AN11651

LPC18xx Secondary USB Host bootloader

Rev. 1 — 04 March 2015 Application note

Document information

Info	Content
Keywords	LPC18xx, USB mass storage Host, IAP
Abstract	This application note describes how to add a custom secondary USB mass storage Host bootloader to LPC18xx. The USB0 port on LPC18xx is configured as a Host to read user application from USB flash drive and then the user application is written to LPC18xx internal flash.

Revision history

Rev	Date	Description
1	20150304	Initial version.

Contact information

For additional information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Overview

The NXP LPC18xx is an ARM Cortex-M3 based microcontroller for embedded applications requiring a high level of integration and low power dissipation.

The LPC18xx can run at frequencies of up to 180 MHz and includes up to 200 kB of on-chip SRAM (flashless parts) or up to 136 kB of on-chip SRAM and up to 1 MB of flash program memory (parts with on-chip flash) with In-System Programming (ISP) and In-Application Programming (IAP) capabilities. It also includes one High-speed USB 2.0 Host/Device/OTG interface with on-chip high speed PHY and one High-speed USB 2.0 Host/Device interface with on-chip full-speed PHY and ULPI interface to an external high-speed PHY.

The LPC18xx provides the user a convenient way to update the flash contents in the field for bug fixes or product updates. This can be achieved using the following two methods:

- ISP: In-System Programming is programming or re-programming the on-chip flash memory, using the bootloader software and UART0 serial port. This can be done even when the part resides on the application board.
- IAP: In-Application Programming performs erase and write operations on on-chip flash memory, as directed by the user application code.

In this application example, a secondary bootloader is a piece of code that allows a user application code to be downloaded using alternative channels other than the standard UART0 used by the on-chip bootloader. Here the primary bootloader is the firmware that resides in an LPC18xx microcontroller's boot ROM block and is executed at power-on and after resets. After the boot ROM's execution, the secondary bootloader is executed, which then executes the user application.

This application note uses USB Mass Storage Host as an example for developing a secondary bootloader on LPC1857. The user application is read from the USB flash drive, through the USB port and then written in the LPC1857 internal flash. The booting process can be controlled and shown on the PC by the software terminal program using a COM port.

2. Environment

2.1 Hardware

- **Board**: MCB1800 Board with LPC1857
- **Debugger**: KEIL ULINK2
- **Others:** USB flash drive, COM cable
- **Board setup**: The board setup for this application example is shown in Fig 1

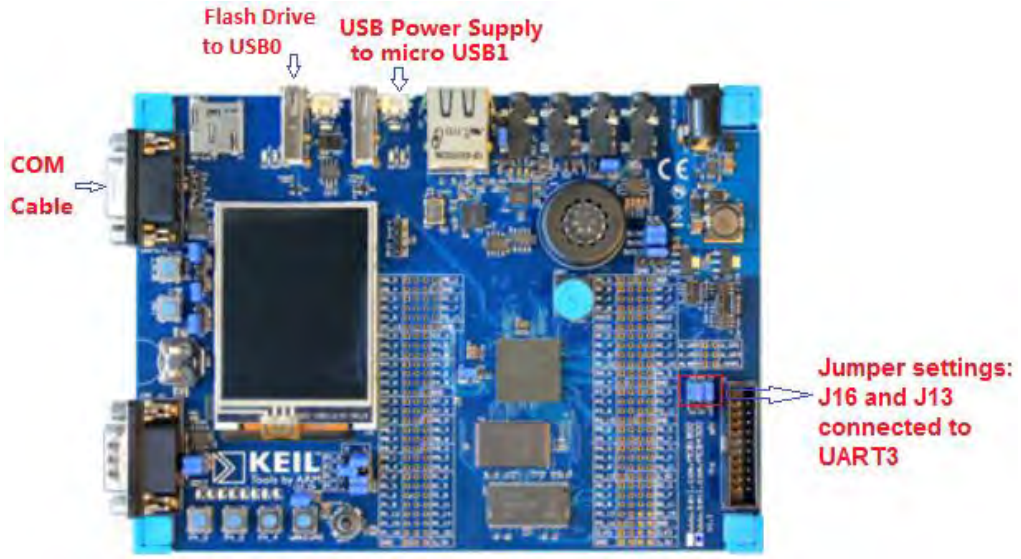

Fig 1. Board setup

2.2 Software

- Development IDE: KEIL uVision5 (V5.10.0.2)
- Tool: Software terminal program on the PC, such as PuTTy. The communication protocol is configured as 115200 bps, 8 data bits, 1 stop bit, no parity, and no flow control.

3. Flashing the LPC18xx

3.1 Flash sectors

The LPC18xx device family has up to 1 MB (Mega Byte) of on-chip user flash memory. For more detailed information about the specific device, please check the LPC18xx User Manual UM10430. Fig 2 shows how this user flash space is divided into a number of sectors.

To make any modification (even if it is just one byte) to a particular sector, the entire sector must be first erased and then re-written.

Flash bank	Sector number	Sector size [kB]	Start address	End address	LPC18x2	LPC18x3	LPC18x5	LPC18x7
A	0	8	0x1A00 0000	0x1A00 1FFF	yes	yes	yes	yes
A	1	8	0x1A00 2000	0x1A00 3FFF	yes	yes	yes	yes
A	2	8	0x1A00 4000	0x1A00 5FFF	yes	yes	yes	yes
A	3	8	0x1A00 6000	0x1A00 7FFF	yes	yes	yes	yes
A	4	8	0x1A00 8000	0x1A00 9FFF	yes	yes	yes	yes
A	5	8	0x1A00 A000	0x1A00 BFFF	yes	yes	yes	yes
A	6	8	0x1A00 C000	0x1A00 DFFF	yes	yes	yes	yes
A	7	8	0x1A00 E000	0x1A00 FFFF	yes	yes	yes	yes
A	8	64	0x1A01 0000	0x1A01 FFFF	yes	yes	yes	yes
A	9	64	0x1A02 0000	0x1A02 FFFF	yes	yes	yes	yes
A	10	64	0x1A03 0000	0x1A03 FFFF	yes	yes	yes	yes
A	11	64	0x1A04 0000	0x1A04 FFFF	yes	-	yes	yes
A	12	64	0x1A05 0000	0x1A05 FFFF	yes	-	yes	yes
A	13	64	0x1A06 0000	0x1A06 FFFF	yes	-	-	yes
A	14	64	0x1A07 0000	0x1A07 FFFF	yes	-	-	yes

Fig 2. Flash sectors

Both the secondary USB Host bootloader and the user application reside in the flash. Therefore, for the secondary USB Host bootloader to flash the user application without modifying any of its own code, the user application should be flashed starting at the next available sector.

3.2 IAP

IAP is a feature that allows a user application to erase and write to internal flash memory. The IAP commands need to be utilized for the secondary bootloader to flash the user application into the on-chip flash.

A detailed description of the IAP commands can be found in the LPC18xx user manual. IAP APIs calls reside in ‘iap_18xx_43xx.c’ file in the “bootloader” project.

4. Driver and File system

USB and UART are used in this secondary bootloader. The secondary USB bootloader code is based on LPCOpen platform V2.12 and therefore the software architecture is similar to that of LPCOpen V2.12. The libraries are built from chip drivers and board drivers that are used by "Bootloader" and "periph_blinky" projects. The source files for the library projects are included in the ‘software’ folder of the package.

There are many USB Device Classes, for example, DFU (Device Field Upgrade), HID (Human Interface Device), and MSCH (Mass Storage Class Host). The MSCH class allows the embedded system to act as a Host to access USB mass storage device such as USB flash drive with file system which can store a file from PC.

To make the LPC18xx access USB flash drive, a FAT (File Allocation Table) file system is needed. To understand how the FAT file system works, one can search the Internet for details on File Allocation Table and Storage Class Devices.

The example code in this application note implements a FAT32 file system.

A simple console through the COM port based on UART protocol is implemented to control the booting process with a few simple commands.

5. Secondary bootloader

5.1 Code placement in flash

The secondary bootloader is placed at the starting address 0x1A000000 so that it will be executed by the LPC18xx after reset. The USB communication within the bootloader utilizes interrupts and therefore contains a vector table. See Fig 3.

Flash programming is based on a sector-by-sector basis. This means that the code for the user application should not be stored in any of the flash sectors that contain the secondary bootloader.

For efficient use of flash space, the user application should be flashed into the next available empty sector after the bootloader. Note that in this application example the user application also contains its own interrupt vector table as shown in Fig 3.

Before executing user application, the secondary bootloader modifies the VTOR register so that NVIC can use the NVIC table of user application instead of that of the secondary bootloader. The stack pointer (SP) and program counter (PC) registers are also updated for the execution of the user application. The SP points to the new location where the user application has allocated the top of its stack (the stack grows downwards in memory). The PC on the other hand contains the location of the first executable instruction in the user application. From here on, the CPU continues normal execution and initializations specified in the user application. See Fig 3.

Note: The secondary bootloader is designed to use a simple console mode to boot the user application. The user can create other booting modes according to application requirements.

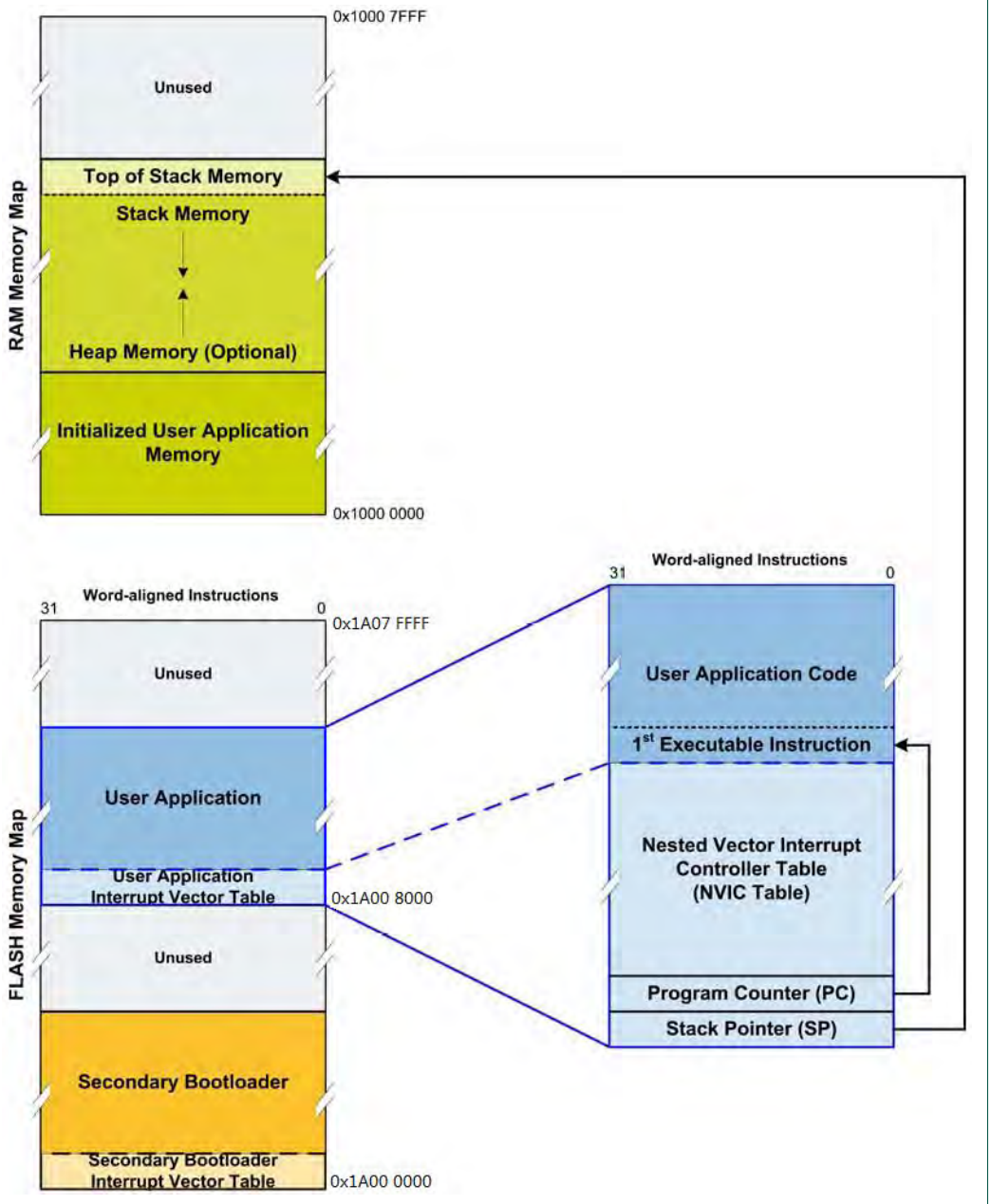

Fig 3. Bootloader and user application in flash

5.2 Using the secondary bootloader

5.2.1 Installing the bootloader

After setting up the hardware and software environment, open a software terminal program (here PuTTy is used) and use the following steps to install the secondary bootloader:

1. Open the secondary USB bootloader sample project “bootloader.prj” in the path of \applications\lpc18xx_43xx\keil_uvision_projects\bootloader\. Or open the multi-project “mcb1857_bootloader_example.uvmpw” that supports the showing and managing of all projects and source codes in one opened IDE window. It is located at the location \applications\lpc18xx_43xx\keil_uvision_projects\ . Using the multi-project method is recommended.
2. If necessary, make any desired code changes.
3. Build the project.
4. Erase the LPC18xx.
5. Flash the LPC18xx with the bootloader through a debugger such as ULink2.

5.2.2 Running the bootloader

After installing the secondary bootloader, reset the MCB1800 board. The secondary bootloader starts and does not find the user application since the flash has been erased or is blank. The report and command list that can be used for booting is shown on terminal program console of the computer screen, see Fig 4.

```
        <Secondary USB Massstorage Host Bootloader>
                ---Version 1.00---

No user application! Please update with U-disk

-------- Command List -------

 ls - list root files in U-disk
 ld - load user application to flash like 'ld xxx.bin'
 go - run user application
```

Fig 4. Start secondary bootloader

Next, copy and paste the binary file of the user application to the root directory of the USB flash drive inserted in one of the USB ports on the computer. The application binary file is named ‘blinky.bin’ and it is created in the folder:

\applications\ lpc18xx_43xx\keil_uvision_projects\periph_blinky\keil_output\

Section 6 shows the steps to create a user application binary file ‘blinky.bin’ from the "periph_blinky" project.

The USB flash drive with the binary file is inserted in USB port 0 on the MCB1800 board. The USB flash drive is detected and the information is shown on the terminal program console, see Fig 5:

```
Device Attached on port 0

>
```

Fig 5. USB device attached

To list the content of the flash drive, at the command prompt in the terminal window type “ls” and press “Enter” key.

The root files are listed, see Fig 6:

```
> ls
BLINKY~1.BIN
MESSAGE.TXT
BLINKY.BIN
OS.BIN
```

Fig 6. List root files in USB flash drive

Now, type the command “ld blinky.bin” and press “Enter” key to load the binary file of the user application from the USB flash drive and program it into the internal flash of the LPC1857 device. Fig 7 shows this instruction and the output in the computer terminal program window.

```
> ld blinky.bin

File blinky.bin Opened
File size: 5768
Loading to sectors: 4 - 4
Done!
```

Fig 7. Load the binary file of the user application

The last step is to type the command “go” and press “Enter” key to jump to the user application. When this user application executes, the LED labeled “PD_10” on the board blinks.

When the user application is already in the flash and power is supplied or reset is applied to the MCB1800 board, the secondary bootloader will delay several seconds to wait for the user to press “Enter” key to assume booting control. See Fig 8.

```
        <Secondary USB Massstorage Host Bootloader>
              ---Version 1.00---
Press [Enter] to input command
Delay... 1 2
```

Fig 8. Wait for controlling boot process

If “Enter” key is not pressed before boot time-out, the secondary bootloader automatically jumps and executes the user application. Pressing “Enter” key, before boot time-out, shows “command list” and command prompt to control booting.

To use USB port 1 on the board to update the user program, set “.PortNumber” to “1” in the structure “FlashDisk_MS_Interface” in “MassStorageHost.c” file. See Fig 9.

```
USB_ClassInfo_MS_Host_t FlashDisk_MS_Interface = {
  .Config = {
    .DataINPipeNumber       = 1,
    .DataINPipeDoubleBank   = false,

    .DataOUTPipeNumber      = 2,
    .DataOUTPipeDoubleBank  = false,
    .PortNumber = 1,
  },
};
```

Fig 9. Setting USB1 port as Host Port

When using USB port 1 to connect USB flash drive, the power should be supplied by micro USB port 0.

6. User application

This section briefly describes how to create the binary file from user application which can be used by the secondary bootloader.

- Open the project “periph_blinky.prj” in: \applications\lpc18xx_43xx\keil_uvision_projects\periph_blinky\. Or open the multi-project “mcb1857_bootloader_example.uvmpw” that supports the showing and managing of all projects and source codes in one opened IDE window. It is located in: \applications\lpc18xx_43xx\keil_uvision_projects\.
- Open the “Target Options”.
- Change the starting address to “0x1A008000” on “Target” Tab as follows. See Fig 10.

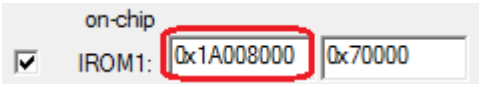

Fig 10. Change starting address

- Change the base address for creating the flash signature on the “User” tab. The base address to create the flash signature for this project should be 0x1A008000 because the "periph_blinky" project starts from address 0x1A008000. Configurations for generating binary file must be set on the “User” tab. Check the “Run #1:” and “Run #2:” boxes under “Run User Programs After/Rebuild” and type the commands shown in Fig 11.

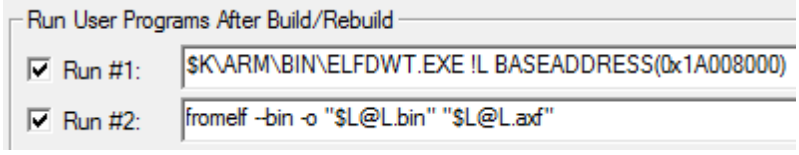

Fig 11. Configure to generate binary file

Note: CRP can be implemented in the secondary bootloader, it is not included in this application note.

- Build the project. The binary file “blinky.bin” for updating can be found in the path of \applications\lpc18xx_43xx\keil_uvision_projects\periph_blinky\keil_output\ after building successfully.

7. Conclusion

By using a secondary bootloader it is possible to conveniently perform In-application software updates without using additional development hardware, such as JTAG/SWD. In this case, the secondary bootloader uses USB to transfer the binaries to the LPC18xx, but other channels such as Ethernet and SPI can also be utilized.

This application note serves as a reference on how to use, create, and modify a secondary USB mass storage Host bootloader. The secondary USB mass storage Host bootloader has been designed as a standalone project that contains all of its dependent source files.

The user applications used in this demonstration are modified code examples that are included with Keil's uVision toolchain.

8. References

[1] NXP LPC18xx User Manual UM10430, NXP Semiconductors

9. Legal information

9.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

9.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

9.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

10. Contents

Document information **1**
1. Overview **3**
2. Environment **3**
2.1 Hardware 3
2.2 Software 4
3. Flashing the LPC18xx **4**
3.1 Flash sectors 4
3.2 IAP 5
4. Driver and File system **5**
5. Secondary bootloader **6**
5.1 Code placement in flash 6
5.2 Using the secondary bootloader 7
5.2.1 Installing the bootloader 7
5.2.2 Running the bootloader 8
6. User application **10**
7. Conclusion **11**
8. References **12**
9. Legal information **13**
9.1 Definitions 13
9.2 Disclaimers 13
9.3 Trademarks 13
10. Contents **14**

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 04 March 2015
Document identifier: AN11651