows the response for QSPI memory available on the LPC540xx platform. The command requires two arguments, the property ID, 25, to read external memory attributes and memory ID, and 0xA, for external Quad SPI memory devices.

```

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- get-property 25 0xa
Inject command 'get-property'
Response status = 0 (0x0) Success.
Response word 1 = 15 (0xf)
Response word 2 = 268435456 (0x10000000)
Response word 3 = 16384 (0x4000)
Response word 4 = 256 (0x100)
Response word 5 = 4096 (0x1000)
Response word 6 = 0 (0x0)
UNKNOWN Attributes: Start Address = 0x10000000 Total Size = 16 MB Page Size = 256 bytes Sector Size = 4 KB

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>

```

Figure 5. Response for QSPI memory

### 3.2.2 Programming application to Quad SPI memory

As previously stated, once the Quad SPI is configured using the configure-memory command, the external memory becomes accessible. To program the demo application, the flash area must be first erased before programming the application image. The following examples show the external flash memory region being erased using the flash-erase-region command and write-memory command being used to program the application image on the erased region.

```

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -t 100000 -- flash-erase-region 0x10000000 0x100000
Inject command 'flash-erase-region'
Successful generic response to command 'flash-erase-region'
Response status = 0 (0x0) Success.

C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -t 100000 -- write-memory 0x10000000 led_blinky.bin
Inject command 'write-memory'
Preparing to send 1784 (0x6f8) bytes to the target.
Successful generic response to command 'write-memory'
(1/1)100% Completed!
Successful generic response to command 'write-memory'
Response status = 0 (0x0) Success.
Wrote 1784 of 1784 bytes.

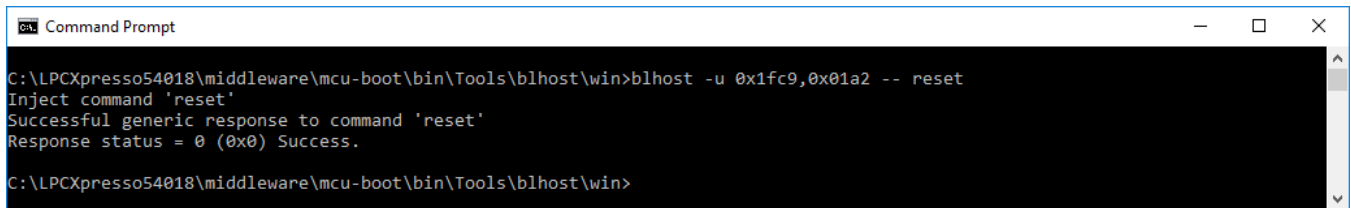
C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>

```

Figure 6. External flash memory erased

Device running flashloader

The application used in the above example is a simple LED blinking example (led\_blinky.bin) from the MCUXpresso SDK package. The application begins executing on the next boot or reset. The bootloader command `-- reset` can also be used to reset the device to boot and execute the led blinking application from the Quad SPI memory.



```
Command Prompt
C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- reset
Inject command 'reset'
Successful generic response to command 'reset'
Response status = 0 (0x0) Success.
C:\LPCXpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>
```

**Figure 7. Reset command**

**NOTE**

For a description of each bootloader command, see the *blhost User's Guide* (document MCUBLHOSTUG).



# Chapter 4

## Quad SPI configuration structure

The FlexSPI NOR Configuration Option Block is organized by 4-bit unit. It is expandable, and current definition of the block is as shown in the following table. The Flashloader detects FNORCB using the read SFDP command supported by most flash devices that are JESD216(A/B)- compliant. However, JESD216A/B only define the dummy cycles for Quad SDR read. In order to get the dummy cycles for DDR/DTR read mode, the flashloader supports auto probing by writing test patterns to offset 0x200 on the external memory devices. To get optimal timing, the readSampleClkSrc is set to 1 in Flashloader for Flash devices that do not support external provided DQS pad input. It is set to 3 in Flashloader for flash devices that do support external provided DQS pad input, such as HyperFlash. FlexSPI\_DQS pad is not used for any other purpose.

**Table 1. FlexSPI NOR Configuration Option Block**

Offset	Field	Description							
0	Option0								
		<b>TAG</b> [31:28]	<b>Option size</b> [27:24]	<b>Device detection type</b> [23:20]	<b>Query CMD Pad(s)</b> [19:16]	<b>CMD Pad(s)</b> [15:12]	<b>Quad Enable Type</b> [11:8]	<b>Misc</b> [11:4]	<b>Max Freq</b> [3:0]
		0x0C	Size in bytes = (Option Size + 1) * 4	0 - QuadSPI SDR 1 - QuadSPI DDR 2 - HyperFlash 3 - HyperFlash 3V 4 - MXIC OPI DDR 6 - Micron OPI DDR 8 - Adesto OPI DDR	0 - 1 2 - 4 3 - 8	0 - 1 2 - 4 3 - 8	1 - QE bit is bit 6 in StatusR eg1 2 - QE bit is bit 1 in StatusR eg2 3 - QE bit is in bit7 in StatusR eg2 4 - QE bit is bit 1 in StatusR eg2, enable command is 0x31	3 - Byte order is swapped under OPI DDR mode	Device-specific, see System Boot chapter in SoC RM for more details

**Table 1. FlexSPI NOR Configuration Option Block (continued)**

Offset	Field	Description			
4	Option1 Optional				
		<b>flash_connection[31:28]</b>	<b>drive_strength[27:24]</b>	<b>Reserved [23:8]</b>	<b>Dummy Cycle [7:0]</b>
		0 - PortA 1 - Parallel Mode	The drive strength of FlexSPI pad. See IOMUXC chapter in SoC RM for more details.	Reserved for future use	0 - Use auto-probing dummy cycle Others - dummy cycles provided in data sheet

- Tag - Fixed as 0x0C
- Option Size - Provide scalability for future use, the option block size equals to (Option size + 1) \* 4 bytes.
- Device Detection type - Software defined device types used for config block auto detection.
- Query Command Pad(s) - Command pads (1/4/8) for the SFDP command.
- CMD pad(s) - Commands pads for the Flash device (1/4/8). For devices that works under 1-1-4, 1-4-4, 1-1-8, or 1-8-8 mode, CMD pad(s) value is always 0x0. For devices that only support 4-4-4 mode for high performance, CMD pads value is 2. For devices that only support 8-8-8 mode for high performance, CMD pads value is 3.
- Quad Enable Type - Specify the Quad Enable sequence. Only applicable for devices that are only JESD216-compliant. This field is ignored if device supports JESD216A or later version.
- Misc - Specify miscellaneous mode for selected flash type.
- Max Frequency - The maximum work frequency for the specified flash device.
- Dummy Cycle - User provided dummy cycles for SDR/DDR read command.

#### Typical use cases for FlexSPI NOR Configuration Block

- QuadSPI NOR - Quad SDR Read: option0 = 0xc0000006 (100 MHz).
- QuadSPI NOR - Quad DDR Read: option0 = 0xc0100003 (60 MHz).
- HyperFlash 1V8: option0 = 0xc0233007 (133 MHz).
- HyperFlash 3V0: option0 = 0xc0333006 (100 MHz).
- HyperFlash 3V0: option0 = 0xc0333006 (100 MHz).
- MXIC OPI DDR: option0 = 0xc0403006 (100 MHz), for devices where data order is consistent between SDR mode and DDR mode, option0 = 0xc0403036, for device that the data order is inconsistent between SDR mode and DDR mode.
- Micron Octal DDR: option0=0xc0600006 (100 MHz).
- Micron OPI DDR: option0=0xc0603006 (100 MHz), SPI->OPI DDR.
- Micron OPI DDR (DDR read enabled by default): option0=0xc0633006(100 MHz).
- Adesto OPI DDR: option0=0xc0803007(133 MHz).

## Chapter 5

# One Time Programmable (OTP) memory

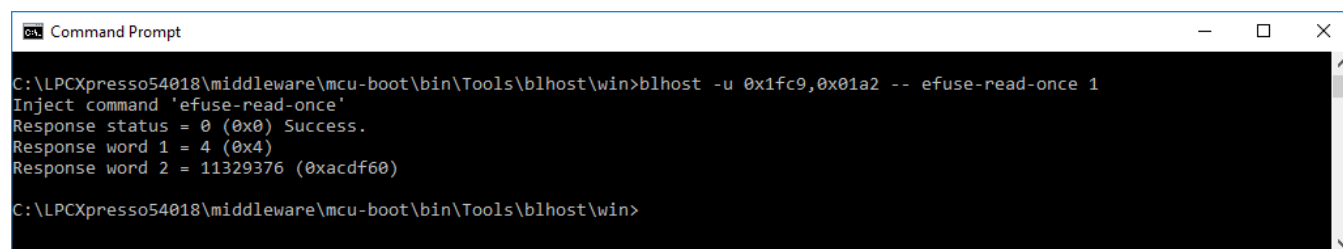
The flashloader also provides commands to program and read the on-chip one-time programmable (OTP) fuse memory from the LPC540xx device. The efuse-program-once command is used for programming the fuses, and efuse-read-once to read the programmed data from the fuse memory. The OTP memory for LPC540xx devices contains four memory banks of 128 bits each. Every 128-bit bank is comprised of four 32-bit words. From Bank 0 Register 0 to Bank 3 Register 3, there are sixteen 32-bit words. A single word can be read or programmed in each call to the bootloader fuse commands. See Chapter 45, LPC540xx OTP in [UM11060 LPC540xx/LPC54S0xx User's Manual](#) to see OTP functional details.

### 5.1 efuse-read-once

The efuse-read-once command requires one argument, an index to one of 0 to 15 32-bit fuse words.

index 0: OTP Bank 0 Word 0  
 index 1: OTP Bank 0 Word 1  
 index 2: OTP Bank 0 Word 2  
 index 3: OTP Bank 0 Word 3  
 index 4: OTP Bank 1 Word 0  
 index 5: OTP Bank 1 Word 1  
 index 6: OTP Bank 1 Word 2  
 index 7: OTP Bank 1 Word 3  
 index 8: OTP Bank 2 Word 0  
 index 9: OTP Bank 2 Word 1  
 index 10: OTP Bank 2 Word 2  
 index 11: OTP Bank 2 Word 3  
 index 12: OTP Bank 3 Word 0  
 index 13: OTP Bank 3 Word 1  
 index 14: OTP Bank 3 Word 2  
 index 15: OTP Bank 3 Word 3

The index can be obtained by calculating  $4 * \text{OTP\_Bank\_Index} + \text{Word\_Index}$ . The following figure shows the results from fuse-read-once command for OTP Bank 0 Word 1.



```

C:\LPCpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>blhost -u 0x1fc9,0x01a2 -- efuse-read-once 1
Inject command 'efuse-read-once'
Response status = 0 (0x0) Success.
Response word 1 = 4 (0x4)
Response word 2 = 11329376 (0xacdf60)

C:\LPCpresso54018\middleware\mcu-boot\bin\Tools\blhost\win>
  
```

Figure 8. efuse-read-once command results OTP Bank 0 Word 1

## 5.2 efuse-program-once

The efuse-program-once command requires two arguments. The first is an index to one of 0 to 15 fuse words, and the second argument is the 32-bit data to program to the fuse word specified in the first argument. For LPC540xx devices, the first memory bank (OTP Bank 0) is reserved. The other three OTP banks are programmable. So, index 0, 1, 2, and 3 are invalid. See Chapter 45, LPC540xx OTP in [UM11060 LPC540xx/LPC54S0xx User's Manual](#) to see OTP functional details. The following example shows the blhost calling to program bit 0 of OTP Bank 1 Word 3.

```
blhost.exe -u 0x1fc9,0x01a2 - efuse-program-once 7 00000001
```

```
nxp68779@NXL40813 /C/Users/nxp68779/Desktop/dfu-util-0.9-win64
$ blhost.exe -u 0x1fc9,0x01a2 -- efuse-read-once 7
Inject command 'efuse-read-once'
Response status = 0 (0x0) Success.
Response word 1 = 4 (0x4)
Response word 2 = 0 (0x0)

nxp68779@NXL40813 /C/Users/nxp68779/Desktop/dfu-util-0.9-win64
$ blhost.exe -u 0x1fc9,0x01a2 -- efuse-program-once 7 00000100
Inject command 'efuse-program-once'
Successful generic response to command 'efuse-program-once'
Response status = 0 (0x0) Success.

nxp68779@NXL40813 /C/Users/nxp68779/Desktop/dfu-util-0.9-win64
$ blhost.exe -u 0x1fc9,0x01a2 -- efuse-read-once 7
Inject command 'efuse-read-once'
Response status = 0 (0x0) Success.
Response word 1 = 4 (0x4)
Response word 2 = 256 (0x100)
```

**Figure 9. efuse-program-once command results**

### NOTE

The second argument should be in hex without the prefix "0x" to the 32-bit hex word.

# Chapter 6

## Revision history

This is the first revision of this document.

**How To Reach Us****Home Page:**[nxp.com](http://nxp.com)**Web Support:**[nxp.com/support](http://nxp.com/support)

Information in this document is provided solely to enable system and software implementers to use NXP 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. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP 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 NXP 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. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [nxp.com/SalesTermsandConditions](http://nxp.com/SalesTermsandConditions).

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, All other product or service names are the property of their respective owners. ARM, AMBA, ARM Powered, are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

© 2018 NXP B.V.

