AN11628

LPC177x_8x Secondary USB Host Bootloader

Rev. 1.1 — 05 March 2015 Application note

Document information

Info	Content
Keywords	LPC177x_8x, USB mass storage Host, IAP
Abstract	This application note describes how to add a custom secondary USB mass storage Host bootloader to LPC177x_8x. The USB on LPC177x_8x is configured as a Host to read user application from the USB flash drive and then the user application is written to the internal flash on LPC177x_8x.

Revision history

Rev	Date	Description
1	20150219	Initial version.
1.1	20150305	Updated cross reference to Fig 1 Changed the text U-disk to USB flash drive

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

NXP's LPC178x/177x is an ARM Cortex-M3 based microcontroller for embedded applications requiring a high level of integration and low power dissipation.
The LPC178x/177x can run at frequencies of up to 120MHz and has up to 512KB on-chip flash program memory with In-System Programming (ISP) and In-Application Programming (IAP) capabilities. It also includes USB 2.0 full-speed controller that can be configured for either device or Host, or OTG operation with an on-chip PHY for device and Host functions.

The LPC178x/177x provides the user a convenient way to update the flash content 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 boot loader software and UART0 serial port. This can be done when the part resides on the end-user board.

• IAP: In-Application programming is performing erase and write operations on the on-chip flash memory, as directed by the end-user application code.

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 primary bootloader (on-chip). The primary bootloader is the firmware that resides in a microcontroller's boot ROM block and is executed on power-up and resets. After the boot ROM's execution, the secondary bootloader is executed, which then executes the end-user application.

This application note uses USB mass storage Host as an example for developing a secondary bootloader on LPC1788. The user application is read from the USB flash drive through the USB port and then written in the LPC1788 internal flash. The booting process can be controlled and shown by the COM tool on the PC.

2. Environment

2.1 Hardware

- **Board**: EA LPC1788 OEM Board Rev B (core board) and EA OEM Base Board PB1
- **Debugger**: KEIL ULINK2
- **Other:** USB flash, COM cable
- **Board setup**: the board setup is shown in See Fig 1.

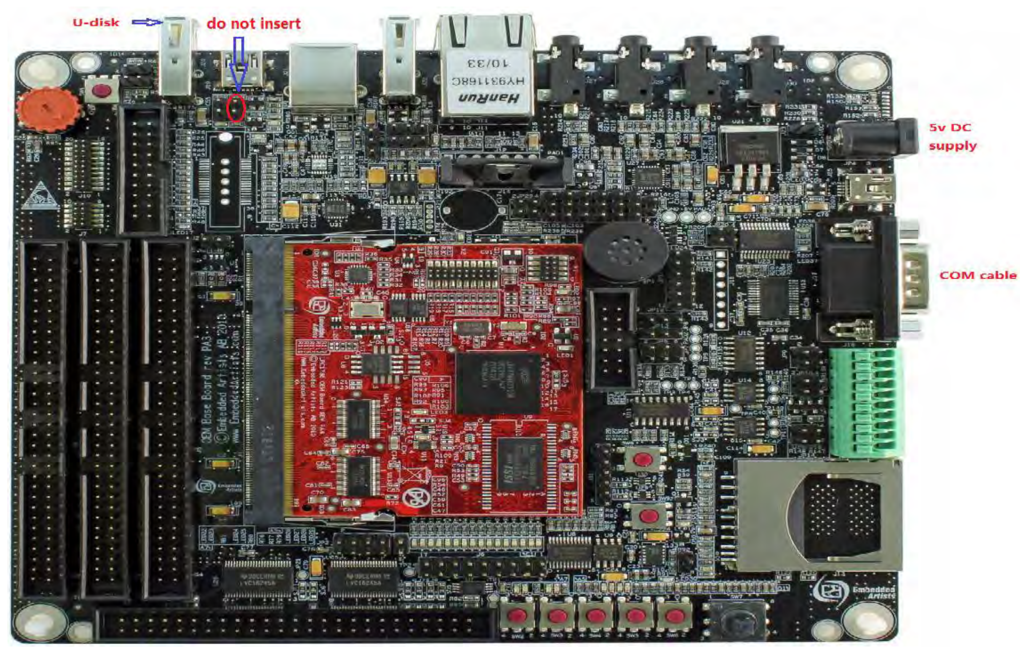

Fig 1. Board setup

2.2 Software

- **Development IDE**: KEIL uVision4 (V4.70.0)
- **Tool**: COM tool on the PC, such as PuTTy. The communication protocol is configured as 115200bps, 8 data bits, 1 stop bit, no parity, no flow control.

3. Flashing the LPC177x/8x

3.1 Flash sectors

Depending on the LPC1700 part variant, the user has up to 512kB (Bytes) of on-chip flash available. This user flash space is divided up into sectors, see Fig 2.

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

Sector Number	Sector Size [kB]	Start Address	End Address	32 kB Part	64 kB Part	128 kB Part	256 kB Part	512 kB Part
0	4	0X0000 0000	0X0000 0FFF	x	x	x	x	x
1	4	0X0000 1000	0X0000 1FFF	x	x	x	x	x
2	4	0X0000 2000	0X0000 2FFF	x	x	x	x	x
3	4	0X0000 3000	0X0000 3FFF	x	x	x	x	x
4	4	0X0000 4000	0X0000 4FFF	x	x	x	x	x
5	4	0X0000 5000	0X0000 5FFF	x	x	x	x	x
6	4	0X0000 6000	0X0000 6FFF	x	x	x	x	x
7	4	0X0000 7000	0X0000 7FFF	x	x	x	x	x
8	4	0x0000 8000	0X0000 8FFF		x	x	x	x
9	4	0x0000 9000	0X0000 9FFF		x	x	x	x
10 (0x0A)	4	0x0000 A000	0X0000 AFFF		x	x	x	x
11 (0x0B)	4	0x0000 B000	0X0000 BFFF		x	x	x	x
12 (0x0C)	4	0x0000 C000	0X0000 CFFF		x	x	x	x
13 (0x0D)	4	0x0000 D000	0X0000 DFFF		x	x	x	x
14 (0X0E)	4	0x0000 E000	0X0000 EFFF		x	x	x	x
15 (0x0F)	4	0x0000 F000	0X0000 FFFF		x	x	x	x
16 (0x10)	32	0x0001 0000	0x0001 7FFF			x	x	x
17 (0x11)	32	0x0001 8000	0x0001 FFFF			x	x	x
18 (0x12)	32	0x0002 0000	0x0002 7FFF				x	x
19 (0x13)	32	0x0002 8000	0x0002 FFFF				x	x
20 (0x14)	32	0x0003 0000	0x0003 7FFF				x	x
21 (0x15)	32	0x0003 8000	0x0003 FFFF				x	x
22 (0x16)	32	0x0004 0000	0x0004 7FFF					x
23 (0x17)	32	0x0004 8000	0x0004 FFFF					x
24 (0x18)	32	0x0005 0000	0x0005 7FFF					x
25 (0x19)	32	0x0005 8000	0x0005 FFFF					x
26 (0x1A)	32	0x0006 0000	0x0006 7FFF					x
27 (0x1B)	32	0x0006 8000	0x0006 FFFF					x
28 (0x1C)	32	0x0007 0000	0x0007 7FFF					x
29 (0x1D)	32	0x0007 8000	0x0007 FFFF					x

Fig 2. Flash sectors

Both the secondary USB Host bootloader and the user application reside in 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 with the next available sector.

3.2 IAP

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

A detailed description of the IAP commands can be found in the LPC177x_8x user manual. IAP APIs can be found in the lpc17xx_iap.c file.

4. Driver and File system

USB and UART are used in this secondary bootloader. In this application note, implementation details of the drivers are not explained. This secondary USB Bootloader code is modified based on LPCOpen platform V2.10. Therefore the software architecture is similar to LPCOpen V2.10. USB and board driver can be built to a library respectively. The source files for the 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 function as a Host to access USB mass storage device such as USB flash drive with a file system which can store a file from PC.
To make the LPC1700 access the USB flash drive, a FAT (File Allocation Table) file system is needed. Details on File Allocation Table and Storage Class Devices are available on the Internet.

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

A simple console through COM port based on UART protocol is implemented to control 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 0x0 so that it will be executed by the LPC177x/8x after reset. The USB communication within the bootloader utilizes interrupts and therefore contains a vector table.

Flash programming is executed on a sector-by-sector basis. This implies that the code for the user application should not be stored in any of the same flash sectors as 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 the user application also contains its own interrupt vector table.

Before executing user application, the secondary bootloader modifies the VTOR register so that NVIC can use the NVIC table of the 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 of initializations specified on the user application.

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 his requirement.

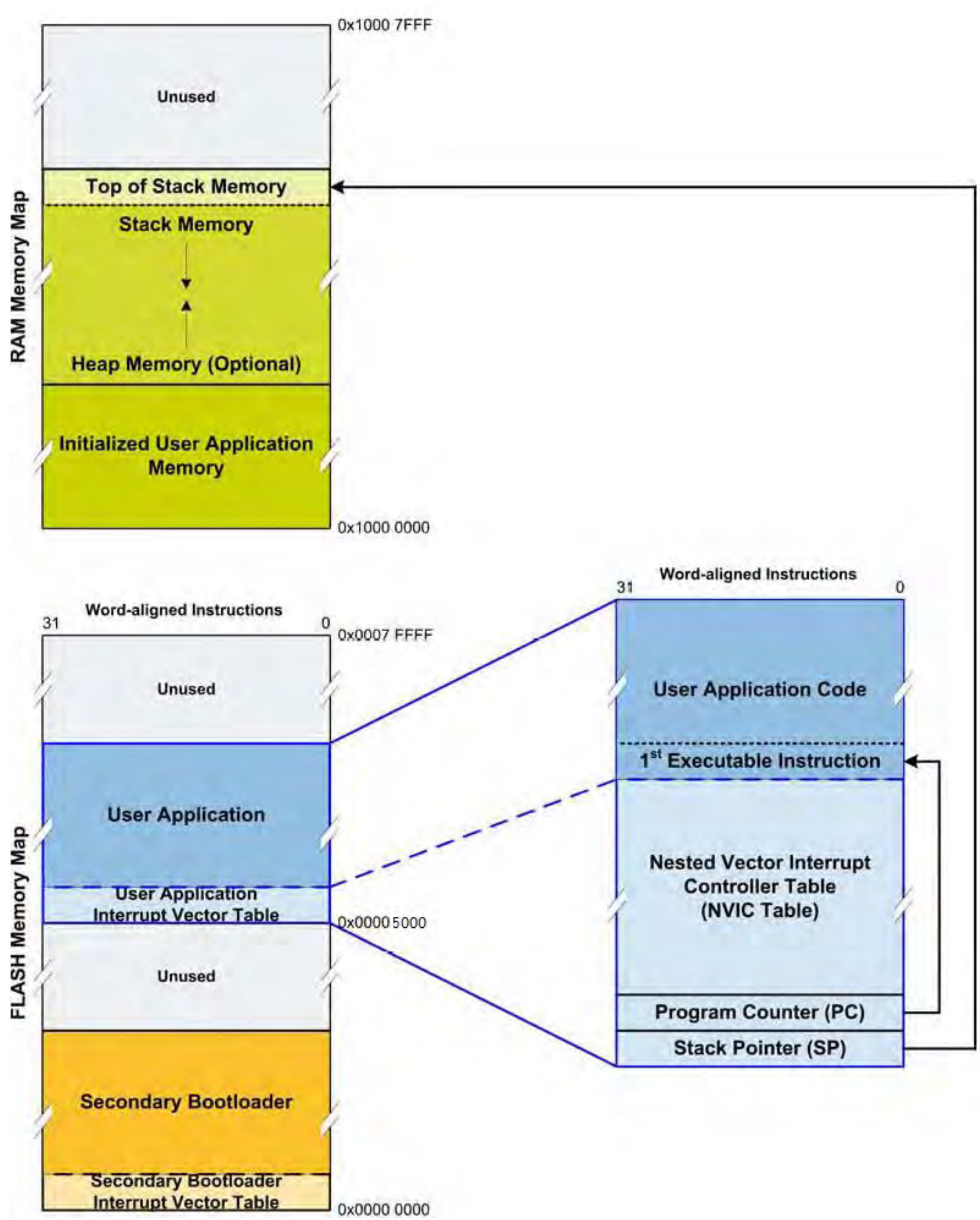

Fig 3. Bootloader and user application in flash

5.2 Using the secondary bootloader

5.2.1 Installing the bootloader

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

- Open the secondary USB bootloader sample project “bootloader.prj” in the path of \applications\ lpc17xx_40xx\keil_uvision_projects\bootloader.
- If necessary, make any desired code changes.

- Build the project.
- Erase the LPC177x_8x.
- Flash the LPC177x_8x with the bootloader through debugger such as ULink2.

5.2.2 Running the bootloader

After installing the secondary bootloader, reset the EA LPC1788 board. The secondary bootloader starts and finds no user application since the flash is erased. The report and command list that can be used for booting is shown on COM console on the PC. 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

Copy and paste the binary file of user application which is named ‘blinky.bin’ in the folder of \applications\lpc177x_8x\examples\periph_blinky\keil_output\iflash_ea_devkit_1788 (See Section 6 about how to create user application binary file) to the root directory of the USB flash drive inserted in a USB port of the PC.

Insert the USB flash drive with the binary file to USB port 1 (or 2) on the EA LPC1788 board. The USB flash drive device will be detected and the information will be shown on the COM console. See Fig 5:

```
Device Attached on port 1

>
```

Fig 5. USB device attached

Type command “ls” and press “Enter”, 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

Type the command “ld blinky.bin” and press “Enter” to load the binary file of the user application from the USB flash drive and program it to the internal flash of the LPC1788 board. See Fig 7.

```
> ld blinky.bin

File blinky.bin Opened
File size: 4148
Loading to sectors: 5 - 6
Done!
```

Fig 7. Load the binary file of the user application

Type the command “go” and press “Enter” to jump to the user application which is then executed.

When the user application exists in flash and power is supplied to the LPC1788 board, the secondary bootloader waits several seconds for the user to press “Enter” for booting control, as shown in 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” is not pressed within this delay period, the secondary bootloader automatically jumps to execute the user application. Otherwise, pressing “Enter” shows “command list” and waits for the command input for booting control.

6. User application

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

- Open the project “periph_blinky.prj” in the path of \applications\ lpc17xx_40xx\keil_uvision_projects\periph_blinky.
- Open the “Target Options”
- Change the starting address to “0x5000” on “Target” Tab as follows. See Fig 9.

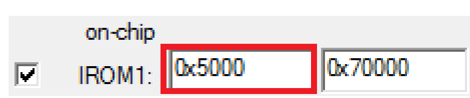

Fig 9. Change starting address

To generate binary file in Keil MDK IDE, configure it on the “User” tab. Check the “Run #1:” from the “Run User Programs After/Rebuild” and type the command shown in Fig 10.

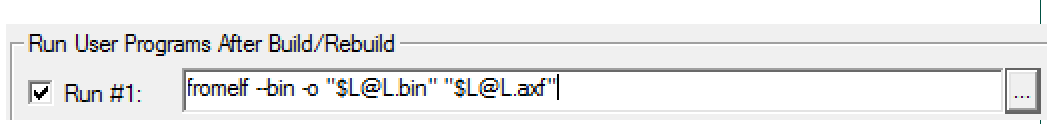

Fig 10. Configure to generate binary file

Disable the CRP allocation in the 'keil_startup_lpc17xx_40xx.s" on the "Asm" Tab, see Fig 11.

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

Fig 11. Configure to NO_CRP

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 LPC1700, but other channels such as Ethernet and SPI are also possible.

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 for this demonstration are modified code examples that are included with Keil's uVision toolchain.

8. References

[1] NXP LPC178x/7x User Manual UM10407, NXP Semiconductors

[2] AN10866_3 LPC1700 secondary USB bootloader

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

Date of release: 05 March 2015
Document identifier: AN11628