AN11335

Migrating from the LPC177x/8x to LPC407x/8x

Rev. 1 — 1 October 2013 Application note

Document information

Info	Content
Keywords	LPC177x_8x LPC407x_8x Migration Cortex-M3 Cortex-M4 SPIFI FPU
Abstract	This application note provides key pointers to migrate development from the Cortex-M3 family LPC177x_8x to the Cortex-M4 family LPC407x_8x

Revision history

Rev	Date	Description
1	20131001	Initial version.

Contact information

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

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

1. Introduction

NXP's LPC407x_8x microcontroller family is a high performance Cortex-M4 series based on the successful LPC177x_8x Cortex-M3 family. Both families are targeted to operate at up to 120 MHz CPU frequency under worst case commercial conditions. LPC407x_8x is pin compatible with LPC177x_8x with enhancements in the core and advanced peripherals. This application note highlights some of the differences between these two families and provides guidance when switching from the LPC177x_8x based hardware/software to the LPC407x_8x systems.

Fig 1 shows the block diagram of the LPC407x_8x family with the red area being an enhancement or additional functionality compared with the LPC177x_8x family.

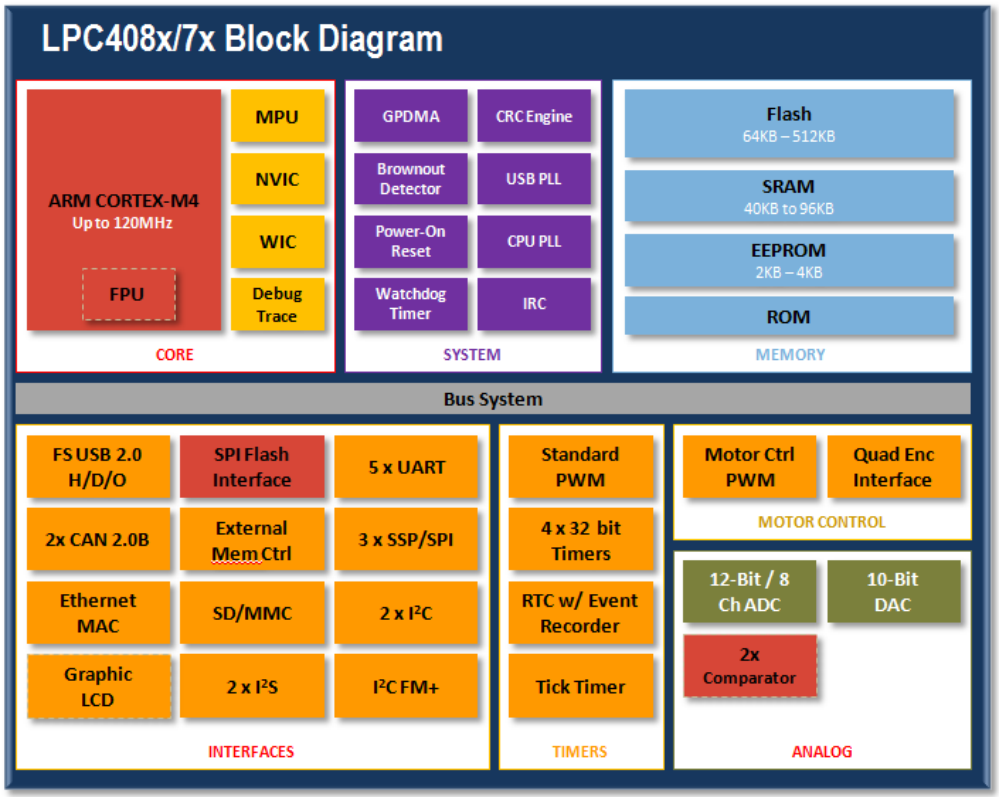

Fig 1. LPC407x_8x block diagram

As shown in Fig 1, the majority of the peripheral functionalities are common among these two MCU families which is a great help when migrating from LPC177x_8x to LPC407x_8x. In addition, the register names and functions are the same for all the common peripherals, making migration from LPC177x to LPC407x/8x a straightforward transition.

2. Architectural evolvement

2.1 Common architecture

The LPC177x_8x MCU family implements Cortex-M3 architecture while the LPC407x_8x MCU family implements Cortex-M4 architecture. From the core's perspective, Cortex-M4 is a Cortex-M3 plus DSP instructions and an optional Floating-Point Unit (FPU). There is much in common among NXP's LPC177x_8x and LPC407x_8x: same internal architecture, same fixed memory map, same privilege modes and stacks. Both the LPC177x_8x and LPC407x_8x have incorporated the Memory Protection Unit (MPU). They also share the same interrupt handing scheme and the Nested Vectored Interrupt Controller (NVIC). The same power management scheme is implemented on LPC177x_8x and LPC407x_8x. From the hardware and software designer's point of view, the models of both families are very similar.

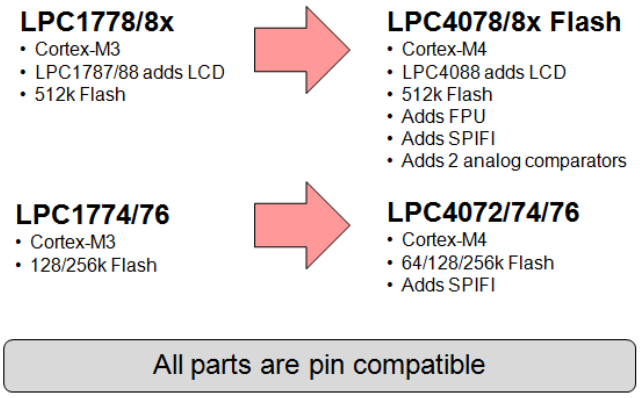

Fig 2. LPC177x_8x to LPC407x_8x comparison

2.2 Core architectural differences

- **Additional Instructions in LPC407x_8x**
  With the Cortex-M4 core implementation, the LPC407x_8x adds DSP and FPU enhanced instructions. The M4 core features a single-cycle Multiply-Accumulate (MAC) unit, optimized Single Instruction Multiple Data (SIMD) instructions and saturating arithmetic instructions. For integer performance, the Cortex-M3 is comparable to Cortex-M4. However, the Cortex-M4 provides higher performance when performing digital signal processing algorithms.
- **Floating Point Unit Coprocessor**
  The Cortex-M4 Floating Point Unit (FPU) is an optional coprocessor that is implemented in many members of the LPC407x_8x MCU family. The FPU supports single-precision floating-point computation functionality in compliance with the ANSI/IEEE Standard 754-2008. The FPU provides add, subtract, multiply, divide, accumulate and square root operations. It also performs a variety of conversions between fixed-point, floating-point and integer data formats.

3. Peripheral enhancements on LPC407x_8x

The LPC177x_8x is a powerful Cortex-M3 family of microcontrollers with many popular peripherals like the External Memory Controller (EMC), USB host/device/OTG and full LCD graphics controller. In Fig 1 we can see that the SPI Flash Interface (SPIFI) and the Dual Analog Comparator are two additional peripherals implemented in the LPC407x_8x.

3.1 SPIFI (SPI flash interfaces)

The SPIFI interface is an NXP unique peripheral that maps low cost serial flash memories into the internal memory system. SPIFI uses an SPI bus superset with 4 data lines to access off-chip quad SPI flash memory at a much higher rate than is possible using standard SPI or SSP interfaces. The SPIFI function allows memory mapping the contents of the off-chip SPI flash memory such that it can be executed as if it were on-chip code memory.

Unlike the LPC1800 and LPC4300 families which allow booting from the SPIFI interface, the LPC407x_8x cannot boot from an external SPIFI device.

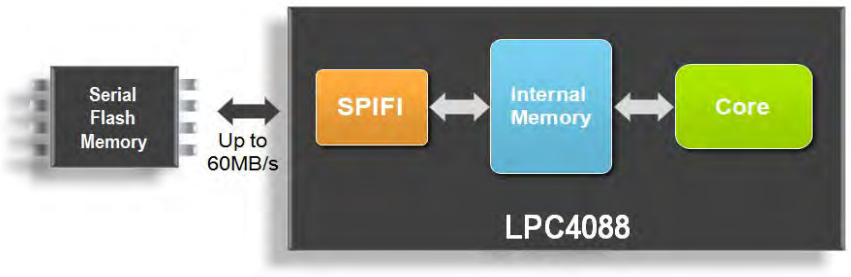

Fig 3. SPIFI overview

3.1.1 Some highlights on the SPIFI performance

- SPIFI is easy to use
- SPIFI saves board space and cost
- SPIFI execute-in-place at about 70 % the speed of internal flash (max speed is 80 MHz)
- SPIFI is faster than external 16-bit NOR flash
- Unlike NOR flash, SPIFI uses six signal lines instead of 40 or more lines that are required when interfacing with a 32-bit memory device with a parallel interface. All of these features make SPIFI a possible NAND and NOR flash replacement.

3.2 Dual analog comparators

3.2.1 Overview

The dual analog comparator interface has up to 5 selectable external sources per comparator, a 32-stage voltage ladder internal reference, relaxation oscillator circuitry output and two timers for edge counting or time measuring. Notice that some LPC407x_8x parts do not have the dual analog comparators. Please refer to the ordering information in the data sheet for details.

3.2.2 Block diagram

Below is the block diagram of the dual comparator module.

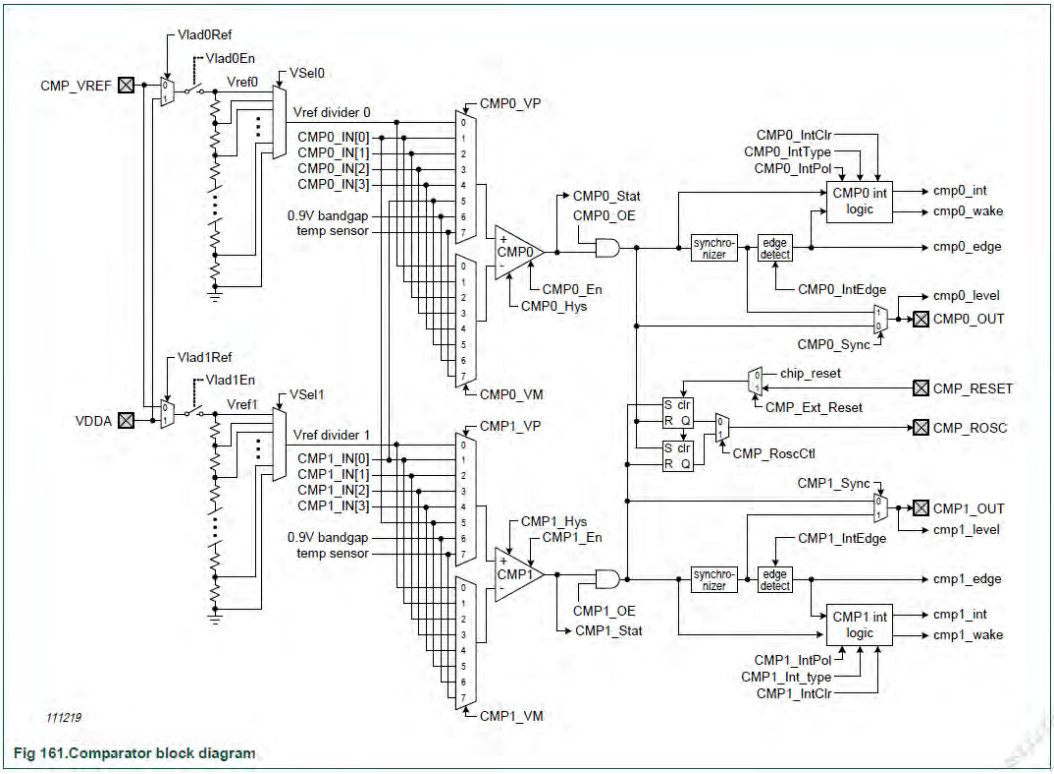

Fig 4. Dual comparator block diagram LPC4088

Some possible applications with the dual comparators could be Zero Crossing Detector or sigma delta ADC.

3.3 Pin compatibility

LPC407x_8x keeps all available functionality and pin assignment of the LPC177x_8x part with additional configurations assigned with the IOCON selection registers due to the addition of the SPIFI and dual Comparator interface.

The following is a summary of added pin mux functionality on the LPC407x_8x part. Note that the LPC1773 SPIFI interface uses the same pin as the LPC407x_8x.

Related Pins	added IOCON selection/functionality
P0[4]	5: CMP_ROSC (Comparator relaxation oscillator output)
P0[5]	5: CMP_RESET (Comparator reset input)
P0[6]	5: CMP_ROSC (Comparator relaxation oscillator output)
P0[7]	5: CMP_VREF (Comparator reference voltage)
P0[8]	5: CMP1_IN[3] (Comparator 1, input 3)
P0[9]	5: CMP1_IN[2] (Comparator 1, input 2)
P1[5]	5: CMP1_IN[1] (Comparator 1, input 1)
P1[6]	5: CMP0_IN[3] (Comparator 0, input 3)
P1[7]	5: CMP1_IN[0] (Comparator 1, input 0)
P1[12]	5: CMP1_OUT (Comparator 1, output)
P1[14]	5: CMP0_IN[0] (Comparator 0, input 0)
P1[16]	5: CMP0_IN[1] (Comparator 0, input 1)
P1[17]	5: CMP0_IN[2] (Comparator 0, input 2)
P4[30]	5: CMP0_OUT (Comparator 0, output)
P0[15]	5: SPIFI_IO[2] (Data bit 2 of SPIFI)
P0[16]	5: SPIFI_IO[3] (Data bit 3 of SPIFI)
P0[17]	5: SPIFI_IO[1] (Data bit 1 of SPIFI)
P0[18]	5: SPIFI_IO[0] (Data bit 0 of SPIFI)
P0[22]	5: SPIFI_CLK (Clock output for SPIFI)
P2[7]	5: SPIFI_CS (Chip select output for SPIFI)

Fig 5. Additional functionality pin assignment on LPC407x_8x

4. Customer sample code usage guide

Both the current Peripheral Driver Library (PDL) and LPCOpen Platform provide sample code for two device families.

Sample projects for LPC407x_8x are available with the LPCOpen platform at:

http://www.lpcware.com/content/nxpfile/lpcopen-platform

Sample projects are also available with the Peripheral Driver Library located at:

http://www.lpcware.com/content/nxpfile/lpc177x-and-lpc178x-cmsis-compliant-standard-peripheral-firmware-driver-library-keil

Sample projects with three IDEs (Keil, IAR and LPCXpresso) are provided in a very systematic manner in the LPCOpen platform. Below are some general guidelines.

4.1 Working with the LPCOpen sample projects

Discussions of the LPCOpen platform in this section are based on the LPCOpen version 1.03.

Under the LPCOpen platform root folder, there are two subfolders: software and applications. The folder “software” contains the core definition and all the low level device driver and development board configurations. The “applications” folder contains the sample projects based on the three IDEs mentioned above. Under subfolder “lpc17xx_40xx” within folder “applications”, we can see the following subfolders. Fig 6 provides brief description of the contents of each folder.

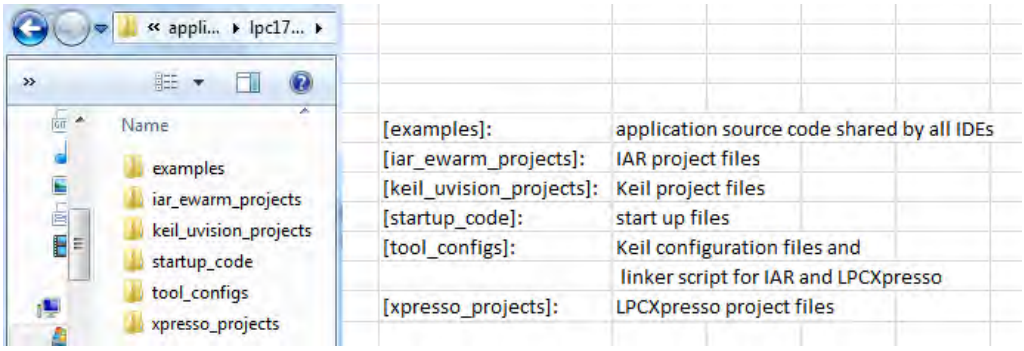

Fig 6. LPCOpen lpc17xx_40xx folder

Below are some common features with these three IDEs when migrating from LPC177x_8x to LPC407x_8x. For more detailed usage guide to the LPCOpen example projects for LPC177x_8x and LPC407x_8x, please refer to the following link:

http://docs.lpcware.com/lpcopen/v1.03/lpc_open_17xx_40xx_build_procs_keil.html

4.1.1 Access the sample projects

4.1.1.1 Keil and IAR

Keil and IAR sample projects are provided in a unified way and we will introduce the method to switching from a LPC177x_8x project to LPC407x_8x project using a Keil example.

With the LPCOpen platform, click into the subfolder \keil_uvision_projects and activate the Keil uvision IDE by double clicking lpcopen_periph_17xx_40xx.unmpw and choose to open only the lib and blinky projects, we are presented with the view in Fig 7.

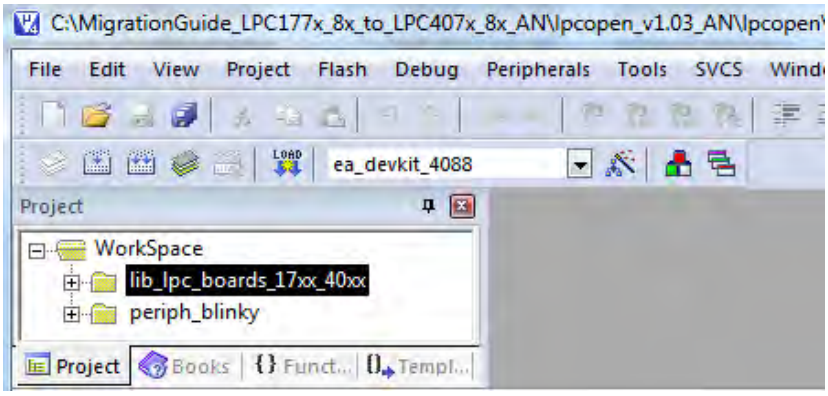

Fig 7. Project structure of LPCOpen platform

The current sample projects for LPC1788 and LPC4088 on the LPCOpen platform are based on the Embedded Artist development boards. When working with LPC1788, choose target “ea_devkit_1788”. When working with LPC4088, choose target ‘ea_devkit_LPC4088” as shown in Fig 7.

With the LPCOpen platform, it is always necessary to compile the library project first and then choose the individual application project to compile. With this multi-project shown,

we should compile lib_lpc_boards_17xx_40xx first before compiling the rest of the peripheral projects.

4.1.1.2 LPCXpresso

The sample project files for LPC177x_8x and LPC407x_8x are located in two separate folders. It is straight forward to switch to LPC407x_8x if you have used the sample projects for LPC177x_8x before. Instructions on how to bring up the LPCXpresso projects are provided for your reference.

While opening the LPCXpresso IDE platform, a dialog box appears requesting the path for the workspace. Open the workspace with the root folder which contains all the sample projects as well as the configuration files as shown in Fig 8.

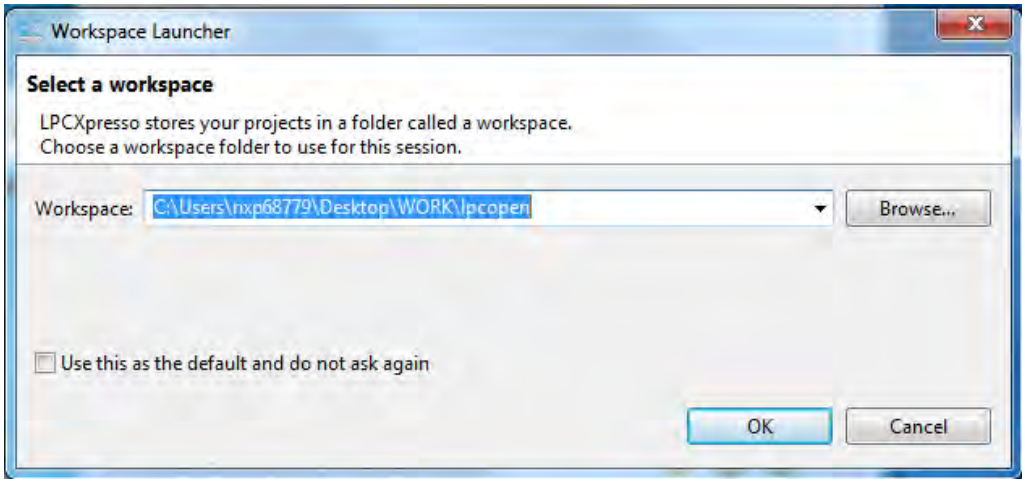

Fig 8. LPCXpresso workspace

To use the projects in LPCXpresso, they must first be imported into LPCXpresso. To import the sample projects, start LPCXpresso with a new workspace and select the "import…" option from the file menu. Select the "Existing Projects into Workspace" option from the General section and press Next.

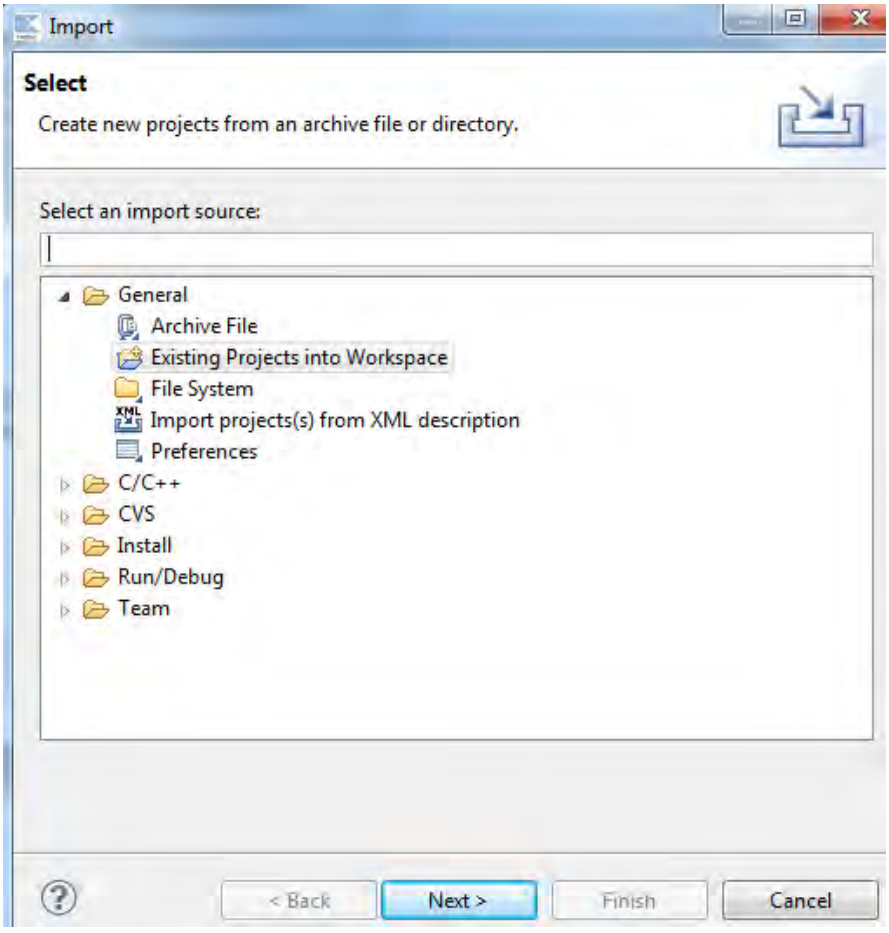

Fig 9. Import sample projects into LPCXpresso

On the import dialog window in the “Select root directory” box, browse to the \ea_devkit_1788 or \ea_devkit_4088 depending on whether you want to try the LPC177x_8x project or the LPC407x_8x project.

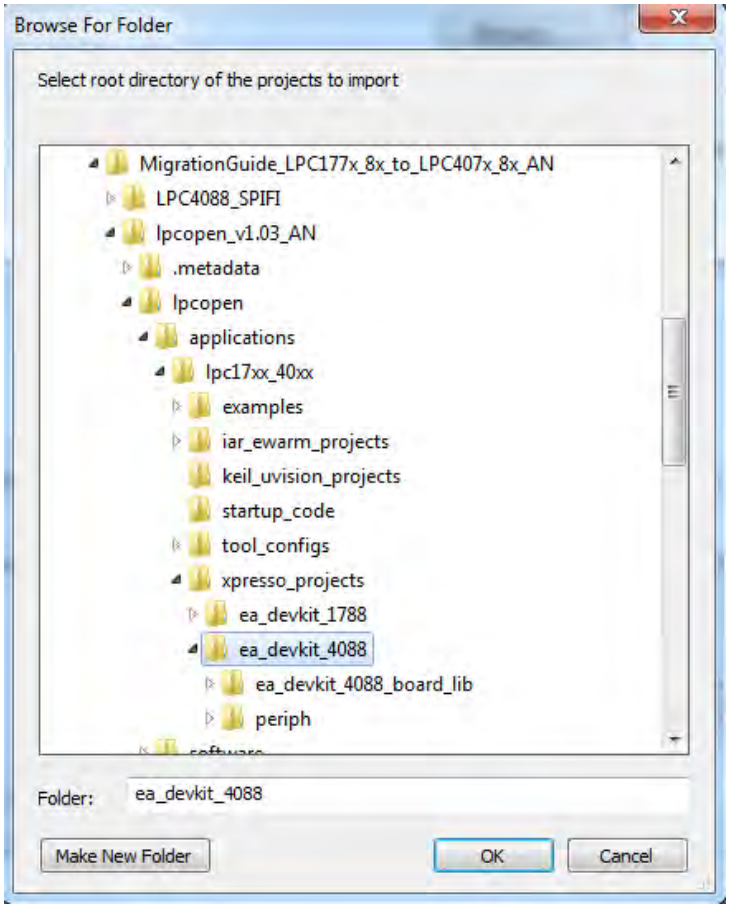

Fig 10. Browse to the sample project folder

Note that the projects under \ea_devkit_1788 are applicable to the entire LPC177x_8x system as long as the device has the features used in the sample projects; similarly, the projects under \ea_devkit_4088 are applicable to the entire LPC407x_8x system as long as the device has the features used in the sample projects. After choosing ea_devkit_4088, Fig 11 shows the two projects that can be added to the work space.

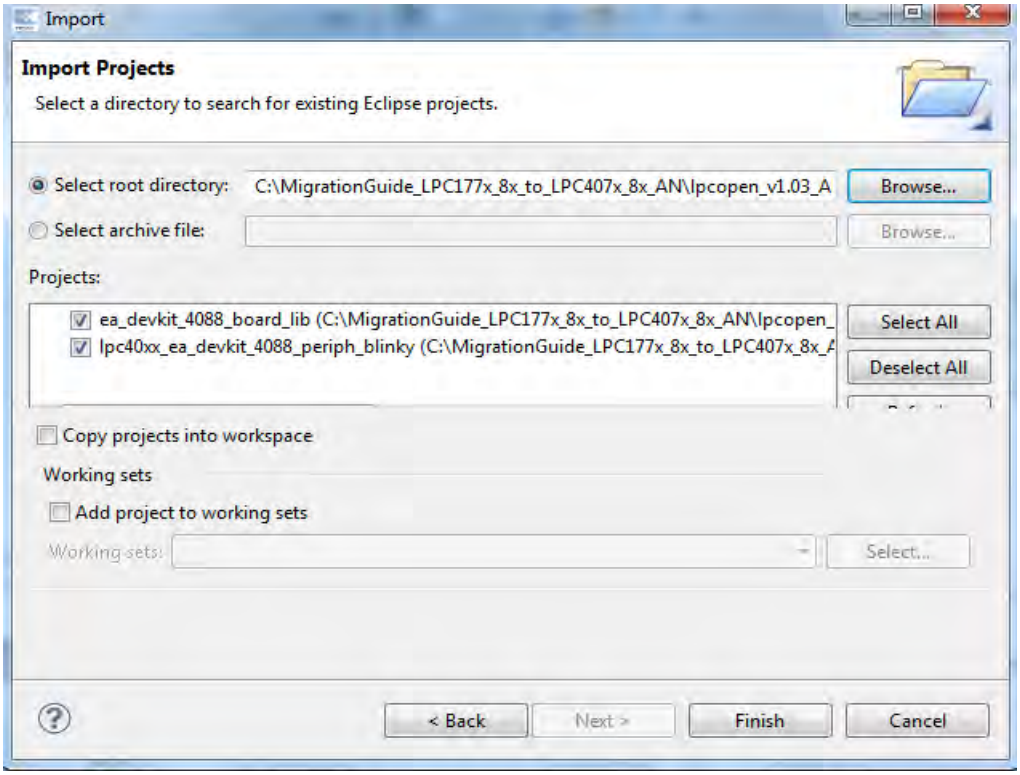

Fig 11. LPCXpresso sample project for LPC4088

Notice that you will be presented with more projects; choose ea_devkit_4088_board_lib and the particular sample project you are interested in and uncheck the rest of the sample projects. Click Finish to access the sample projects as shown in Fig 12.

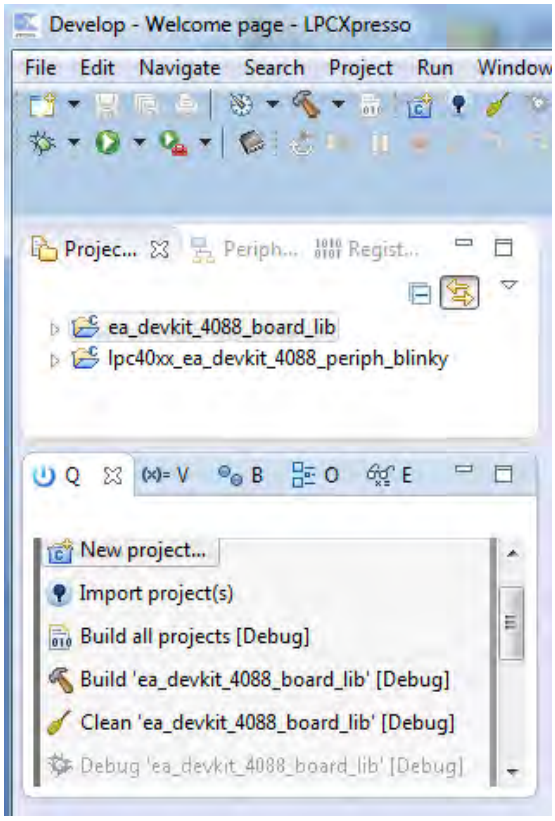

Fig 12. LPCXpresso sample projects for LPC4088

Now you are ready to run and use the sample projects for your prototyping work. Same as using the Keil and IAR sample project, it is necessary to compile ea_devkit_4088_board_lib first before compiling and running the blinky project.

After the above brief description on how to access the sample projects for LPC407x_8x, we introduce some important build time options that are used to convert between LPC177x_8x and LPC407x_8x.

4.1.2 Build time options

"CORE_M4" and "__FPU_PRESENT" are the two key build time options when migrating from an LPC177x_8x sample project to an LPC407x_8x with the LPCOpen platform. This is common to all three IDEs, only Keil IDE screen shots are provided to help illustration.

4.1.2.1 Usage of "CORE_M4"

Several places in the LPCOpen platform use the key work "CORE_M4" to differentiate systems between LPC177x_8x and LPC407x_8x.

In Project Setting

This application note uses Keil as an example to show that CORE_M4 is required to define in the project settings to allow usage of the sample code for LPC407x_8x.

In the C/C++ tab found in the "Target Options" for Target "ea_devkit_4088, "CORE_M4" is defined to allow sample code to pick the right device configurations.

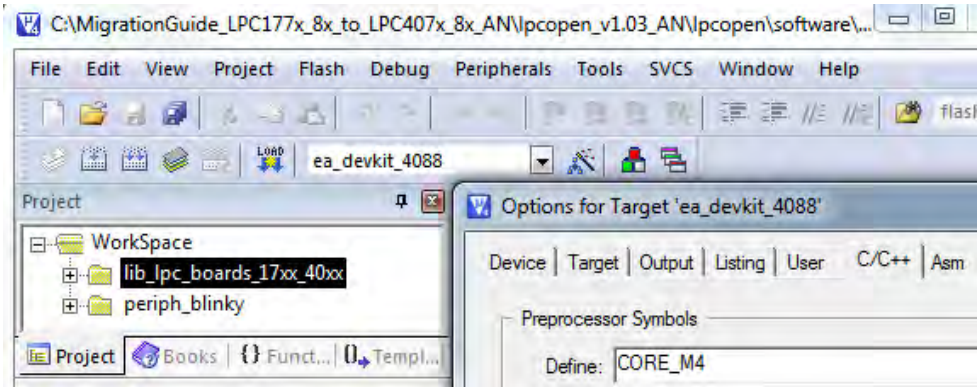

Fig 13. Defining CORE_M4

In cmsis.h

In cmsis.h, CORE_M4 definition allows the correct core and peripheral configuration to be established as detailed in Fig 14.

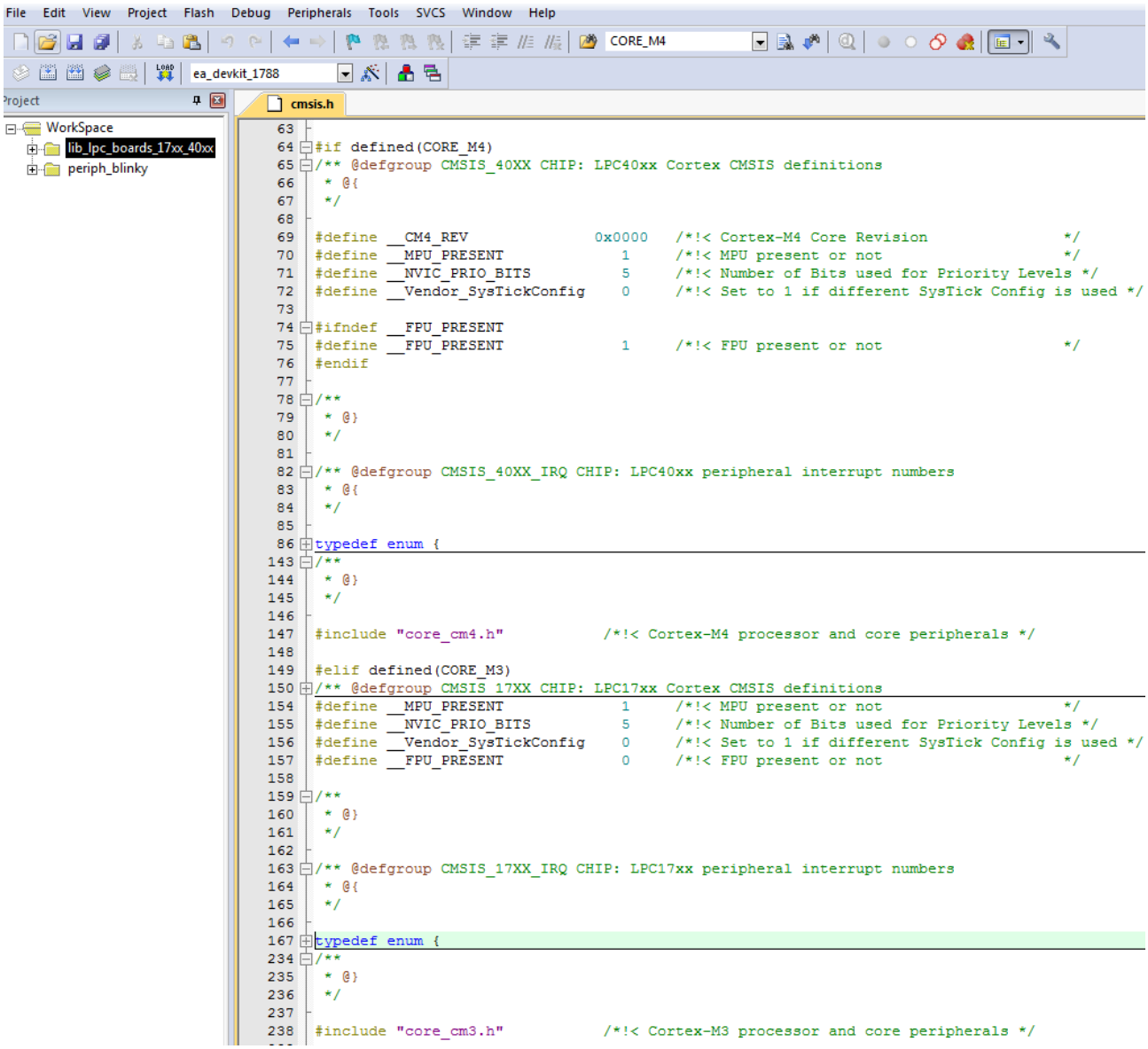

Fig 14. CORE_M4 Activates CMSIS Definition for LPC407x_8x

In fpu_init.c

In addition, CORE_M4 allows the FPU coprocessor to be properly defined and possibly initialized as shown in Fig 15.

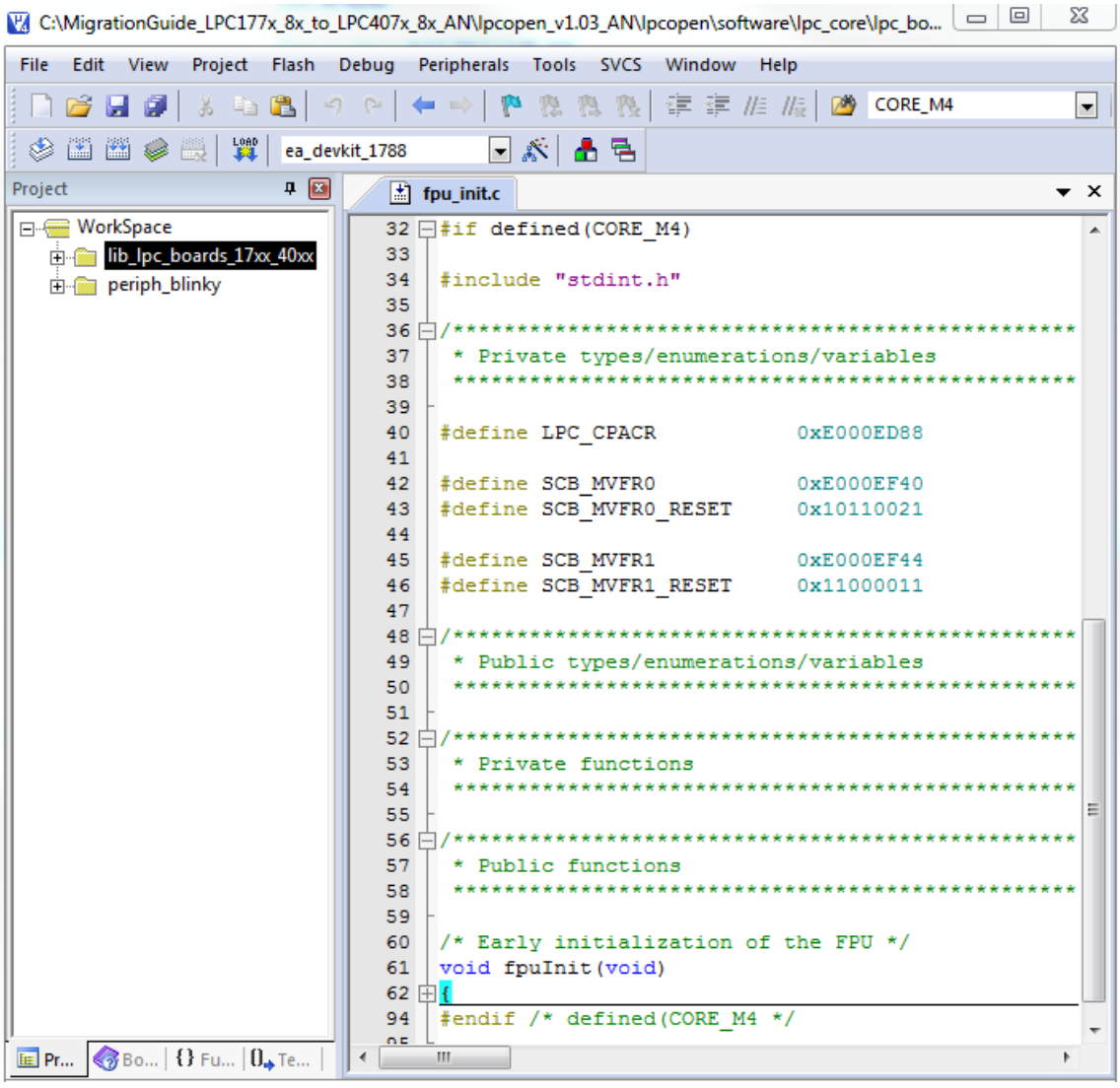

Fig 15. Referencing CORE_M4 in fpu_init.c

4.1.2.2 Usage of "__FPU_PRESENT" as build time option

Definition of __FPU_PRESENT is defined in cmsis.h. There are several parts in the LPC407x_8x family that do not implement the FPU coprocessor, for example LPC4074 and LPC4072. In this case, user should manually change (in cmsis.h) from

#define __FPU_PRESENT 1 /!< FPU present or not */*

to

#define __FPU_PRESENT 0 /!< FPU present or not */*

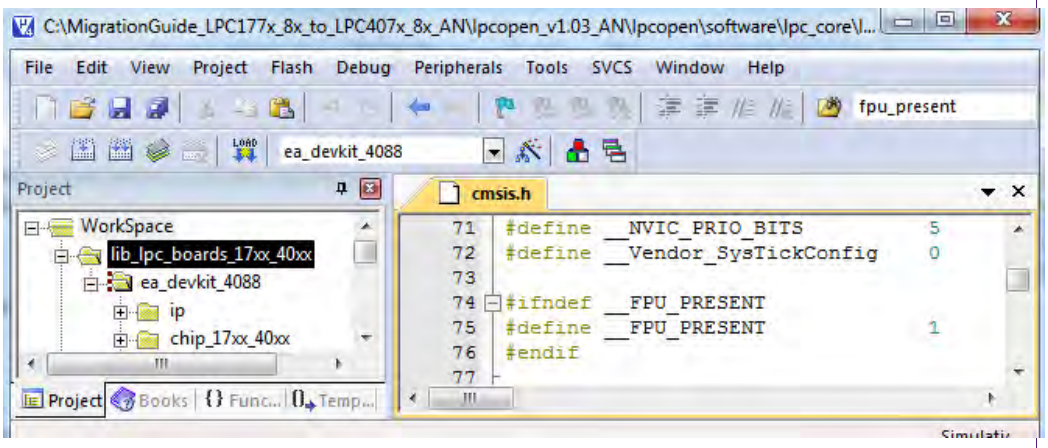

Fig 16. Definition of __FPU_PRESENT

Even though function fpu_int() is defined when CORE_M4 is defined as shown in Fig 15, this function is only called in sysinit_ea_devkit_17884088.c **if** __FPU_PRESENT is set to 1 as shown in Fig 17.

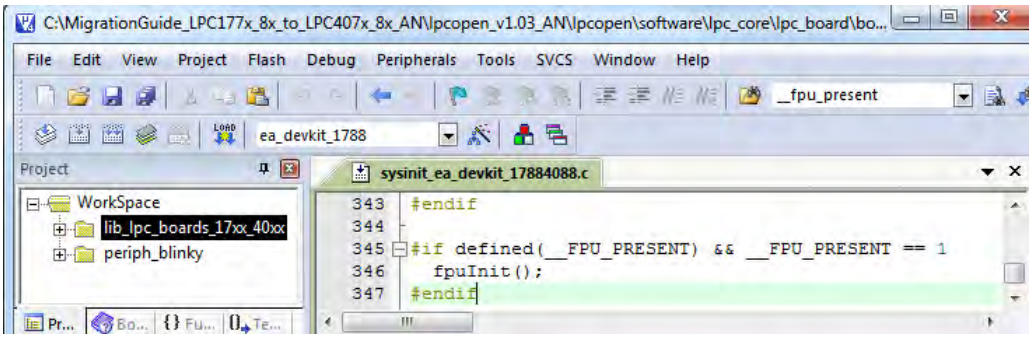

Fig 17. __FPU_PRESENT set to 1 switches on FPU initialization

For more details on these build time options, please refer to the following link from the LPCOpen platform:

http://docs.lpcware.com/lpcopen/v1.03/group_chip_17xx_40xx_driver_options.html

4.1.3 Sample projects with Blinky running out of SPIFI on LPC407x_8x

Besides the Blinky sample projects described in Section 4.2.1, another Blinky project based on the Peripheral Driver Library package is provided with this application note. This project will be added in the LPCOpen platform in the near future.

This project uses the LED blinking speed to signal the potentiometer position on the EA LPC4088 OEM board. The code that controls the LED blinking runs out of the SPIFI memory space and thus this project is a simple example of how to link code to run from the SPIFI device attached to the LPC4088. This project is validated with EA's LPC4088 development board.

The LED blinking code is located in the SPIFI memory area through the Keil compiler options.

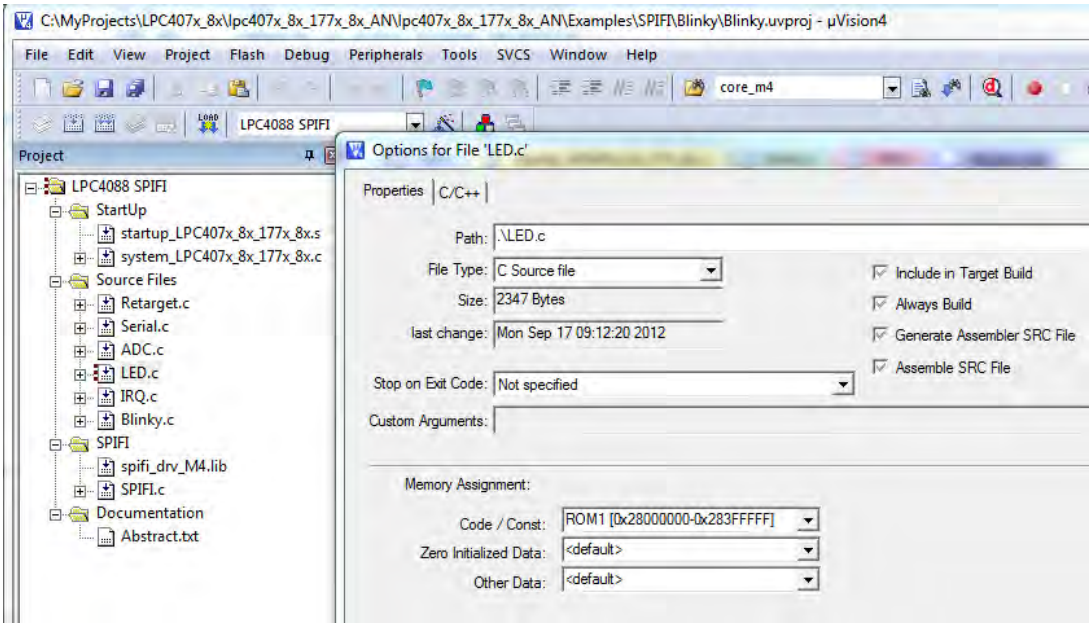

Fig 18. Locate LED.c to SPIFI memory

4.1.4 Specific notes on Embedded Artists' development board

When hardware platform is changed from LPC177x_8x to LPC407x_8x, there may be some hardware discrepancies between these platforms due to the fact that the same numbers of pins are muxed onto a richer group of peripherals for LPC407x_8x. For example, on Embedded Artists' (EA) development kit, there are some hardware differences between the LPC1788 daughter board and the LPC4088 daughter board. For details please see Fig 19. This difference came from the fact that LPC1788 does not have a SPIFI port.

	Hardware Difference	Sample project, driver, BSP influenced	Solutions
1	P2.7 position (on LPC1788) is replaced by P0.10 (since P2.7 used by SPIFI on LPC4088).	LCD data bus	Configure P0.10 as LCD_VD[5] to make the LCD work for LPC4088.
2	SSP0 pins on LPC1788 are replaced by SSP2 pins (since SSP0 pins used by SPIFI on LPC4088).	All that are using SSP0 ports (including LCD touch screen)	switch SSP0 to SSP2
3	P0.10 and P0.11 on LPC1788 is replaced by P4.22 and P4.23 on LPC4088 (since P0.10 is used for LCD on LPC4088)	All that is using P0.10 and P0.11 (UART 2 in LPC1788 and LPC4088)	Reconfigure to use P4.22 and P4.23 in place of P0.10 and P0.11

Fig 19. EA OEM board difference LPC1788 vs. LPC4088

In the sample projects, the LPCOpen platform uses CHIP_LPC177X_8X and CHIP_LPC407X_8X to pick the proper hardware configuration. Fig 20 is an example of such handling used in board_ea_devkit_17884088.c for SSP communication port selection as requested from the hardware difference item 2 shown in Fig 19.

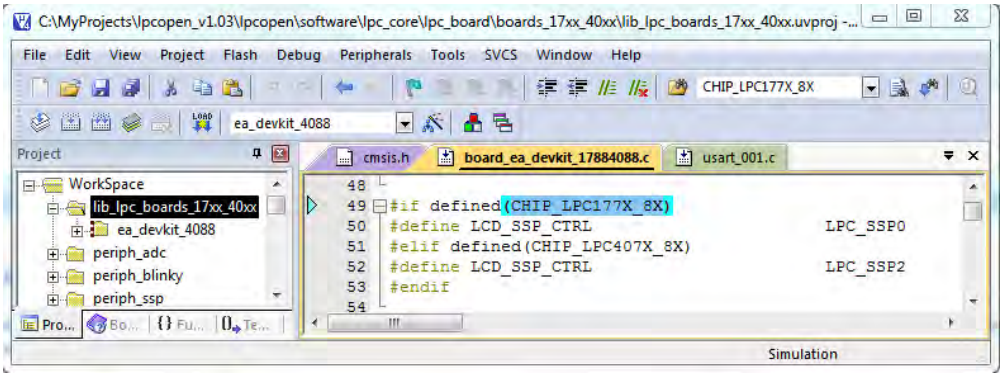

Fig 20. Switching between LPC1788 and LPC4088 on EA development board for SSP communication

5. Conclusion

LPC407x_8x is a pin-compatible Cortex-M4 option for LPC177x_8x. Migration of a project from LPC177x_8x to LPC407x_8x is relatively simple. Functioning hardware for LPC177x_8x should work with LPC407x_8x with the same pin count. There might be minor timing adjustments needed when dropping an LPC407x_8x into an LPC177x_8x hardware design. There is already a huge group of sample projects available on our LPCOpen platform that includes RTOS, LCD graphics, TCP/IP and USB applications for both platforms. For the additional FPU coprocessor, SPIFI interface as well as the dual analog comparator, NXP will provide complete software sample projects to help customers get up to speed and quickly benefit from these enhancements.

6. Legal information

6.1 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.

6.2 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.

6.3 Trademarks

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

7. List of figures

8. Contents

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

Date of release: 1 October 2013
Document identifier: AN11335