AN11281

Quick Start Up Guide PREV601 Demo Board

Rev. 1.5 — 10 November 2013
243915

Application note
COMPANY PUBLIC

Document information

Info	Content
Keywords	PREV601 board, PRH-601, LPC1227, CLRC663, LPCXpresso, LPCD, NxpRdLib, HTRC110, MCU, Code Red, eclipse, LPC1227, reader library
Abstract	This application note is related to the installation procedures of the PREV601 board. It describes the actions to be done to become acquainted with the demo reader.

Revision history

Rev	Date	Description
1.5	20131110	Added a note about the LPCXpresso IDE version in chapter 4
1.4	20130613	Added description about the P2P Snep Client
1.3	20130608	Small alignments between new project structure and this document
1.2	20130116	Small alignments between the referenced code and this guide.
1.1	20130107	Added chapter 6.4 about the hardware setting Added chapter 6.5 about the LPCD mode
1.0	20121019	First release

Contact information

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

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

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

1. Introduction

This application note gives a detailed overview of the hardware for working with the PRH601 integrated reader module (**Chapter** 2), the installation procedures of the Development Environment (**Chapter 4.1**) and the handling of the reader projects using the NXP Reader Library (**Chapter 4.2**).

The PREV601 board can be used for software development for the PR601 as well as the PRH601 reader IC. The only difference between these ICs is the additional 125kHz capability of the PRH601 IC.

The projects used in this documentation are:

- Communication with MIFARE Ultralight → **Chapter 5.1**
- Communication with MIFARE Classic → **Chapter 5.2**
- Communication with MIFARE DESFire → **Chapter 5.3**
- Polling for Tags in the RF - field → **Chapter 5.4**

All projects can also operate in Low Power Card Detection Mode [11].

2. Hardware overview of the Demo Reader

The demo reader is made up of one single board including two reader ICs and one processor with flash memory. The IC that contains these reader ICs and the processor is the PRH601HL. For more details, please see our website at [9]. The multi chip package contains the following ICs:

- A CLRC663 Reader IC. For detailed information please see [2].
- A HITAG HTRC110. For detailed information please see [7].
  The HTRC110 is supported by breakout pads. A developer has to attach his own antenna incl. matching to make use of the 125KHz functionality of the PRH601. Software for the 125Khz functionality can be found on the NXP website [8].
- A LP1227 IC. For detailed information please see [3]

To flash the LPC1227 micro controller, one also needs a LPC-Link board (LPC3154)

2.1 PREV601 board

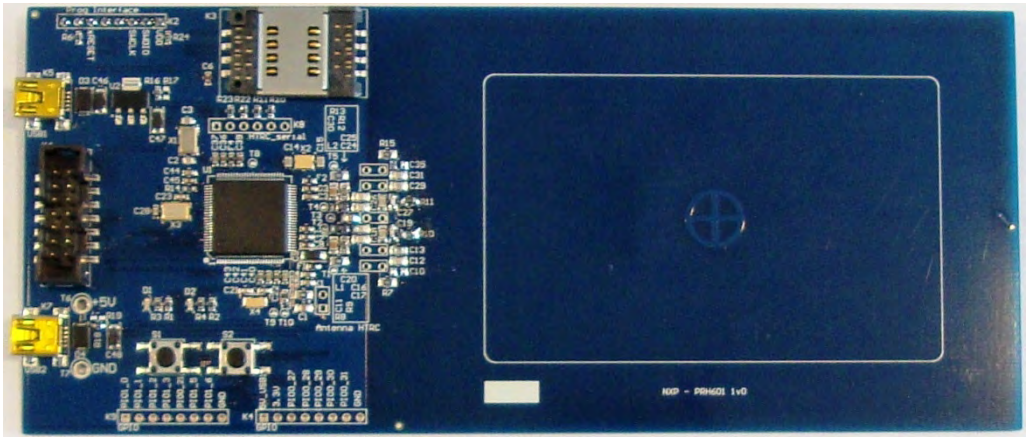

Fig 1. Picture of the PREV601 board

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

The PREV601 board embeds the contactless communication transceiver IC CLRC663, HTRC and LPC1227 processor.

The software provided with this evaluation board also works with the significantly smaller PREV601M Microboard.

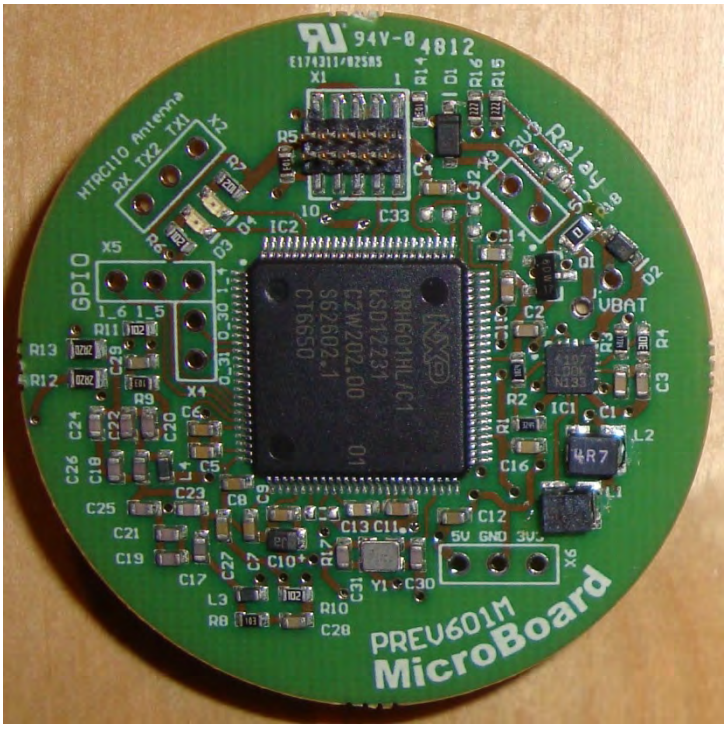

Fig 2. PREV601M Microboard

2.2 LPC-Link LPC3154

The LPC-Link is equipped with an 8-pin JTAG header and it seamlessly connects the target via USB (the USB interface and other debug features are provided by NXP's ARM9 based LPC3154 MCU).

Since this LPC-Link is not available as single device, one has to cut that part from any LPCXpresso board. The LPCXpresso boards containing the LPC-Link with the LPC3154 MCU are as follows:

- LPC11C24 LPCXpresso Board
- LPC11U14 LPCXpresso Board
- LPC1114 LPCXpresso Board
- LPC1115 LPCXpresso Board
- LPC1227 LPCXpresso Board
- LPC1347 LPCXpresso Board
- LPC1343 LPCXpresso Board
- LPC1769 LPCXpresso Board

To use the LPC-Link, one has to cut it off from the LPCXpresso board.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

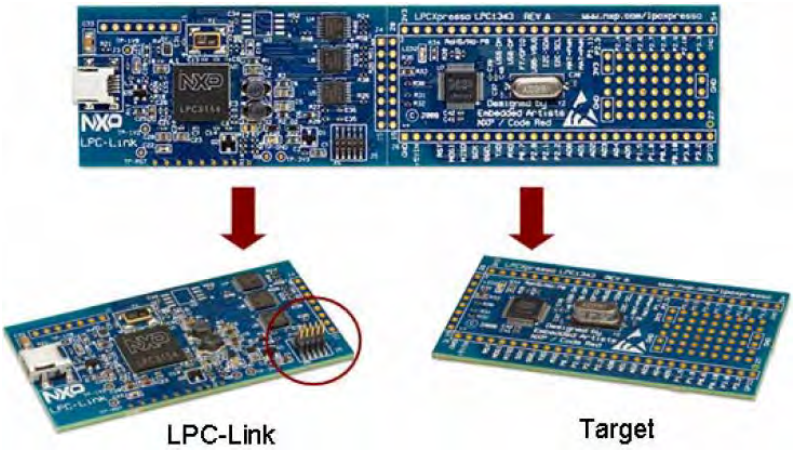

Fig 3. LPC-Link

2.3 Preparation of the hardware

The first step after unpacking the PREV601 board and the LPC-Link board is soldering the connectors onto the boards in order to attach them. In our example we use a multipoint connector as one can see on the pictures below.

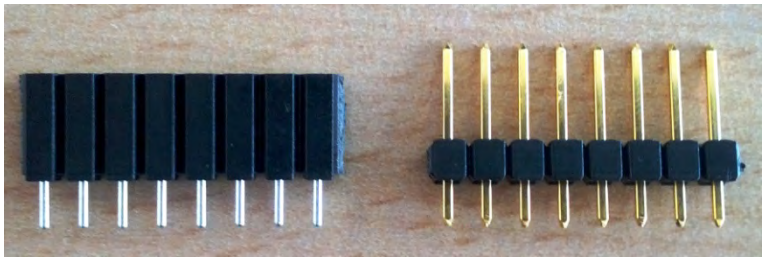

Fig 4. Multipoint Connectors

One may buy these connectors at any electronic store. Here are some examples [4]. After soldering the connectors, connect the boards as shown on the following figure.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

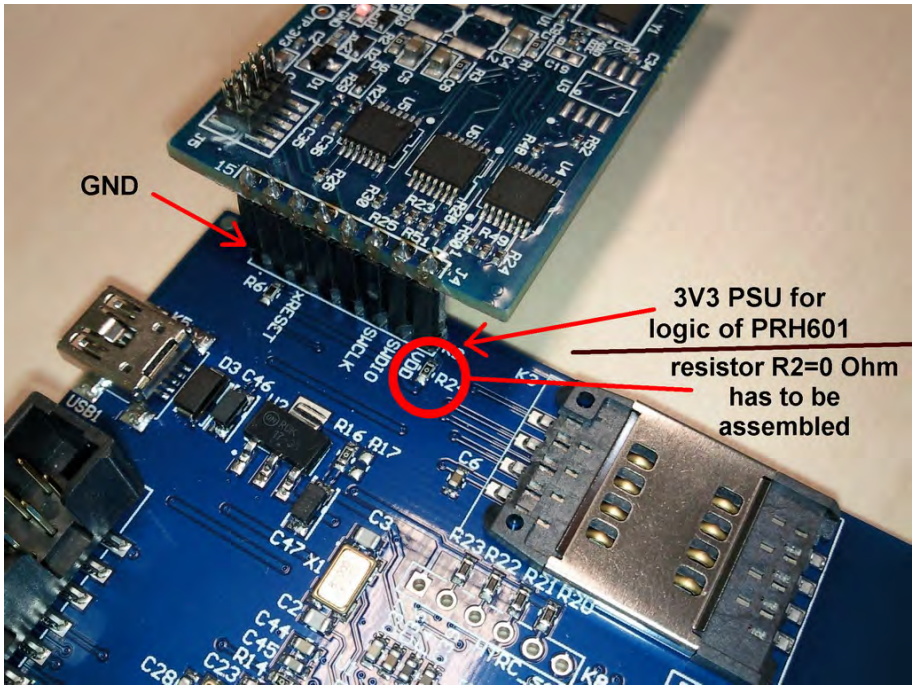

Fig 5. Connecting the PREV601 board to the LPC Link



Fig 6. LPCXpresso with the Multipoint Connectors

While soldering the connectors onto the board, an opportunity arises to solder either a bridge or - as shown on Fig 5- a zero ohm resistor near the program interface (R24). After bridging this connection, you'll need two instead of three USB cables connected like shown on Fig 6.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

Now the hardware is ready for use.

3. Connecting the hardware to a computer

The guidelines to install the reader are as follows:

- Connect the LPC-Link board to a real USB2.0 port of the PC using the mini-USB connector. The PC detects and installs the board automatically.
- Once the board has been installed, open the Device Manager of the PC to check that the installation has been successful. The item “USB Device with DFU Capabilities” is being displayed.

After connecting the LPC-Link board, please also connect the second USB connector on the PREV601 board to the computer. In summary one will need to connect 2 USB cables (see picture above).

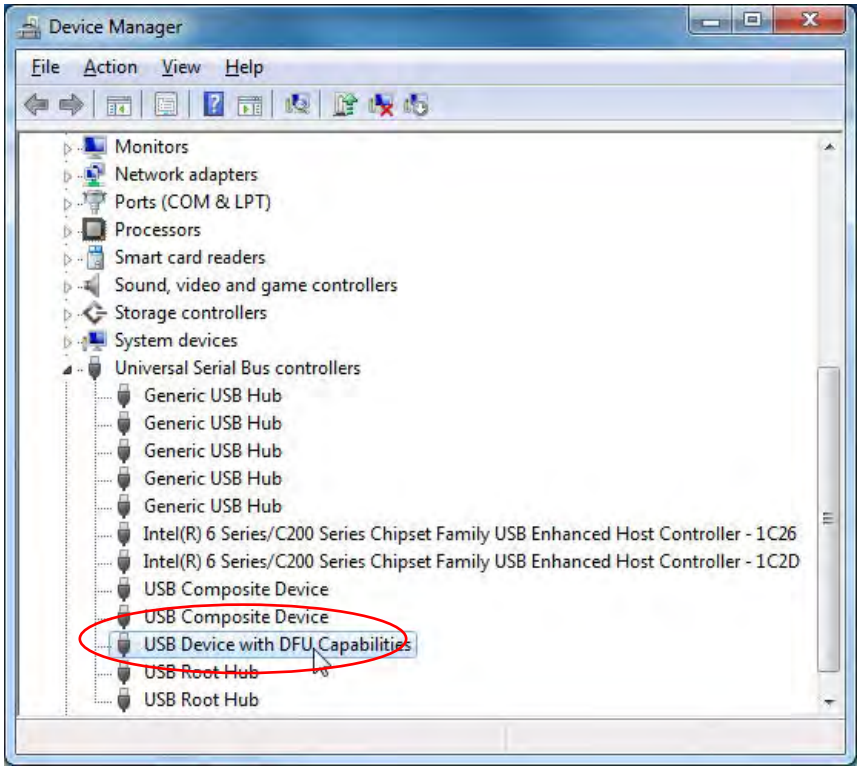

Fig 7. Enumeration of the LPCXpresso Board in Device Manager Window

4. Managing the Demo Reader project with LPCXpresso IDE

The demo reader project is delivered in a zip package. It can be extracted, edited, compiled and linked with LPCXpresso™ IDE.

LPCXpresso™ is a new, low-cost development platform available at NXP. It supports NXP's ARM-based LPC microcontrollers. The platform comprises a simplified Eclipse-based IDE and low-cost target boards which include an attached JTAG debugger.

For development please use the LPCXpresso version 4.x. Newer versions of the IDE are known not to work correctly with the provided software examples.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

This tool can be freely downloaded from the LPCXpresso website [1]. Before one can download the software, it is necessary to create an account. Creating an account is absolutely free.

4.1 Installation of LPCXpresso IDE

The IDE is installed into a single directory, of one's choice. Multiple versions can be installed simultaneously without any issues. The installation starts after double-clicking the installer file. Then click "next" on the setup wizard.



Fig 8. LPCXpresso installation setup wizard 1

Read the license agreement then click next.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

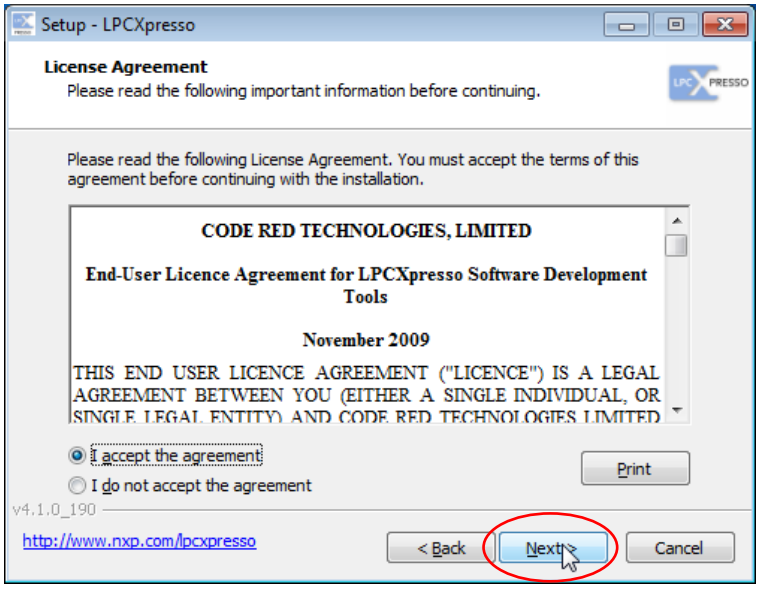

Fig 9. LPCXpresso installation setup wizard 2

There are numbers of other screens on the setup wizard but generally the default options can be accepted. After installation, an information file will be displayed. Click “Next” to accomplish the installation.

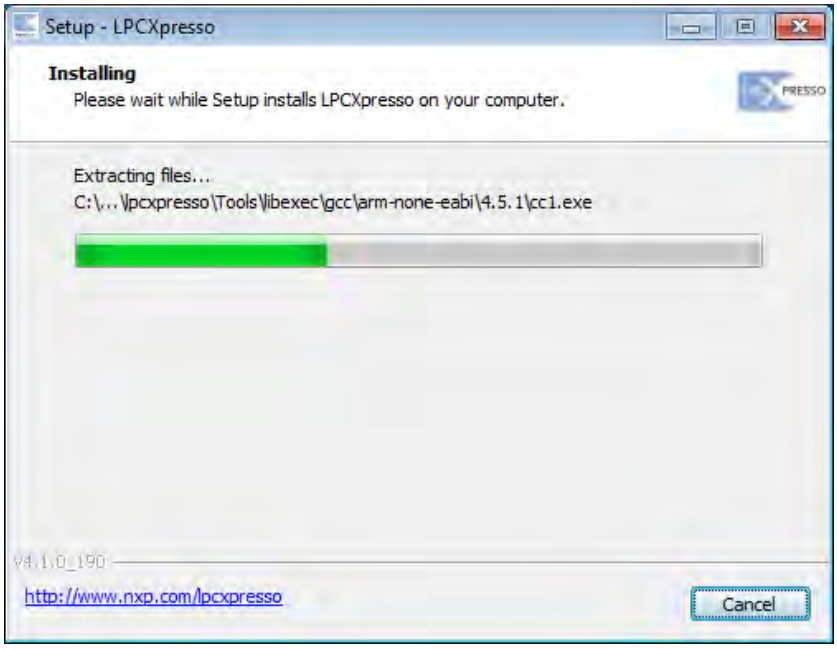

Fig 10. LPCXpresso installation setup wizard 3

After this installation step one will be asked if he wants to install some required drivers. The installation of these drivers should be accepted.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

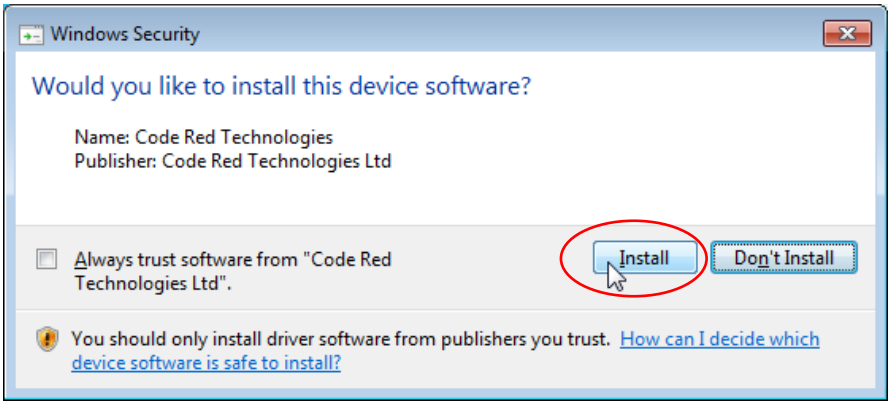

Fig 11. Windows Security dialog

After the setup wizard has finished one can launch the newly installed IDE.

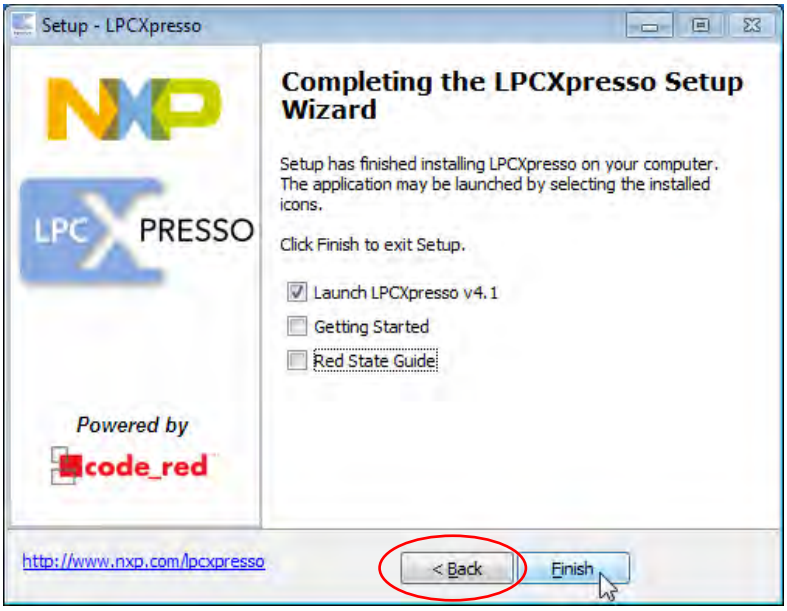

Fig 12. LPCXpresso installation setup wizard 4

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

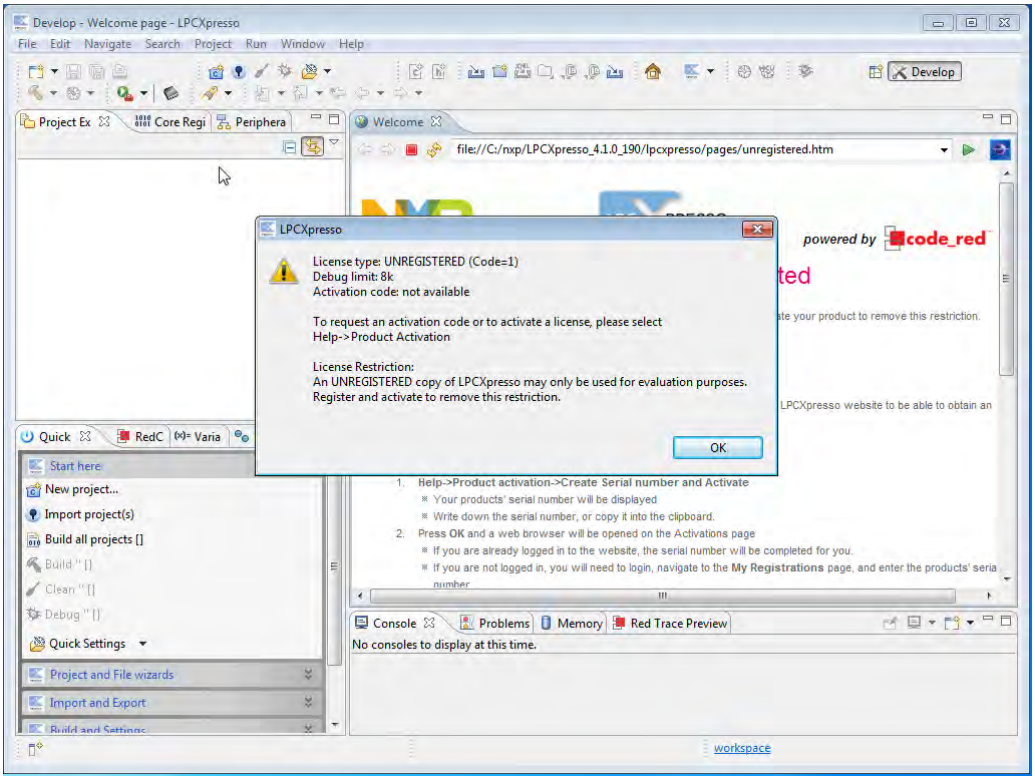

Fig 13. LPCXpresso IDE

Directly after the first start of the Eclipse IDE one will see an info dialogue that this is only an unregistered copy of LPCXpresso IDE. Just confirm the dialog and follow the instructions on the Welcome Screen to get a registered version without the debug limit of 8k. The registration is free and is needed to navigate to the website of Code Red. The Link is shown in the menu, Help → Product activation → Create Serial number and Activate...

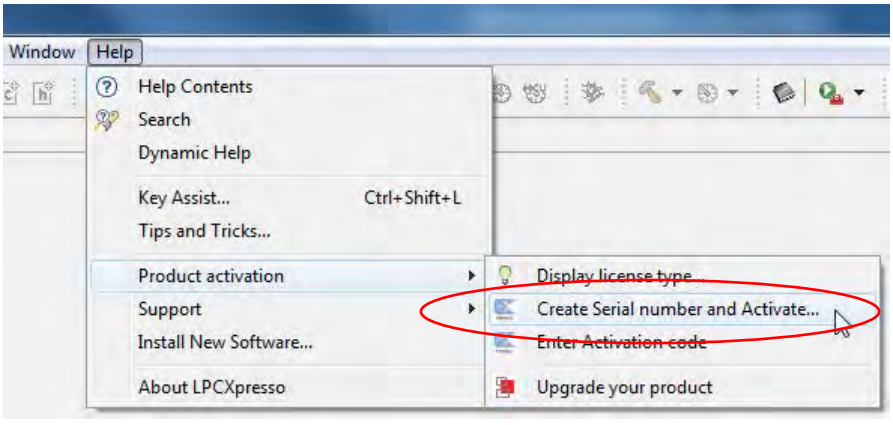

Fig 14. Product activation

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

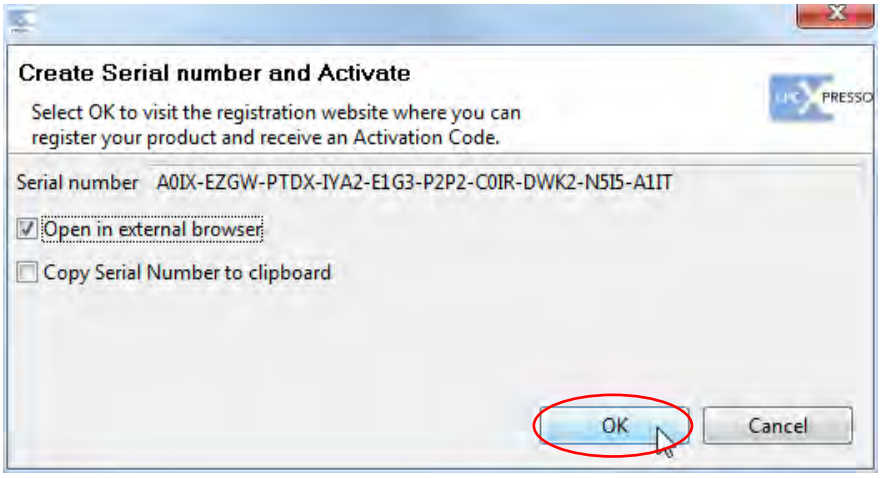

Fig 15. Product activation

If one doesn't already have an account at Code Red, please sign up to get an activation code. The code will be sent to the provided e-mail address.

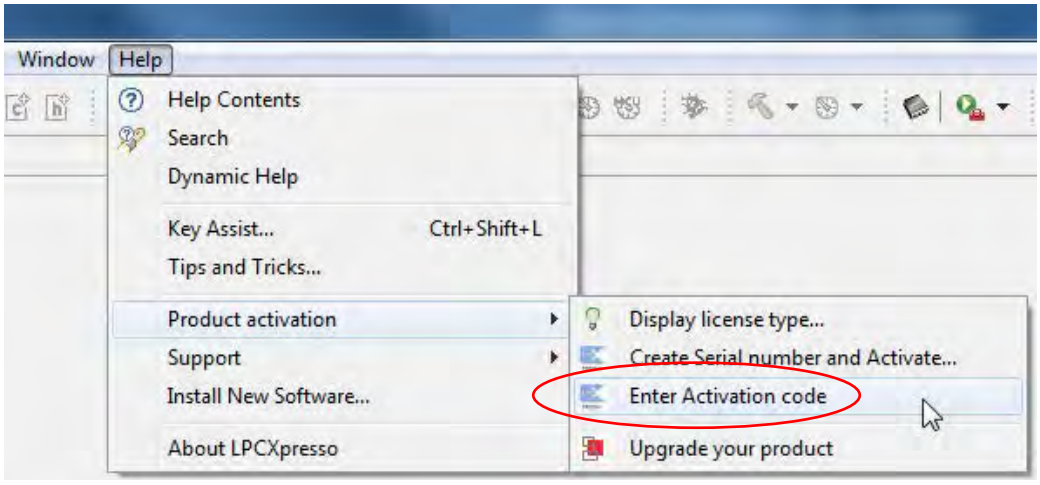

Fig 16. Product activation

Once the activation code arrives please open the activation window by pointing to Help → Product activation → Enter Activation code, and enter the code.

The success of the product activation will be confirmed by an info dialogue.

4.2 Extraction of the demo reader project

Once the LPCXpresso™ IDE has been installed on a computer, the sequence of installing the reference reader project is indicated:

- Start the LPCXpresso™ IDE.
- Select the option "Import project(s)" (see picture below).
- Browse the zip archive.
- LPCXpresso™ IDE unzips the software package.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

- The software package is ready for use.

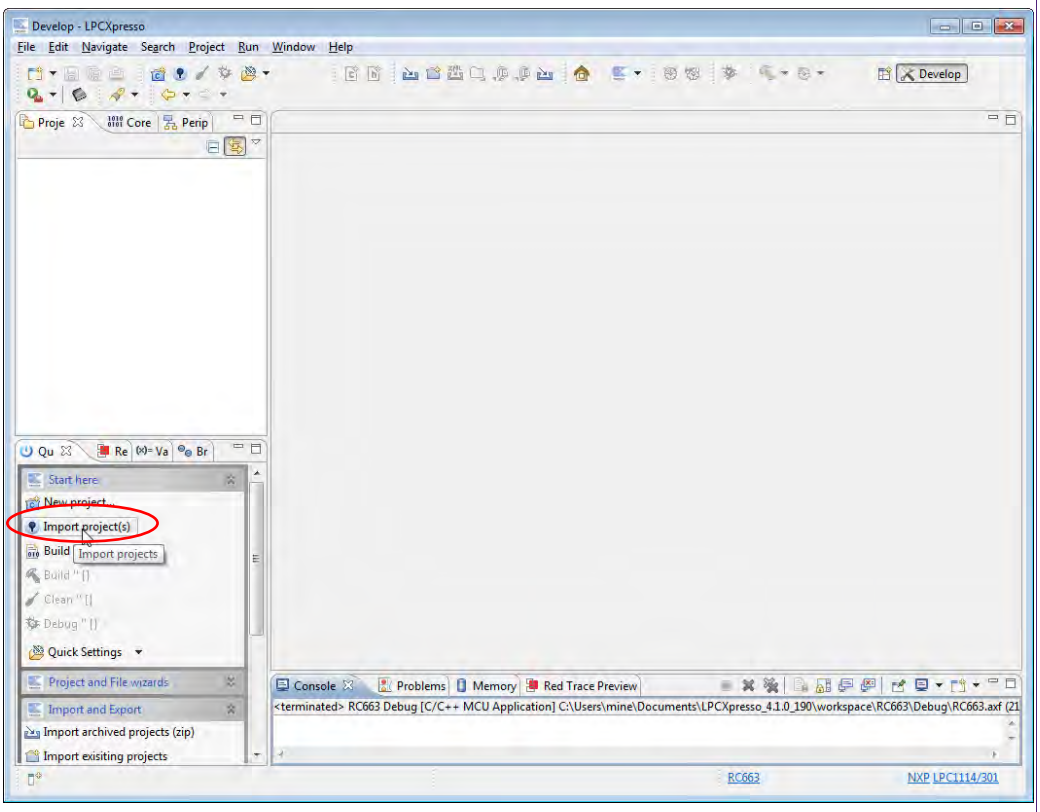

Fig 17. Importing a project into the LPCXpresso IDE

In the Quick Panel on the left hand side, choose “Import projects(s)”.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

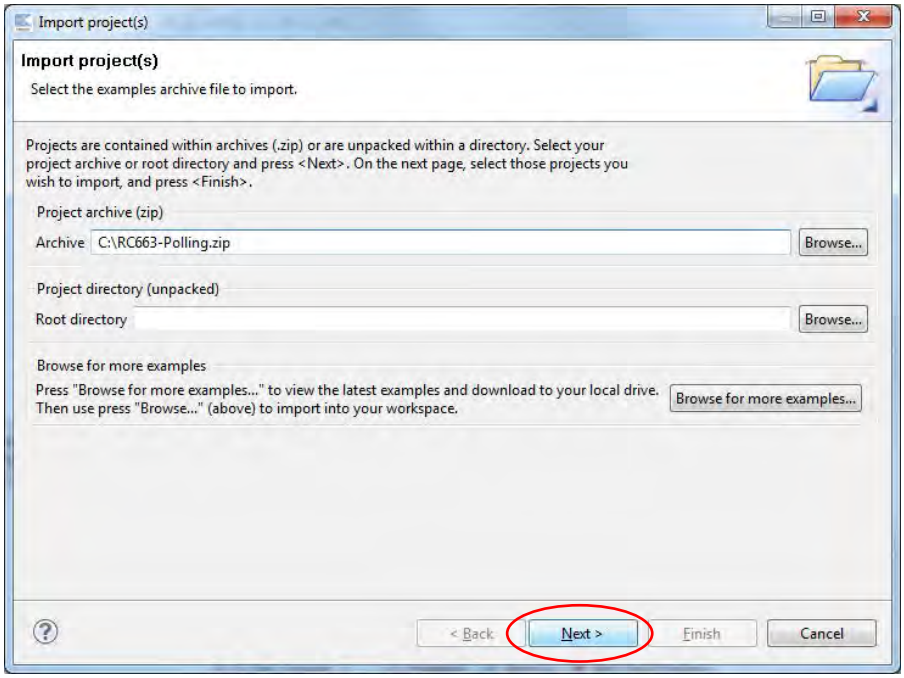

Fig 18. Importing a project into the LPCXpresso IDE

Browse the desired project and click “Next”.

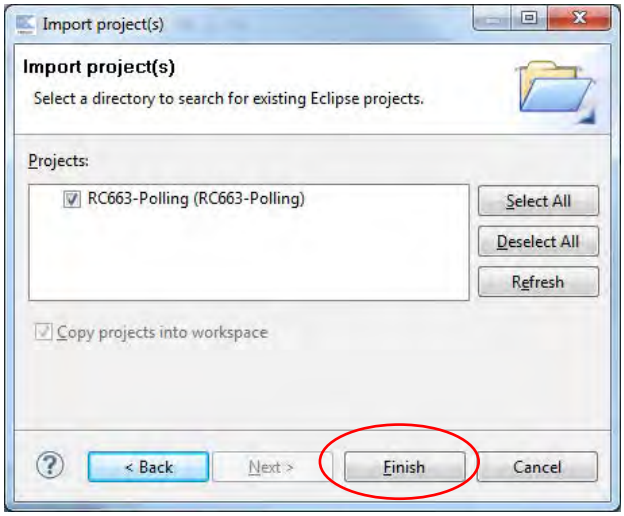

Fig 19. Importing a project into the LPCXpresso IDE

When the import process has finished one can start browsing the code. Most interesting might be the main.c which is located in ../src/main.c in the project.

Before one can run the project, the PREV601 board with the PRH601 needs to be connected to the computer. Wait until the adequate drivers have been installed.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

4.3 Start the project

One can quickly start the reader project by editing the main function in the module **main.c**. This function first performs the hardware initializations of the LPC1227 and the RF transceiver RC663.

Detailed descriptions of the code in the form of comments have been provided in the **main.c** file. This should provide a detailed overview of how to initialize certain components and get data out of and onto the card in the RF field.

4.3.1 Run the project

Before running the project, please ensure that the LPCXpresso with the Blueboard is connected to the computer.

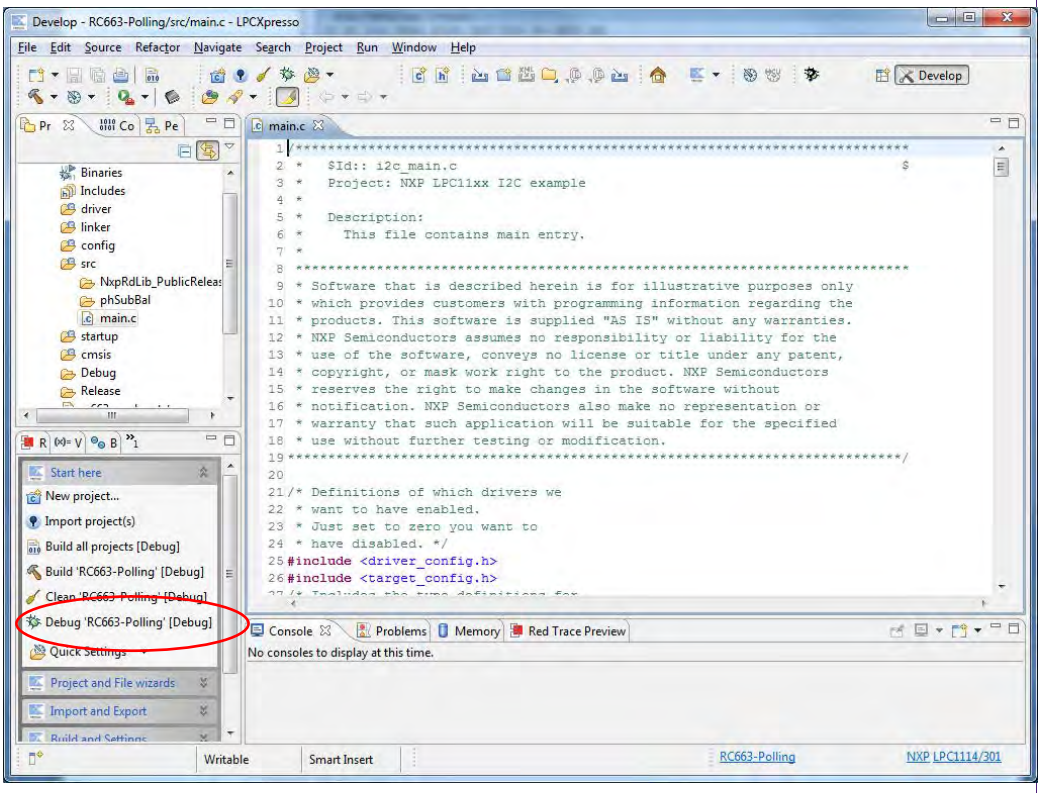

Fig 20. Run the project

Choose the desired project and click the Debug Button on the left hand side as shown in the example picture.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

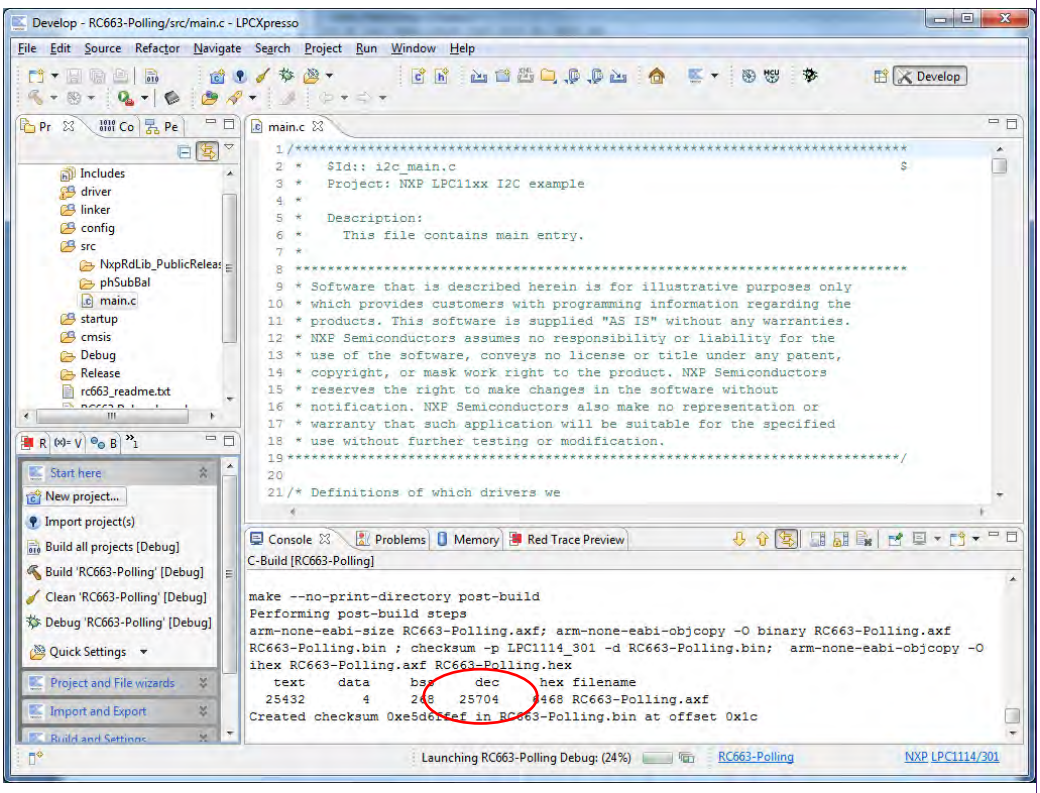

Fig 21. Run the project

After the build process one can see the size of the image in the console window.

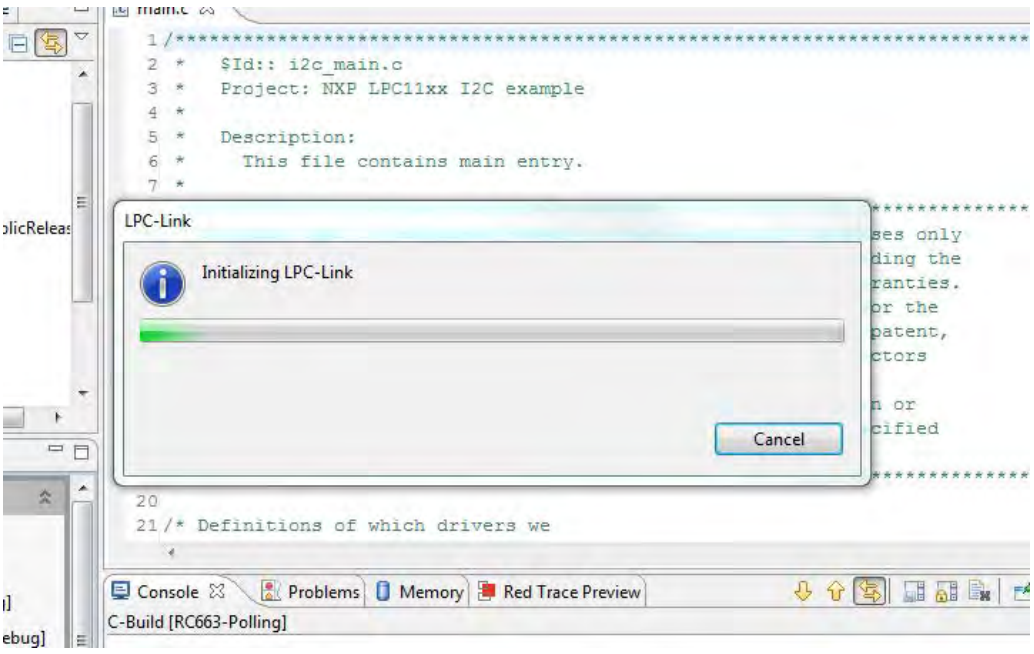

Fig 22. Run the project

The initialization of the LPC-Link can take a few seconds.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

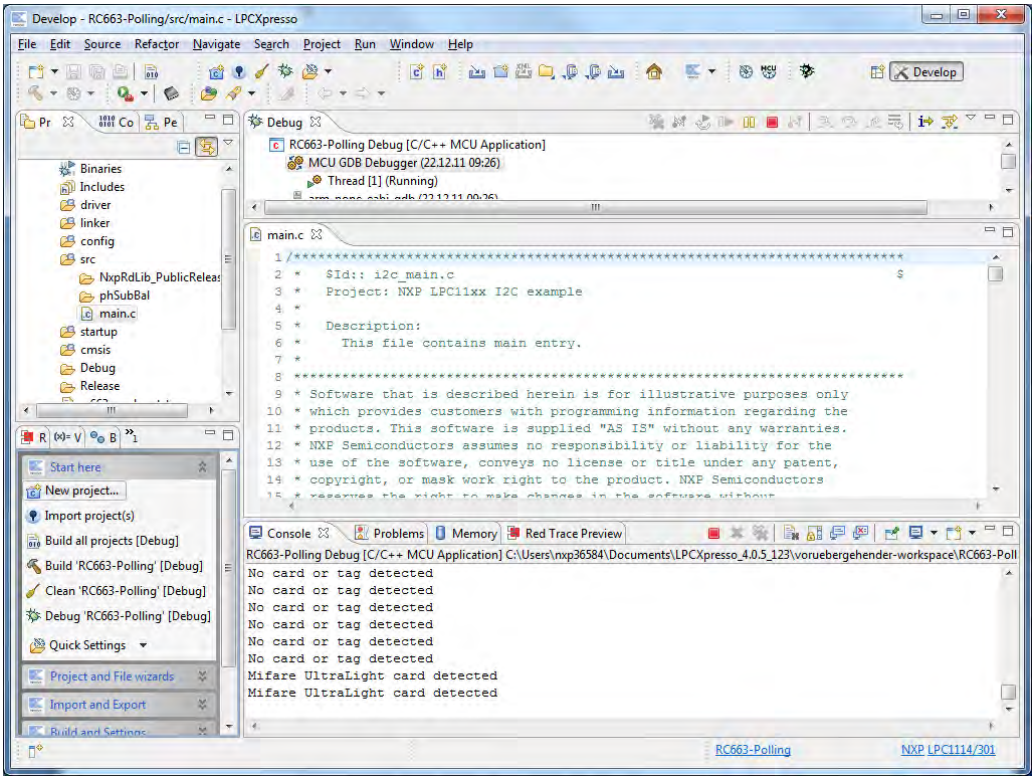

Fig 23. Run the project

After the software upload, the execution of the project starts immediately.

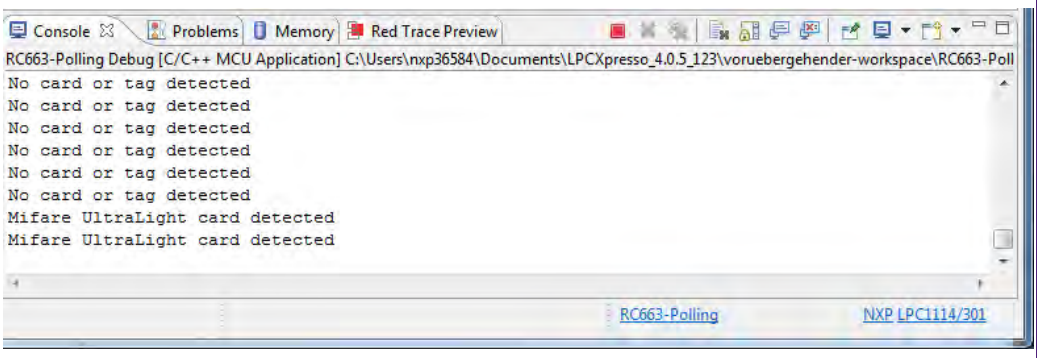

Fig 24. Run the project

In the console window at the bottom one will see the debug output of the execution.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

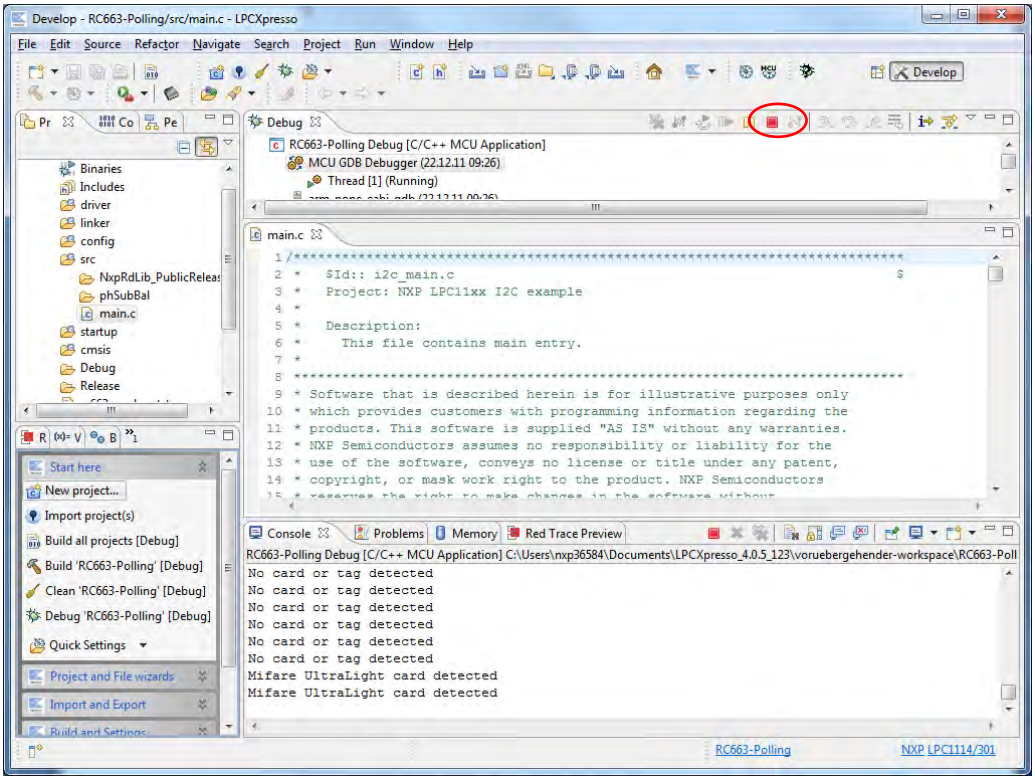

Fig 25. Stop the project

After the execution has reached the end of the main function please click the Terminate button to stop the execution. Otherwise one won't be able to rerun the project.

One can now do the following with the buttons near the top of the "Debug" view:

	Run the program.
	Step over C/C++ line.
	Step into a function.
	Stop the debugger.
	Pause execution of the running program.
	Instruction stepping mode (disassembly).

Fig 26. Debug Buttons

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

5. Associated Projects

5.1 Communication with MIFARE Ultralight

Based on examples the MIFARE Ultralight project shows how read-write access can be achieved on this type of card.
If one uses a card which is not write protected or secured the example program writes a valid NDEF message onto the card. One can read this message with any NFC enabled mobile phone which can read NDEF messages.

5.2 Communication with MIFARE Classic

Based on examples this project shows how read-write access can be achieved on this type of card.

5.3 Communication with MIFARE DESFire

Based on examples this project shows how read access can be achieved on this type of card.

5.4 Polling

Based on examples this project shows how to initiate a basic communication with the following cards:

- MIFARE Ultralight
- MIFARE Classic
- MIFARE Plus
- MIFARE DESFire
- FeliCa compliant cards
- ISO/IEC 14443-B cards
- ISO/IEC 15693 Tags

This example project also looks for cards in range of the RF field in a continuous loop and returns the type of the detected card or tag to the console window.

5.5 Peer to Peer functionality

Based on examples this project shows an implementation of Peer to Peer (P2P) functionality.
Because the P2P implementation is still in alpha phase, one will find very rudimentary support for the protocol.

At the moment the project supports the communication via the SPI protocol and runs on the LPCXpresso LPC1227 development board.

5.5.1 Installation

After downloading and unpacking the zip file, please run the installer. The installer just copies the LPCXpresso project files to the file system. After the installation has finished, please run the included batch file located in the installation directory

`…\NxpRdLibP2PExtensions-x.x.x\NxpRdLib_P2PExtensions\ex\Rc663_Lpc12xx_P2P_Demo`

After the batch file has been executed successfully, please start the LPCXpresso IDE and import the project. Browse to the root of the installation directory.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

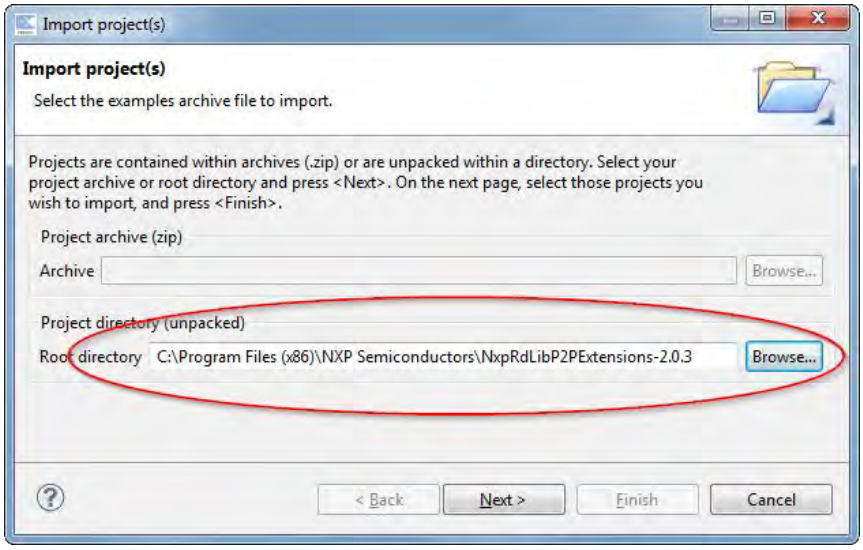

Fig 27. Importing the project

After the import there are two new projects in the workspace - one for the use with the RC663 Blueboard or PREV601 board and one for the use with the PN512 board. Basically these two projects provide the same functionality.

5.5.2 SNEP client project

In this project the PREV601 board behaves like SNEP client with performing the Put request functionality only [14][14]. On the other side there must be a device – peer capable of NFC communication and providing the SNEP server service. These conditions are fulfilled by a mobile device with Android platform (4.0 or later).

Here is a short instruction list how to use the SNEP client software:

1. Run the SNEP client project as described in section 4.3.1 with respect to LPC1227 microcontroller restriction and hardware configuration in section 6.4.
2. Hold your NFC device (smartphone) approximately 5 centimeters above antenna of the PREV601 board.
3. Once the NFC detected in the RF field of the transmission of the NDEF message is started. You may be notified by a sound or vibration of your device. The transmission may take several seconds depending on the size of the NDEF message. Hold your NFC device in range of PREV601 board's field during entire transmission.
4. After transmission completed there should be the transmitted image displayed in the device's screen.

5.5.2.1 What is going on inside?

The SNEP client sends a hardcoded NDEF message encapsulated in a SNEP message to the NFC device. Thanks to the SNEP client project implements the P2P Reader Library Extension compliance with the LLCP and ISO18092 protocols on the PNEV512-Board's side are ensured, implicitly a correct SNEP client-server communication.

Execution of the SNEP client software can be summarized in following steps:

1. Hardware initialization

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

2. Detecting the RF field for an NFC peer of the tag type F.

   The software checks the RF field whether there is tag type F capable of performing the P2P communication

3. Once such device is found, the LLC link is activated in compliance with the procedure defined by the NFC forum [15].

4. LLCP socket creation and establishing connection with other peer – SNEP server.

5. Transmission of a given image file to the SNEP server:

   The SNEP client sends an initial fragment 128 bytes long. Then it waits for a response from the server. Because in SNEP header it is declared longer SNEP message than one fragment, the server should response with the Continue response. The SNEP client can go on with sending the rest of the SNEP message. As soon as the entire SNEP message has been transmitted, the SNEP client shall receive the SNEP Success response from the mobile device and the transmitted picture should be immediately displayed on the mobile's screen.

The SNEP client software is deeply described in [13] in "Sample code".

5.5.2.2 Choosing the NDEF message

By default the software sends image of the NXP logo as NDEF message. There are more NDEF messages prepared in dedicated header files (see Table 1). Only a single header can be compiled with the SNEP client application. To choose another content of the NDEF message for transmission just follow the instructions below. In case of URI or text message just skip from step 1 to step 4 (steps 2 and 3 are for selection of an image). Concurrently with instructions there are parts of source code to demonstrate choosing of hardcoded PNG image file for transmission.

1. Open for editing the source file *ndef_message.c* located in folder *src/*.

2. Uncomment the line with content a header file to be transmitted. Let all the other lines commented.

```
/* select the required type of transported data */
//#include <c_tabletxt.h>
#include <c_tablepng.h>
//#include <c_tablepng_NXP_QR.h>
//#include <c_tablejpg_NXP_logo.h>
```

3. In Table 1 in the same row as the chosen file look up two identifiers corresponding to the chosen file.

4. In the array `nmess[]` uncomment the line with the couple of identifiers corresponding to the type of the chosen file. Comment all the other lines.

5. Save the changes and recompile the project.

```
NDEF_messages n_mess[]={ /* type, parameter, string */
//      {NDEF_TYPE_IMAGE,NDEF_IMAGE_JPEG,c_table, sizeof(c_table)},
      {NDEF_TYPE_IMAGE,NDEF_IMAGE_PNG,c_table, sizeof(c_table)},
//      {NDEF_TYPE_IMAGE,NDEF_IMAGE_TIFF,c_table, sizeof(c_table)},
//      {'T', LANG_NO, text1, sizeof(text1)},
//      {'T', LANG_EN, c_table, sizeof(c_table)},
//      {'T', LANG_EN, text1, sizeof(text1)},
```


All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

Table 1. Table of files prepared for NDEF message transmission

Identifiers from the last two columns are necessary for choosing the right line from `n_mess[]`.

Content	Header Name	NDEF message identifier	File type identifier
PNG image	*c_tablepng.h*	`NDEF_TYPE_IMAGE`	`NDEF_IMAGE_PNG`
QR code of NXP	*c_tableQR.h*	`NDEF_TYPE_IMAGE`	`NDEF_IMAGE_PNG`
Image of NXP logo	*c_tablenxp.h*	`NDEF_TYPE_IMAGE`	`NDEF_IMAGE_JPEG`
Long text message	*c_tabletxt.t*	`'T'`	`LANG_EN`

6. Supplementary Notes

6.1 Software architecture

The software of the reference reader is based on the NXP reader library [5]. It intends to be simple, modular, easily readable and quickly portable by all the customers. This philosophy is reflected in its architecture which is divided in 4 layers:

- BAL (Bus Abstraction Layer),
- HAL (Hardware Abstraction Layer)
- PAL (Protocol Abstraction Layer)
- AL (Abstraction Layer)

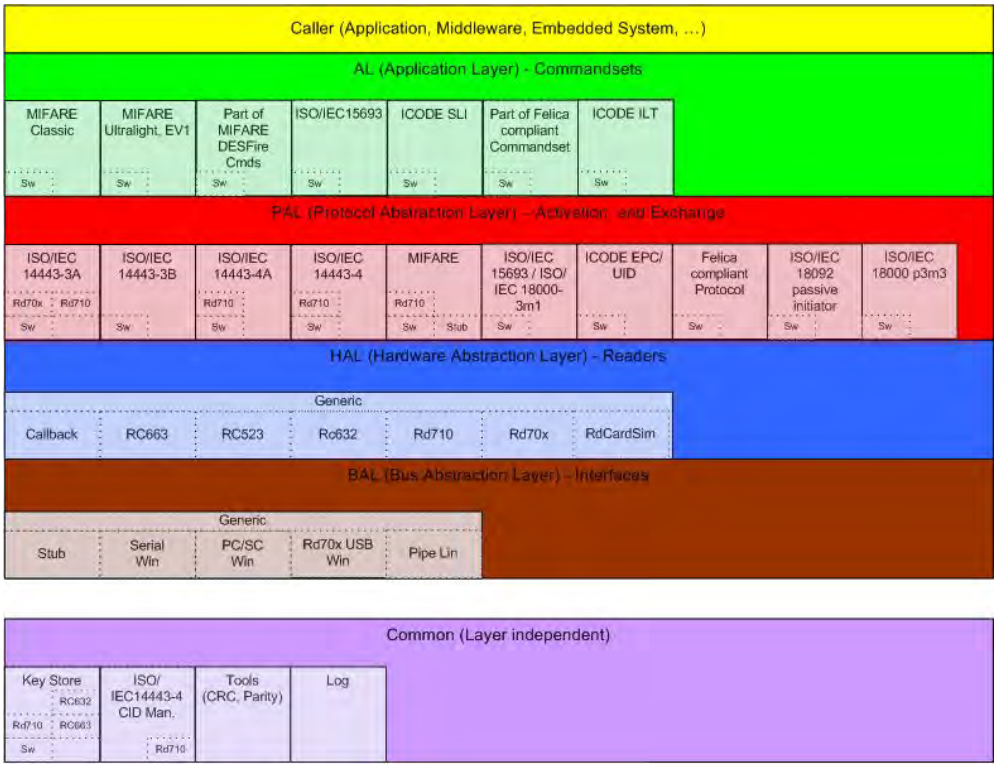

Fig 28. Architecture of the NXP reader library

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

6.1.1 Bus abstraction layer

This layer offers functions to abstract the hardware parts of the LPC1227 microcontroller.

These functions use the specific libraries available for the LPC1227 family microcontroller. Based on these stacks, the communication routines for the relevant physical media I2C/SPI can be easily designed. These drivers are specific for the LPC1227 family and therefore cannot be ported to other microcontrollers.

6.1.2 Hardware abstraction layer

This layer offers functions to abstract the hardware parts of the transceiver RC663.

6.1.3 Protocol abstraction layer

Every PAL function is a low level function realizing a single functionality. It is encapsulated in a module. Every function is independent from the others. The user can easily design his application by doing a drag-and-drop of the relevant module.

The following PAL modules are available in this software package:

- ISO/IEC 14443-3A,
- ISO/IEC 14443-3B,
- ISO/IEC 14443-4A/B,
- MIFARE,
- ISO/IEC15693,
- FeliCa,
- NFC Initiator

6.1.4 Application layer

Lying on the previous software layers, the application layer is on top of the reader software package. It combines elements from the previous three parts into high level functionalities.

For more details on the NXP reader library, the reader is invited to refer to the document RC663 Software Design Guide of the NXPRDLib [6].

6.2 Build configuration

This project comprises 2 build configurations:

- Debug configuration

  This configuration is mainly used when the target board is attached to the PC with the JTAG debugger. It allows the display of debug messages in the console window, which is useful in the early stage of the project.

- Release configuration

  Once the project is debugged and mature, it might be interesting to use the release configuration, to use the hardware stand alone. No debug messages are displayed in the console window.
  Note, that only in Release Configuration one can flash the software onto the Blueboard and start it automatically, once power has been attached to the board.

The build configuration can be selected as follows:

- Click on the project in the project window of the LPCXpresso™ IDE,

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

- Right click of the mouse ➔ Select build configuration,
- Set active Debug build (or Release build).

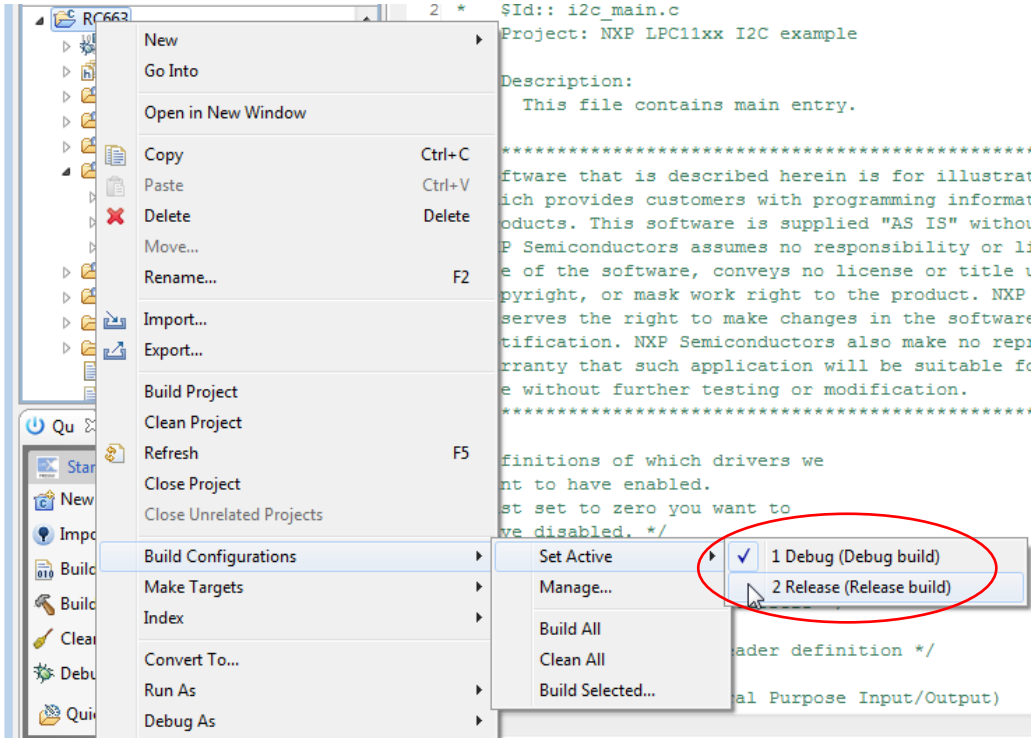

Fig 29. Select the build configuration

6.3 Level of compiler optimization

When the code size at the current compiler level overloads the FLASH size of the target board (128K for the ARM-based microcontroller LPC1227), a higher compiler optimization level can be selected to reduce the code size of the project.

The following steps can be followed to select a level of compiler optimization:

- Click on the project in the project window of the LPCXpresso™ IDE,
- Right click of the mouse ➔ Select properties ➔ Select C/C++ build,
- Select Settings ➔ Optimization,
- Choose the desired level in the combo box.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

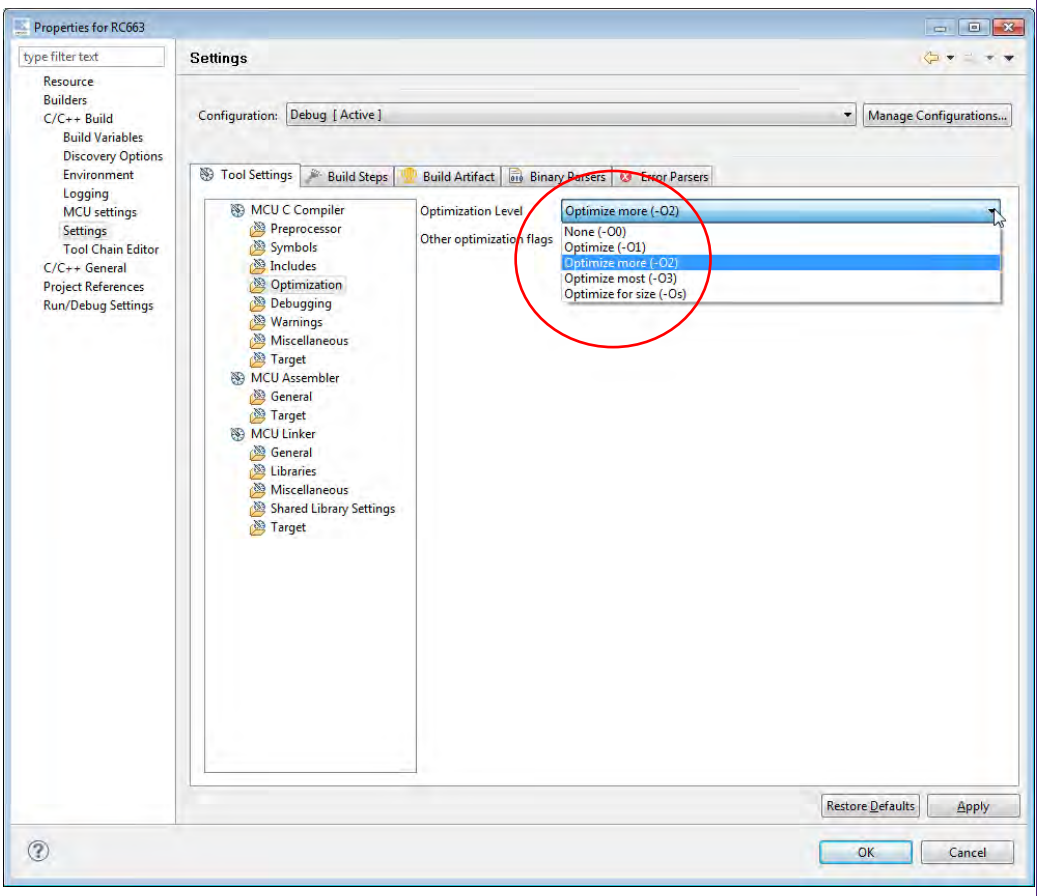

Fig 30. Select the level of optimization

6.3.1 Optimization issues

When optimization is enabled, it will reorder code. What this means is that the code from multiple C lines will be intermingled. In addition, assignments and initializations might be pulled out of loops so they are only executed once. Changes like these will make the code confusing to debug. Some symptoms one might see are breakpoints that only work the first time through, or seeing the debugger's current line indicator fail to advance or even move backwards when clicking step. It is best to always use –O0 for debugging. Since optimization can make such a big difference in code size and performance, it is a good idea to test the project with code optimization on, and plan for a final build that is optimized.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

6.4 Hardware configuration and SPI usage

The hardware configuration – selection of PRH601 is located in the header file hw_config.h. The file is located in …/lpc1227/. The bus interface is configured in header file ph_NxpBuild.h located in …/src/NxpRdLib_PublicRelease/types/.

6.4.1 Hardware configuration

The configuration file hw_config.h provides the following settings:

6.4.1.1 PREV601 board type definition

```
/* ------------------------------------------------------
 * Definition of the used board.
 * If nothing is defined, we assume the CLEV663B board is meant
 * ------------------------------------------------------ */
#define    BOARD_PRH601
#ifdef     BOARD_PRH601
#define    PREV601M
#endif     //BOARD_PRH601
```

The preprocessor directive *BOARD_PRH601* activates the pin configuration according to the hardware layout of the PREV601 board. Each pin setting has its own `#define`. *(SSP_CLK*, *SSP_MOSI*, ...)

Beside the PREV601 board, the software also supports the smaller Microboard (PREV601M). To activate the appropriate hardware configuration, one just needs to uncomment the `#define` **`PREV601M`**. Like the PREV601 board, the Microboard also contains the integrated reader module PRH601HL.

In respect to the software, there are two main differences. As opposed to the PREV601 board, the Microboard uses its internal clock oscillator for the system clock generation. The second difference concerns the Rx threshold register which depends on the antenna pattern. (see chapter 6.5)

6.4.1.2 Reading IC or board type

```
/* ------------------------------------------------------
 * reading IC or board type
 * ------------------------------------------------------ */
#ifndef    BOARD_PRH601
/* TUSA board cannot be used if PRH601 board is set */
//#define TUSA      // definition TUSA card reader instead of BlueBoard
#endif // BOARD_PRH601
```

Because the software projects provided with the PREV601(M) boards are preconfigured for the use with the LPC1227 micro controller, the TUSA board cannot be used with these projects. For more information about this reader module from SILICA, please refer to the quick start guide for the CLEV663B evaluation board [12].

6.4.1.3 Low Power Card Detection mode (LPCD) definition

```
/* ------------------------------------------------------
 * low power mode (LPCD) definition
 * ------------------------------------------------------ */
#ifndef TUSA
/* not allowed for TUSA baord type */
#define LOW_POWER_MODE     /* allow implementation of LPCD mode (low power card
detection)*/
#endif
```


All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

The #define *LOW_POWER_MODE* allows and switches the application into the Low Power Card Detection (LPCD) mode. This setting is not available for TUSA board. The LPCD mode directly works with the reader IC CLRC663 (note: there are some more reader ICs with LPCD functionality)

6.4.1.4 Communication interface definition

The bus interface is configured via #defines in ph_NxpBuild.h in the directory src/NxpRdLib_PublicRelease/types/.

```
/* BAL components */

#define NXPBUILD__PHBAL_REG_STUB               /**< Stub BAL definition */
//#define NXPBUILD__PHBAL_REG_SERIALWIN          /**< SerialWin BAL definition */
//#define NXPBUILD__PHBAL_REG_PCSCWIN            /**< PcscWin BAL definition */
//#define NXPBUILD__PHBAL_REG_RD70XUSBWIN        /**< Rd70X_UsbWin BAL definition */
```

The #define *NXPBUILD__PHBAL_REG_STUB* is prerequisite for enabling any type of bus connection between MCU and reader chip.

```
/* BAL uC components */
#if defined (NXPBUILD__PHBAL_REG_STUB)
    #define NXPBUILD__PHHAL_HW_BUS_SPI               /**< SPI Bus definition */
//    #define NXPBUILD__PHHAL_HW_BUS_I2C             /**< I2C Bus definition */
//    #define NXPBUILD__PHHAL_HW_BUS_UART            /**< UART Bus definition */
    #if defined (NXPBUILD__PHHAL_HW_BUS_SPI) || \
        defined (NXPBUILD__PHHAL_HW_BUS_I2C) || \
        defined (NXPBUILD__PHHAL_HW_BUS_UART)
        #define NXPBUILD__PHHAL_HW_BUS             /**< Generic BUS uC definition */
    #endif
#endif
```

The #defines *NXPBUILD__PHHAL_HW_BUS_SPI* and *NXPBUILD__PHHAL_HW_BUS_I2C* control the communication interface between the microcontroller LPC1227 and reader IC CLRC663. Each define does small adaptations in the files lpc1227/ph_HwBus.c and lpc1227/ph_HwBus.h.

Further interface setting for the PREV601 board are described in chapter 6.4.2.

6.4.1.5 Debug printout definition

```
/* ------------------------------------------------------------
 * debug printouts definition
 * ------------------------------------------------------------ */
    #ifdef LOW_POWER_MODE
    #define DEBUG_PRINT_LPCD        // print debugging for LPCD mode
    #define DEBUG_PRINT_LPCD_2      // register printing during LPCD mode debugging
    #define Q_I_DEBUG_PRINT
    #endif //LOW_POWER_MODE
```

The defines *DEBUG_PRINT_LPCD*, *DEBUG_PRINT_LPCD_2* and *Q_I_DEBUG_PRINT* allows to activate different levels of debug output about the LPCD mode.

6.4.2 Usage of the SPI interface

Because of the hardware design, the PREV601 board can only use the SPI interface for the communication between the microcontroller LPC1227 and the reader IC CLRC663. Nevertheless one may use the I²C interface in a different hardware design with the same integrated reader module.

The SPI selection is described in the chapter 6.4.1.4.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

The defines regarding the SPI interface are as follows:

```
#define SSP_CLK              PIO0_14
#define SSP_MISO             PIO0_16
#define SSP_MOSI             PIO0_17
#define SSP_SSEL             PIO0_15
```

The pin SSP_SSEL is set to the typical GPIO mode and the selection is performed directly from the SPI driver. The remaining three pins are set to SPI mode.

The pins for the selection of the communication interface are as follows:

```
#define PIN_IFSEL0     18   // select pin to define com. interface type for RC663
#define PIN_IFSEL1     19   // select pin to define com. interface type for RC663
```

Based on the datasheet of the reader IC CLRC663 [2] the pins have to be set as follows:

```
/* Select SPI link -> IFSEL0 = 0 & IFSEL1 = 1 */
```

The RESET pin, or better power down pin, of the reader IC CLRC663 is defined as follows:

```
#define PIN_RESET      20   // PDOWN - pin for power down/reset of RC663
```

6.5 LPCD mode configuration

The reader IC CLRC663 is capable to operate in low power card detection mode. This mode is an energy saving mode intended for use cases where the CLRC663 is required to consume minimum current during polling, e. g. for systems powered by battery.

The LPCD feature is independent from the used smartcard/tag communication protocol. Please consult the more detailed application note about the LPCD [11].

The LPCD operates in two phases. The first is the standby mode, controlled by the wake up timer T4. This timer defines the standby duration of the CLRC663 reader IC. The second phase is the detection time. In case that during the detection time an influence of the RF field is recognized, the values of the I and Q channel are stored to the assigned registers and compared with the calculated thresholds.

In case the values of the I and Q channel exceeds the thresholds, the LPCD operation stops and the system continues with the appropriate program flow.
A detuning of the RF field and a resulting stop of the LCPD mode can be evoked by any metallic object.
In the “Polling project” the implementation returns into LPCD mode if communication with a supported card or tag cannot be established.

The software sets the power down time to 10ms and field detection time to 150us.

That means that the LPCD detection works at 100Hz (in accordance to [11]).

6.5.1 Library support

The LPCD operation is also supported by the NxpRdLibrary by the following two functions from the phhalHw_Rc663.c software module:

The function *phhalHw_Rc663_Cmd_LpcdConfig()* takes values of the I and Q field registers.

```
phStatus_t phhalHw_Rc663_Cmd_LpcdConfig(
                                        phhalHw_Rc663_DataParams_t * pDataParams,
                                        uint8_t * pI,
                                        uint8_t * pQ
```


All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

```
                                                    )
```

The function *phhalHw_Rc663_Cmd_LpcdConfig()* performs the LPCD functionality. The input parameters set the I and Q registers, power down time and the detection time. The parameter bMode sets the type of the LPCD (one time operation or cycling operation)

```
phStatus_t phhalHw_Rc663_Cmd_Lpcd(
                                    phhalHw_Rc663_DataParams_t * pDataParams,
                                    uint8_t bMode,
                                    uint8_t bI,
                                    uint8_t bQ,
                                    uint16_t wPowerDownTimeMs,
                                    uint16_t wDetectionTimeUs )
```

6.5.2 LPCD software usage

The reader IC CLRC663 indicates the LPCD card detection via the IRQ1 register (address 07h), bit 5 (LPCD_IRQ). See reference [2]. The additional functionality is an interrupt signal generation at pin 32 (IRQ - interrupt request). But this additional functionality is not yet supported by the NxpRdLibrary.

The following snippet describes the standard LPCD mode setting and detection.

```
    void *pHal;
    /* BFL (Basic Function Library) data parameter storage */
    phbalReg_Stub_DataParams_t balReader;
    phhalHw_Rc663_DataParams_t halReader;

    /* Initialize the Reader HAL (Hardware Abstraction Layer) component */
    status = phhalHw_Rc663_Init(
            &halReader,
            sizeof(phhalHw_Rc663_DataParams_t),
            &balReader,
            0,
            bHalBufferReader,
            sizeof(bHalBufferReader),
            bHalBufferReader,
            sizeof(bHalBufferReader));

    /* Set the generic pointer */
    pHal = &halReader;

#ifdef     LOW_POWER_MODE
    uint8_t bI_Register;
    uint8_t bQ_Register;
#endif // LOW_POWER_MODE

.....
#ifdef     LOW_POWER_MODE
    debug_printf(" ------------------------- ");
    debug_printf(" --- LPCD configuration --- ");

    PH_CHECK_SUCCESS_FCT(status, phhalHw_Rc663_Cmd_LpcdConfig(pHal,
           &bI_Register, &bQ_Register));
```


All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

```
#ifdef DEBUG_PRINT_LPCD
    debug_printf("\n --- register I= ");
    debug_puts_hex(bI_Register);
    debug_printf("\n --- register Q= ");
    debug_puts_hex(bQ_Register);

#endif // DEBUG_PRINT_LPCD
    debug_printf(" --- LPCD detection     --- ");
    PH_CHECK_SUCCESS_FCT(status, phhalHw_Rc663_Cmd_Lpcd(
           pHal,
           PHHAL_HW_RC663_CMD_LPCD_MODE_POWERDOWN,
           bI_Register,
           bQ_Register,
           10,     // power down time
           150));  // operation time
    debug_printf(" --- field activated    ---");
#endif // LOW_POWER_MODE
```

6.5.3 The I and Q Register settings

The values of the I and Q channel are detected and stored in the register map:

- 42h LPCD_I_Result Low-power card detection I channel result register
- 43h LPCD_Q_Result Low-power card detection Q channel result register

At the beginning the values for the empty field, rather for the initial conditions, are stored in these registers.

The detection threshold levels are calculated from the values of the I and Q registers. Registers LPCD_IMin_Reg, LPCD_IMax_Reg; LPCD_QMin_Reg and PCD_QMax_Reg are written to, and stored in the follow registers:

- 3Fh LPCD_QMin Low-power card detection Q channel minimum threshold
- 40h LPCD_QMax Low-power card detection Q channel maximum threshold
- 41h LPCD_IMin Low-power card detection I channel minimum threshold

NOTE: LPCD_Imax is stored at the highest bit of these 3 registers. See [2].

The detection thresholds are calculated in the following area of the NxpRdLibrary:

…\src\NxpRdLib_PublicRelease\comps\phhalHw\src\Rc663\phhalHw_Rc663_Cmd.c Snippet starting at line 180.

```
/* Calculate I/Q min/max. values */
bQMin = bQ - 1;
bQMax = bQ + 1;
bIMin = bI - 1;
bIMax = bI + 1;

/* Set Qmin register */
bRegister = bQMin | (uint8_t)((bIMax & 0x30) << 2);
PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_WriteRegister(pDataParams,
      PHHAL_HW_RC663_REG_LPCD_QMIN, bRegister));

/* Set Qmax register */
bRegister = bQMax | (uint8_t)((bIMax & 0x0C) << 4);
```


All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

```
PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_WriteRegister(pDataParams,
      PHHAL_HW_RC663_REG_LPCD_QMAX, bRegister));

/* Set Imin register */
bRegister = bIMin | (uint8_t)((bIMax & 0x03) << 6);
PH_CHECK_SUCCESS_FCT(statusTmp, phhalHw_WriteRegister(pDataParams,
      PHHAL_HW_RC663_REG_LPCD_IMIN, bRegister));
```

6.5.4 What to do if the PREV601 board does not stay in LPCD mode

If the CLRC663 often leaves the LPCD mode without any metallic object being put into its RF field, a solution may be to enlarge the detection thresholds from +/- 1 to +/-3 or +/-4. The following code shows an example:

File …\src\NxpRdLib_PublicRelease\comps\phhalHw\src\Rc663\phhalHw_Rc663_Cmd.c Snippet starting at line 180.

```
/* Calculate I/Q min/max. values */
bQMin = bQ - 3;
bQMax = bQ + 3;
bIMin = bI - 3;
bIMax = bI + 3;
```


All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

6.6 Removing the initial breakpoint on debug startup

When the debugger starts, it automatically sets a breakpoint at the first statement in the main() function. One can remove this breakpoint as follows:

1. Right click on the project and choose Launch Configurations → Open Current Launch Configuration.

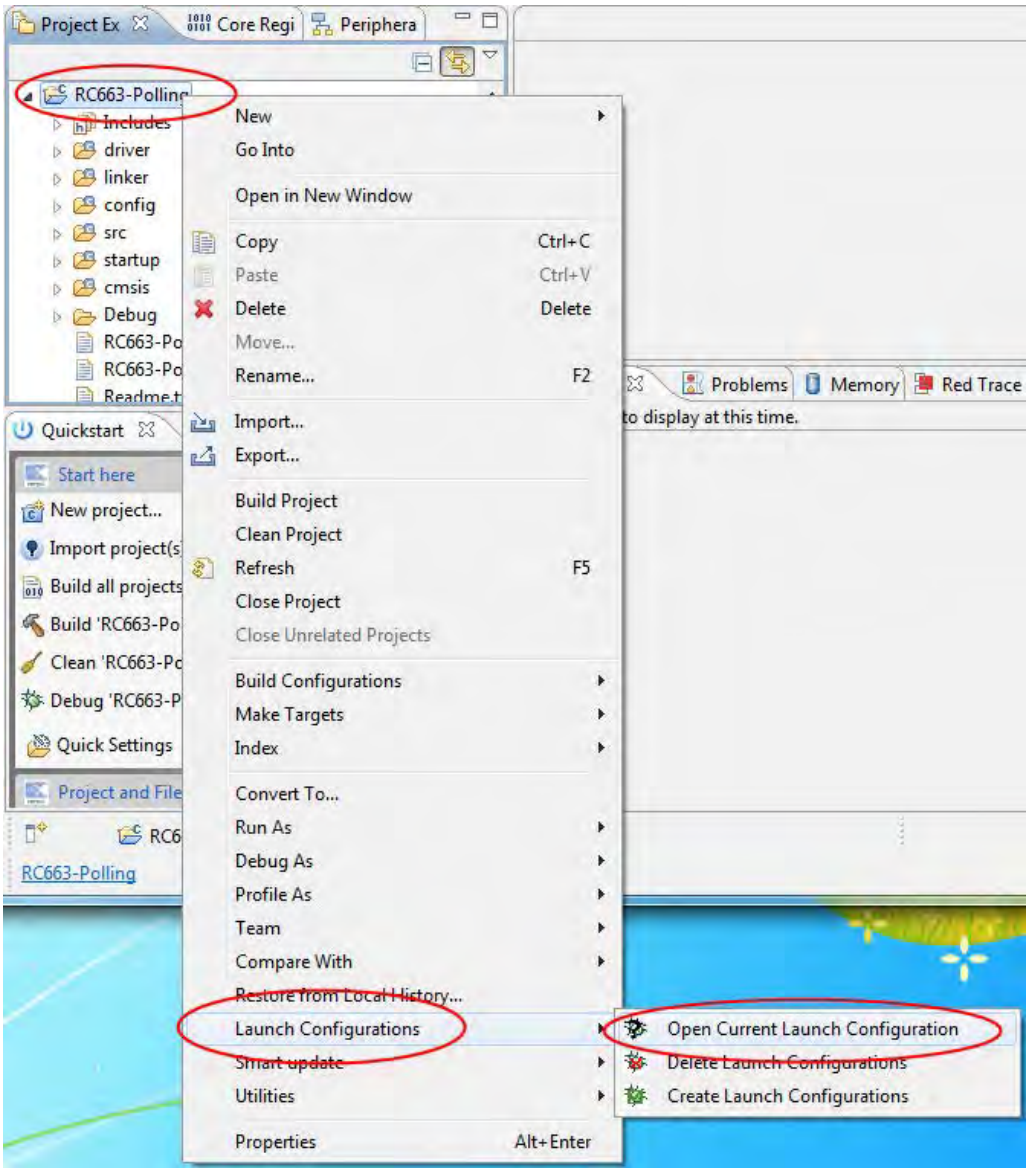

Fig 31. Open Current Launch Configuration

2. Choose the Debug configuration
3. Choose the tab Debugger
4. Uncheck the box near “Stop on startup at:”
5. Click onto Apply and then Close.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

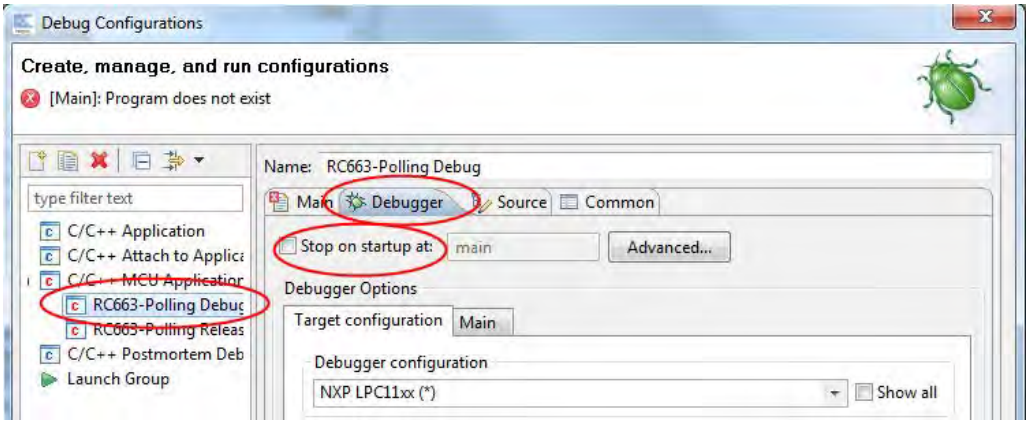

Fig 32. Disable the initial breakpoint

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

7. References

[1] **LPCXpresso website**
www.nxp.com/redirect/lpcware.com/lpcxpresso/downloads/older

[2] **RC663 data sheet**
http://www.nxp.com/documents/data_sheet/CLRC663.pdf

[3] **LPC1227 User Manual**
http://www.nxp.com/documents/user_manual/UM10441.pdf

[4] **Multipoint Connectors we used:**
Grid Dimension: 2.54mm, at least 27 pins
www.nxp.com/redirect/conrad.at/ce/de/product/741119/STIFTLEISTE-1-X-36-POLIG-VERGOL-RM-254
and
www.nxp.com/redirect/conrad.at/ce/de/product/736427/BUCHSENLEISTE-EINREIHIG-36-POLIG-RM254

[5] **Direct link to the NXP Reader Library**
http://www.nxp.com/documents/software/200310.zip

[6] **RC663 and NXP Reader Library**
http://www.nxp.com/documents/application_note/AN11021.pdf

[7] **HITAG HTRC110 data sheet**
http://www.nxp.com/documents/data_sheet/ht037030.pdf

[8] **HTRC11001T**
http://www.nxp.com/products/identification_and_security/nfc_and_reader_ics/hitag_reader_ics/HTRC11001T.html#documentation

[9] **PRH601HL**
http://www.nxp.com/products/identification_and_security/reader_ics/contactless_reader_systems/PRH601HL.html

[10] **PREV601 demo board**
http://www.nxp.com/demoboard/PREV601.html

[11] **AN11145 - low power card detection Quick Start Guide**
http://www.nxp.com/documents/application_note/AN11145.pdf

[12] **Product information page of the CLEV663B evaluation board**
http://www.nxp.com/demoboard/CLEV663B.html

[13] **NXP Reader Library P2P user manual**
http://www.nxp.com/documents/user_manual/UM10721.pdf

[14] **Technical Specification** – Simple NDEF Exchange Protocol, NFCForum-TS-SNEP_1.0, available on www.nxp.com/redirect/nfc-forum.org/specs/spec_license

[15] **Technical Specification** Logical Link Control Protocol, NFCForum-TS-LLCP_1.1, available on www.nxp.com/redirect/nfc-forum.org/specs/spec_license

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

8. Legal information

8.1 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.

8.2 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.

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 accepts 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.

8.3 Licenses

Purchase of NXP ICs with ISO/IEC 14443 type B functionality



This NXP Semiconductors IC is ISO/IEC 14443 Type B software enabled and is licensed under Innovatron's Contactless Card patents license for ISO/IEC 14443 B.

The license includes the right to use the IC in systems and/or end-user equipment.

RATP/Innovatron Technology

8.4 Trademarks

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

MIFARE — is a trademark of NXP B.V.

DESFire — is a trademark of NXP B.V.

MIFARE Ultralight — is a trademark of NXP B.V.

MIFARE Plus — is a trademark of NXP B.V.

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

9. List of figures

Fig 1. Picture of the PREV601 board 3
Fig 2. PREV601M Microboard 4
Fig 3. LPC-Link 5
Fig 4. Multipoint Connectors 5
Fig 5. Connecting the PREV601 board to the LPC Link 6
Fig 6. LPCXpresso with the Multipoint Connectors 6
Fig 7. Enumeration of the LPCXpresso Board in Device Manager Window 7
Fig 8. LPCXpresso installation setup wizard 1 8
Fig 9. LPCXpresso installation setup wizard 2 9
Fig 10. LPCXpresso installation setup wizard 3 9
Fig 11. Windows Security dialog 10
Fig 12. LPCXpresso installation setup wizard 4 10
Fig 13. LPCXpresso IDE 11
Fig 14. Product activation 11
Fig 15. Product activation 12
Fig 16. Product activation 12
Fig 17. Importing a project into the LPCXpresso IDE 13
Fig 18. Importing a project into the LPCXpresso IDE 14
Fig 19. Importing a project into the LPCXpresso IDE 14
Fig 20. Run the project 15
Fig 21. Run the project 16
Fig 22. Run the project 16
Fig 23. Run the project 17
Fig 24. Run the project 17
Fig 25. Stop the project 18
Fig 26. Debug Buttons 18
Fig 27. Importing the project 20
Fig 28. Architecture of the NXP reader library 22
Fig 29. Select the build configuration 24
Fig 30. Select the level of optimization 25
Fig 31. Open Current Launch Configuration 32
Fig 32. Disable the initial breakpoint 33

All information provided in this document is subject to legal disclaimers.

© NXP B.V. 2013. All rights reserved.

10. Contents

1. Introduction 3
2. Hardware overview of the Demo Reader 3
2.1 PREV601 board 3
2.2 LPC-Link LPC3154 4
2.3 Preparation of the hardware 5
3. Connecting the hardware to a computer 7
4. Managing the Demo Reader project with LPCXpresso IDE 7
4.1 Installation of LPCXpresso IDE 8
4.2 Extraction of the demo reader project 12
4.3 Start the project 15
4.3.1 Run the project 15
5. Associated Projects 19
5.1 Communication with MIFARE Ultralight 19
5.2 Communication with MIFARE Classic 19
5.3 Communication with MIFARE DESFire 19
5.4 Polling 19
5.5 Peer to Peer functionality 19
5.5.1 Installation 19
5.5.2 SNEP client project 20
5.5.2.1 What is going on inside? 20
5.5.2.2 Choosing the NDEF message 21
6. Supplementary Notes 22
6.1 Software architecture 22
6.1.1 Bus abstraction layer 23
6.1.2 Hardware abstraction layer 23
6.1.3 Protocol abstraction layer 23
6.1.4 Application layer 23
6.2 Build configuration 23
6.3 Level of compiler optimization 24
6.3.1 Optimization issues 25
6.4 Hardware configuration and SPI usage 26
6.4.1 Hardware configuration 26
6.4.1.1 PREV601 board type definition 26
6.4.1.2 Reading IC or board type 26
6.4.1.3 Low Power Card Detection mode (LPCD) definition 26
6.4.1.4 Communication interface definition 27
6.4.1.5 Debug printout definition 27
6.4.2 Usage of the SPI interface 27
6.5 LPCD mode configuration 28
6.5.1 Library support 28
6.5.2 LPCD software usage 29
6.5.3 The I and Q Register settings 30
6.5.4 What to do if the PREV601 board does not stay in LPCD mode 31
6.6 Removing the initial breakpoint on debug startup 32
7. References 34
8. Legal information 35
8.1 Definitions 35
8.2 Disclaimers 35
8.3 Licenses 35
8.4 Trademarks 35
9. List of figures 36
10. Contents 37

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.

© NXP B.V. 2013. **All rights reserved.**

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

Date of release: 10 November 2013
243915
Document identifier: AN11281