

# AN11211

## Quick Start Up Guide RC663 Blueboard

Rev. 1.6 — 10 November 2013  
226216

Application note  
COMPANY PUBLIC

### Document information

Info	Content
<b>Keywords</b>	RC663, Blueboard, LPCXpresso, MCU, Code Red, eclipse, LPC1114, LPC1115, LPC1227, reader library
<b>Abstract</b>	This application note is related to the installation procedures of the RC663 Blueboard. It describes the actions to be done to become acquainted with the demo reader.



**Revision history**

Rev	Date	Description
1.6	20131110	Added a note about the LPCXpresso IDE version in chapter 4
1.5	20130613	Added description about the P2P Snep Client
1.4	20130221	Added description of the P2P project. Added description of the I <sup>2</sup> C configuration for the Blueboard version 3.0 and above. Added information about the use of the projects in conjunction with the LPC1227 MCU. Added information about the documentation of the NXP Reader Library. Added information about the exemplary project of code size optimization of the NXP Reader Library.
1.3	20120913	Small corrections of the TUSA description
1.2	20120822	Insertion of the description for the 3rd party "Tusa" Board
1.1	20120704	Small text corrections
1.0	20120604	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](mailto:salesaddresses@nxp.com)

## 1. Introduction

This application note gives a detailed overview of the hardware for working with the RC663 contactless reader - we use the LPCXpresso LPC 1114/301 and the Blueboard (**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**).

Detailed information on the RC663 in connection with the NXP Reader Library can be obtained at [\[1\]](#). But with the information provided in this document one will be fine to get started.

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**
- Exemplary Peer to Peer functionality → **Chapter 5.5**

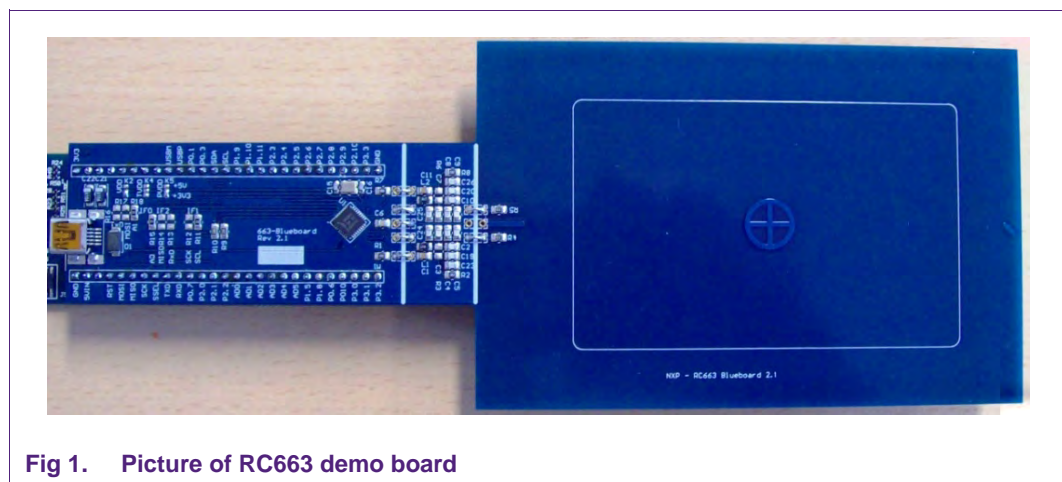
## 2. Hardware overview of the Demo Reader

The demo reader is made up of 2 separate boards:

- A RC663 demo board (called Blueboard) provided by NXP. This board has connectors which are designed to exactly fit the ones of the companion, the LPCXpresso LPC 1114/301 development board.
- A commercial LPCXpresso LPC 1114/301 (**12NC**: 935290886598, **Type**: OM11049+598) development board which can be provided by NXP or bought directly on the market. See [\[2\]](#).

Once the two boards are joined via the connectors, the demo reader is ready for use.

### 2.1 RC663 demo board (Blueboard)



The RC663 demo board embeds the contactless communication transceiver IC RC663 with all its elements needed for transmission: EMC filter, matching network and the

antenna. The RC663 supports different kind of contactless communication methods and protocols at 13.56 MHz:

- Reader/Writer mode supporting ISO/IEC14443A/MIFARE,
- Reader/Writer mode supporting ISO/IEC14443B,
- Reader/Writer mode supporting FeliCa scheme,
- Passive initiator mode according to NFCIP-1
- Reader/writer supporting ISO/IEC 15693,
- Refer to the data sheets of this IC [3] for more details

Thanks to the relevant solder bridge, the host link of the RC663 demo board can be configured for:

- Serial UART,
- I2C,
- SPI

The voltage of the power supply VDD, the pad supply PDD and the transmitter supply can also be configured independently to 3,3 V or 5 V using the solder bridges.

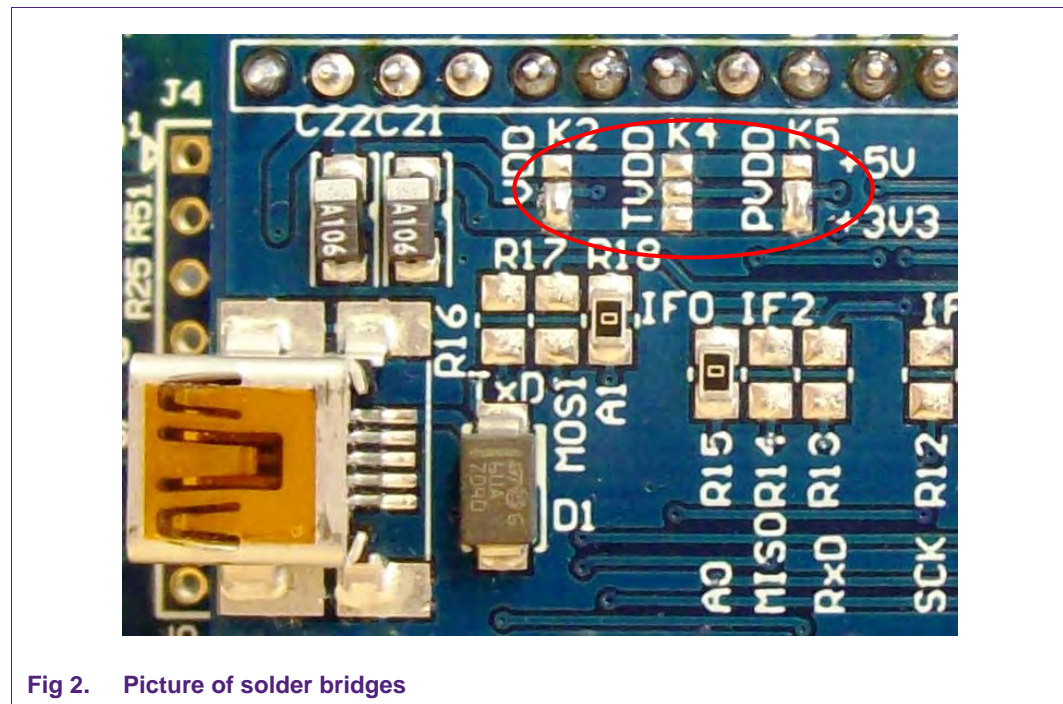


Fig 2. Picture of solder bridges

### 2.1.1 Derivates of the RC663 demo board (Blueboard)

To meet the interests of the market, we also offer Blueboards with a slightly modified RC663 on it.

The four versions are as follows:

- RC663 Blueboard

This Blueboard with the RC663 offers the full functionality. It supports the modes described above. In the following text we always relate to the RC663 Blueboard. If not, this will be indicated by a note.

- SLRC610 Blueboard

This Blueboard with the RC610 works with ISO/IEC 15693 tags only. To use this Blueboard, one will have to manually change one parameter in the NXP Reader Library.

In the folder `../src/NxpRdLib_PublicRelease/intfs/` open the file “`phhalHw_Rc663_Reg.h`” and scroll to the line

```
#define PHHAL_HW_RC663_CMD_LOADPROTOCOL 0x0DU
```

It should be around line 476.

Change the value from `0x0DU` to `0x03U`.

Now, one should be able to use the SLRC610 Blueboard.

- MFRC630 Blueboard

This Blueboard with the MFRC630 works with ISO/IEC 14443A cards only. No modification in the NXP Reader Library is needed.

- MFRC631 Blueboard

This Blueboard with the MFRC631 works with ISO/IEC 14443A and ISO/IEC 14443B cards only. No modification in the NXP Reader Library is needed.

If one has one of the limited Blueboards, one will have code in some of the example projects that will not work with the very board. There is no need to adapt the code for the derivate (except for the one described above). If one for example executes code for ICODE tags and has a MFRC631 Blueboard, this code will have no effect at all. It will behave as if there was no tag in the field.

## 2.2 CE certification of the Blueboard

The current version of the Blueboard (v.3.0) is CE (European Conformity) compliant.

## 2.3 LPCXpresso LPC1114 development board

To work with the provided projects, one will also need an LPCXpresso LPC development board. Such a board is **not included** in the Blueboard hardware package.

The LPC1114 development board integrates an NXP ARM Cortex-M0 microcontroller LPC1114 with 32 Kbytes of Flash memory and 4 Kbytes of RAM. It integrates a lot of hardware parts:

- 1 Serial UART interface,
- 1 SPI controller,
- 1 I2C controller,
- Serial Wire test/debug interface,
- For detailed information, see LPC11xx User Manual [\[4\]](#)

The LPCXpresso board contains a JTAG/ SWD debugger called the “LPC-Link” and a target MCU. LPC-Link is equipped with a 10-pin JTAG header and it seamlessly connects to the target via USB (the USB interface and other debug features are provided by NXP’s ARM9 based LPC3154 MCU).



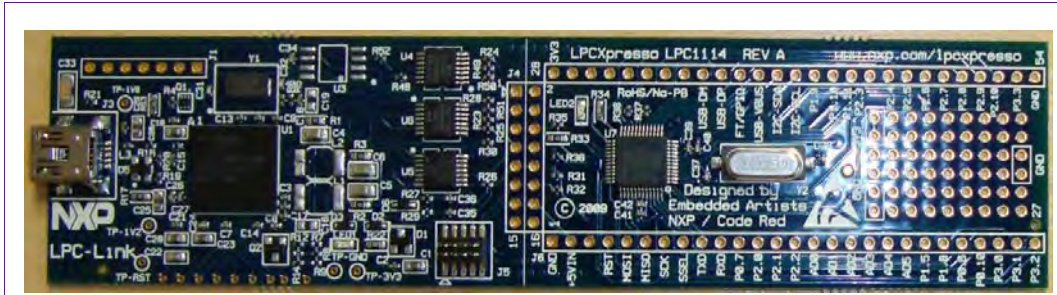


Fig 3. Picture of LPCXpresso LPC1114 development board

### 2.4 Alternative to the LPCXpresso LPC1114

With the provided code one can use the LPCXpresso **LPC1115** (**12NC:** 935297664598, **Type:** OM13035+598) instead of the LPCXpresso LPC1114 without the need of any adaptation in the code. Its advantage is the larger flash memory of 64KB instead of 32KB. That makes the debugging easier because one doesn't need to have the optimization turned on to make the code fit into the flash. For instructions on how to port the project to work with the LPC1115 see the description in chapter 7.7.

As an alternative, one can also use the LPCXpresso **LPC1227** (**12NC:** 935294603598, **Type:** OM13008+598) but needs to use the ported projects that are provided on the product web site of the CLEV663B. For more information please see chapter 7.8

### 2.5 Preparation of the hardware

The first step after unpacking the Blue Board and the LPCXpresso is soldering the connectors onto the boards in order to join them later. In our example we use a multipoint connector as one can see on the pictures below.

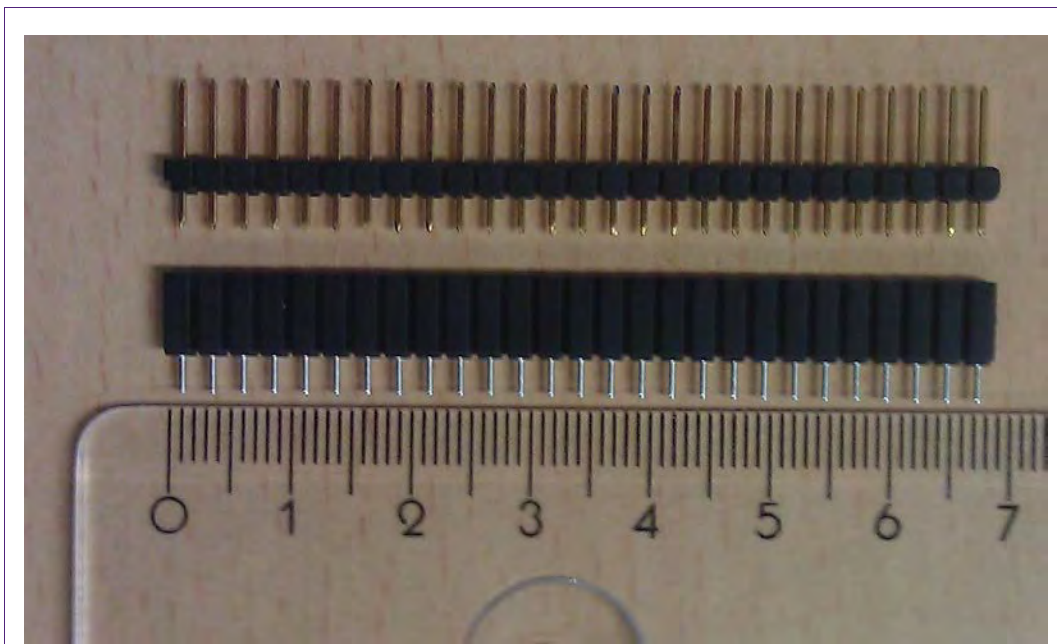


Fig 4. Multipoint Connectors we used

One may buy these connectors at any electronic store. Here are some examples [\[5\]](#).

After soldering the connectors connect the boards as shown on the following figures.

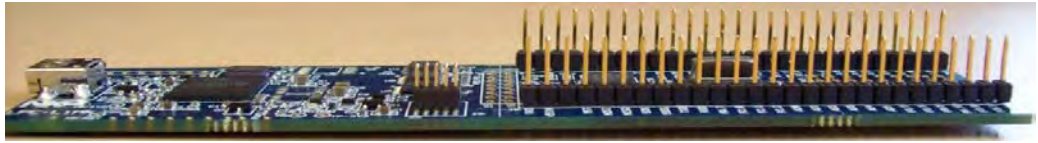


Fig 5. LPCXpresso with the Multipoint Connectors

Now the hardware is ready for use. Please connect the LPCXpresso board to the Blueboard.

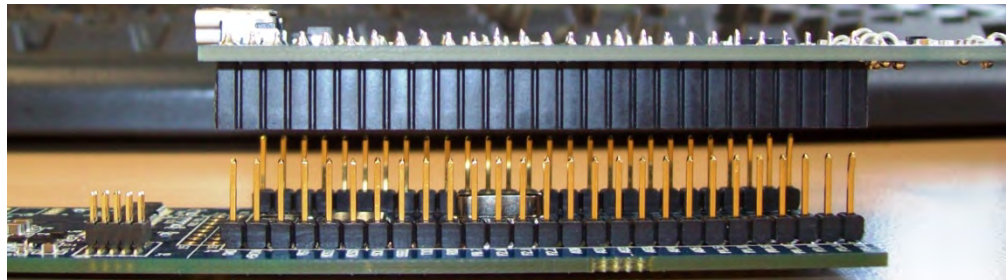


Fig 6. Connect the two boards

Be informed that there is the possibility of arranging the boards vice versa. The pictures below will illustrate how that is meant.

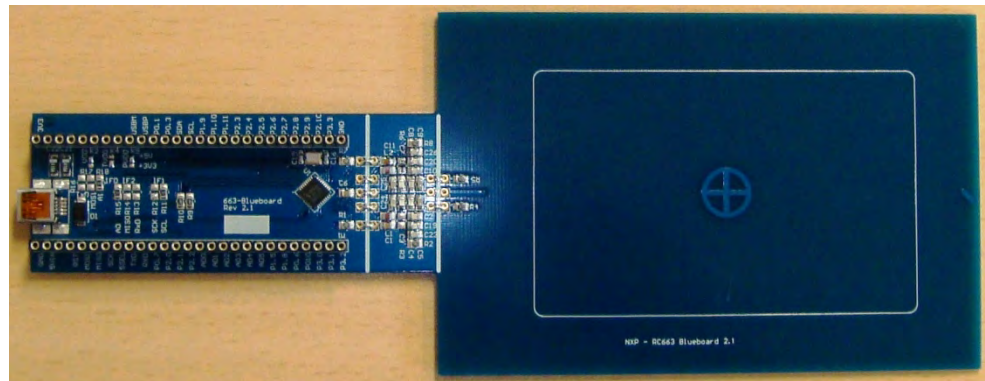


Fig 7. Picture of RC663 demo board with the connectors joined alternatively

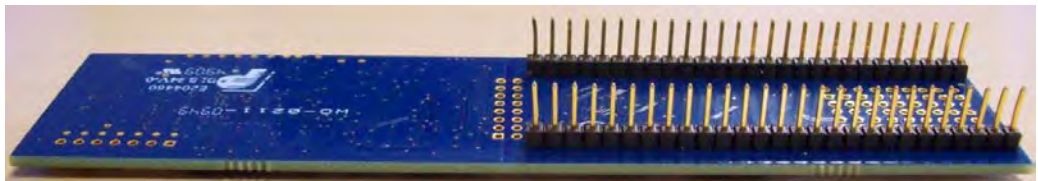


Fig 8. LPCXpresso with the Multipoint Connectors used in the alternative way

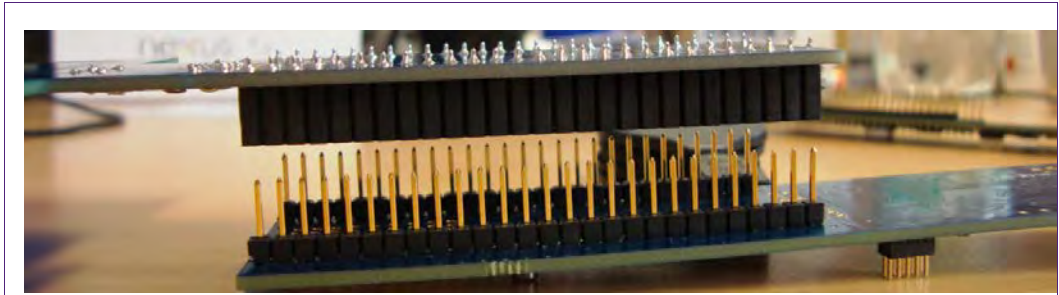


Fig 9. Connect the two boards the alternative way

### 2.6 Interesting points of measurement

Although the Blueboard is not designed to do extended measurements, there are some points of measurement one might be interested in.

To give some examples, a few of these points of measurement will be described in the following.

#### 2.6.1 RXP - receiver input pin for the received RF signal

RC663 – Pin 12

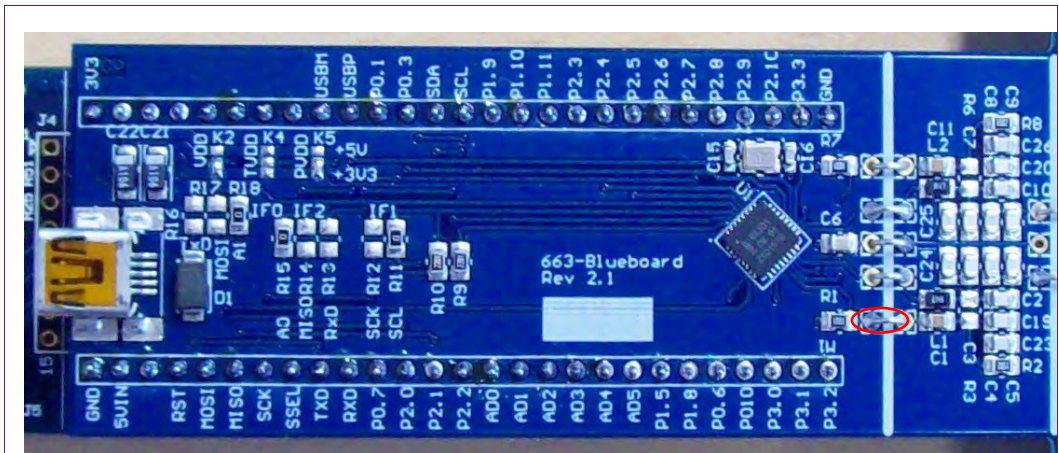


Fig 10. Pin RXP

#### 2.6.2 RXN - receiver input pin for the received RF signal

RC663 – Pin 13



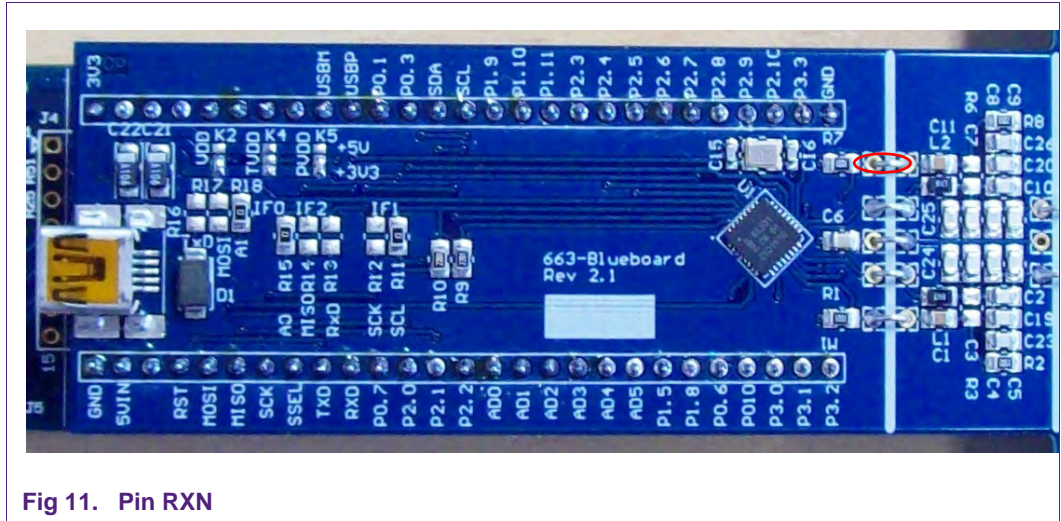


Fig 11. Pin RXN

### 2.6.3 TVDD - transmitter voltage supply

RC663 – Pin 18

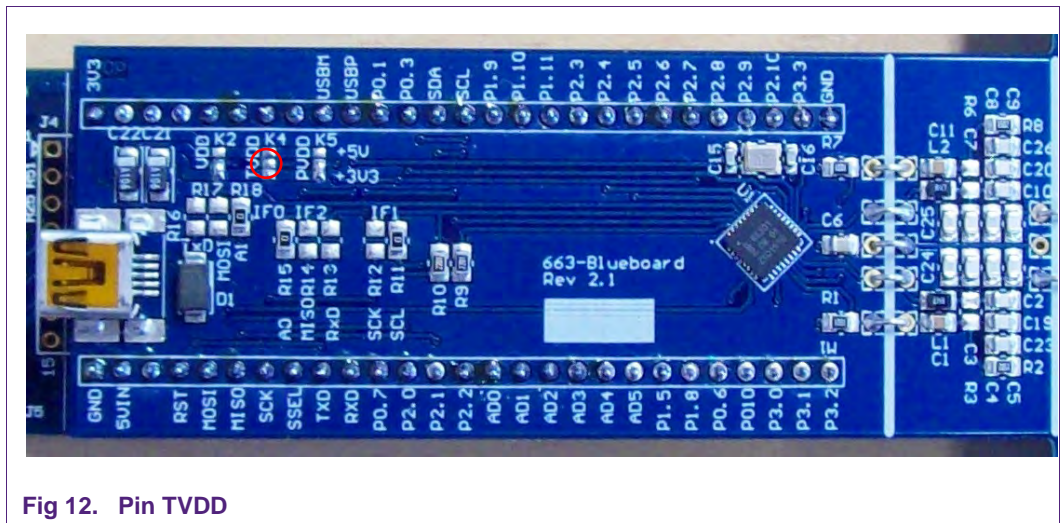


Fig 12. Pin TVDD

## 2.7 Preparing the Blueboard for the use with SPI or I<sup>2</sup>C

The Blueboard is generally delivered in I<sup>2</sup>C configuration, for this reason one only needs to change the configuration if the use of the board in SPI configuration is desired.

From Blueboard version 3.0 on the layout slightly changed. So we provide two different descriptions for changing the interface from I<sup>2</sup>C to SPI.

### 2.7.1 Blueboard version 2.1 and below

There are exactly six solder bridges to change.

1. Open the bridge at R18
2. Open the bridge at R15
3. Open the bridge at R11
4. Close the bridge at R17
5. Close the bridge at R14

- Close the bridge at R12

See the picture of the Blueboard in SPI configuration below.

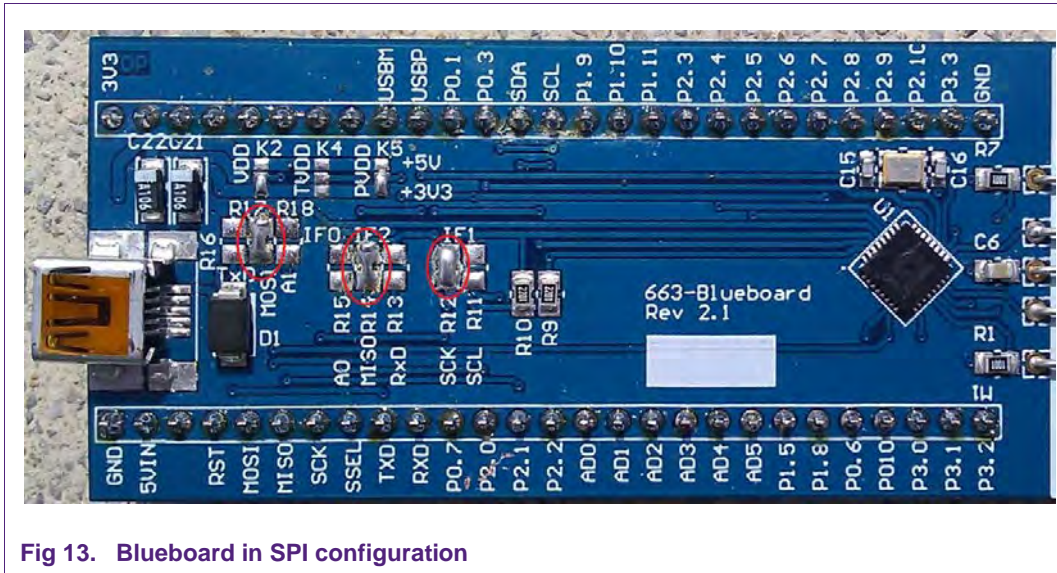


Fig 13. Blueboard in SPI configuration

To use the Blueboard in SPI configuration with the provided software projects, one has to execute two small adaptations in the code, which are described in section 7.5.

### 2.7.2 Blueboard version 3.0 and above

These boards are delivered in SPI configuration. To change that to I<sup>2</sup>C, one needs to change six solder bridges.

- Open the bridge at R15
- Open the bridge at R21
- Open the bridge at R26
- Close the bridge at R16
- Close the bridge at R20
- Close the bridge at R29

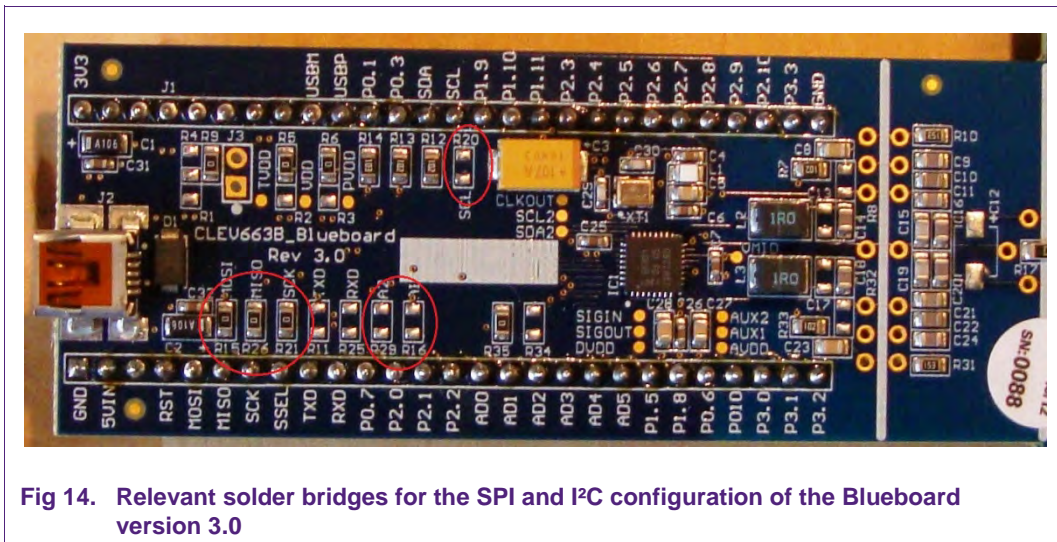


Fig 14. Relevant solder bridges for the SPI and I<sup>2</sup>C configuration of the Blueboard version 3.0



### 3. Installation of the LPCXpresso Board

The guidelines to install the reader are as follows:

- Connect the LPCXpresso Board to a real USB2.0 port of the PC (for speed reasons) using the mini-USB connector. The PC detects and installs the Board automatically.
- Once the Board is 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.

Please be sure to always connect both USB ports to the computer. If the USB port of the Blueboard is not connected to a USB port, it won't work because of the missing power.

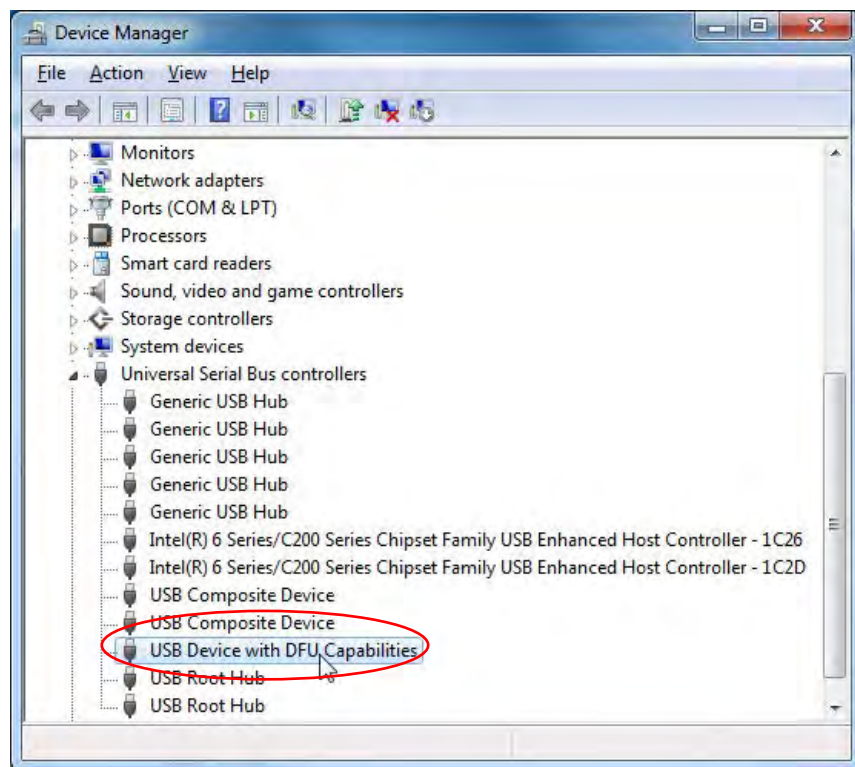


Fig 15. Enumeration of the LPCXpresso Board in Device Manager Window

### 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.

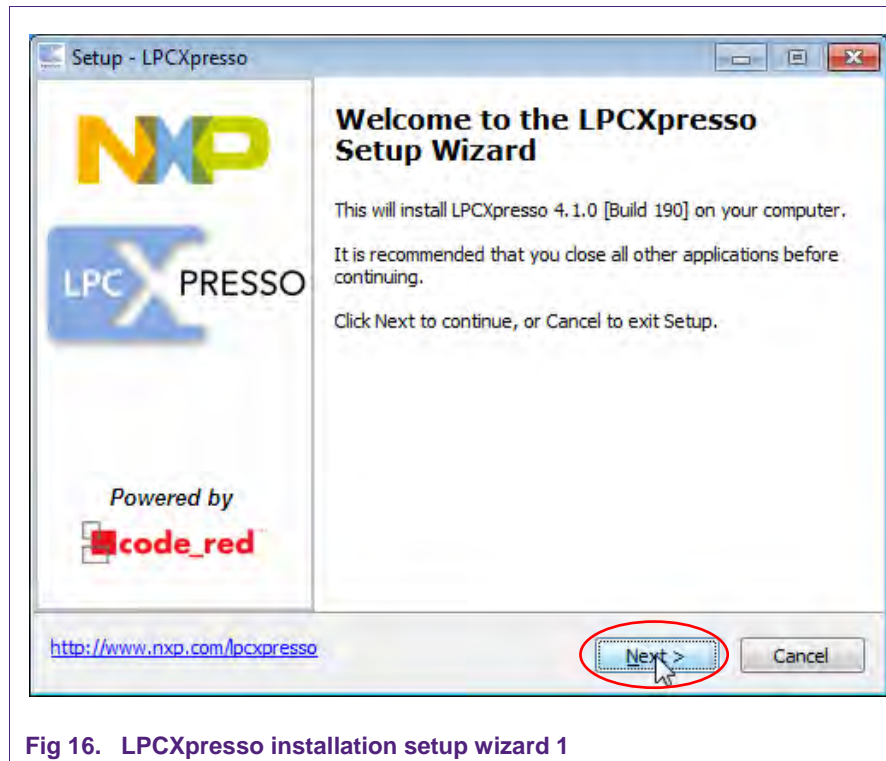
LPCXpresso™ is a new, low-cost development platform available at NXP. It supports NXP's ARM-based LPC microcontrollers. The platform is comprised of a simplified Eclipse-based IDE and low-cost target boards which include an attached JTAG debugger.

**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.**

This tool can freely be downloaded from the LPCXpresso website [2]. 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.



Read the license agreement, then click "Next".



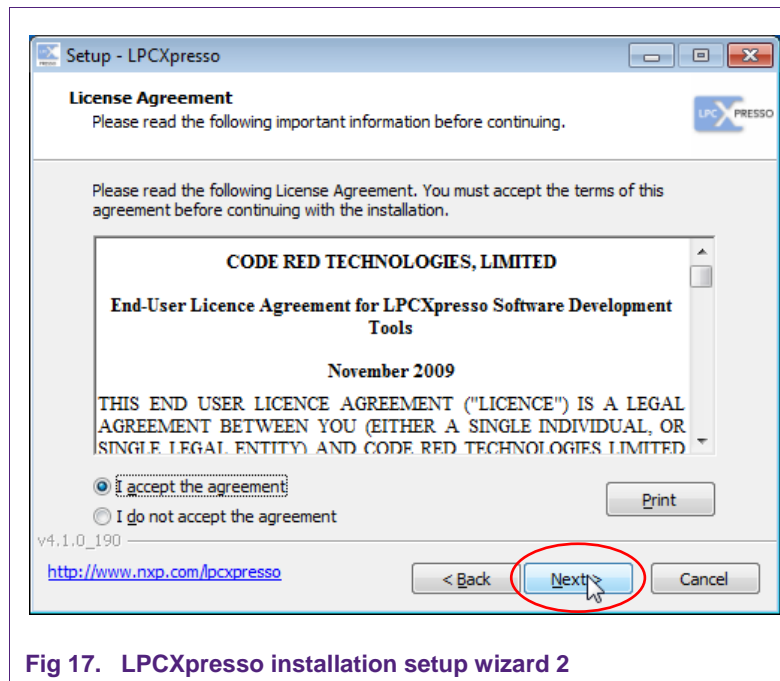


Fig 17. LPCXpresso installation setup wizard 2

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.

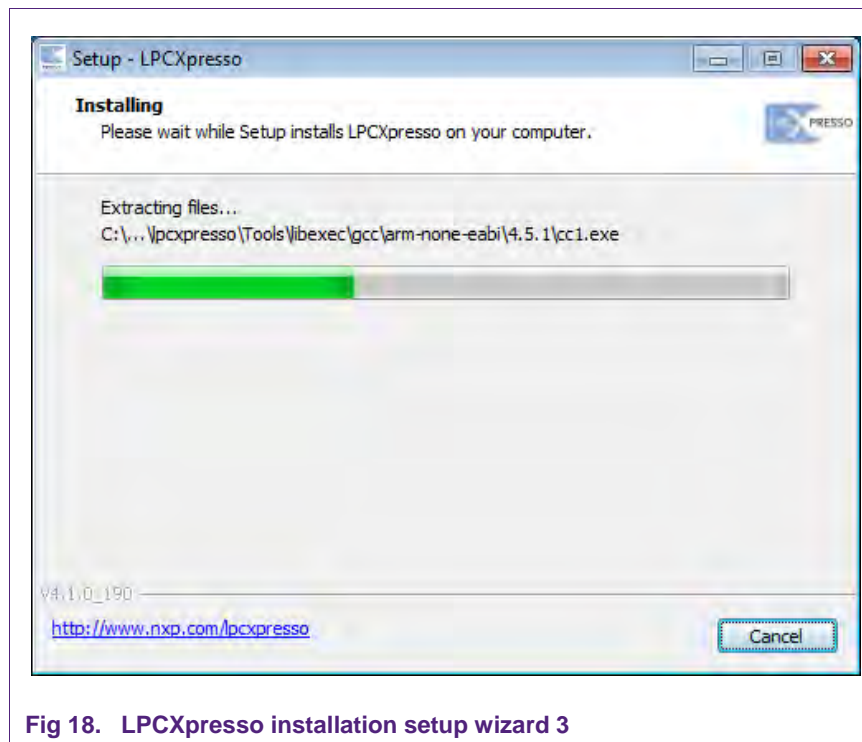


Fig 18. LPCXpresso installation setup wizard 3

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

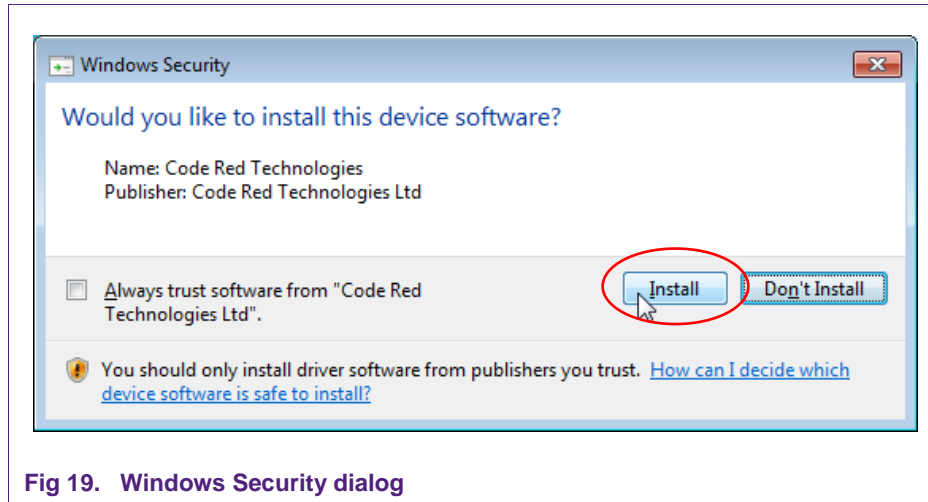


Fig 19. Windows Security dialog

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



Fig 20. LPCXpresso installation setup wizard 4

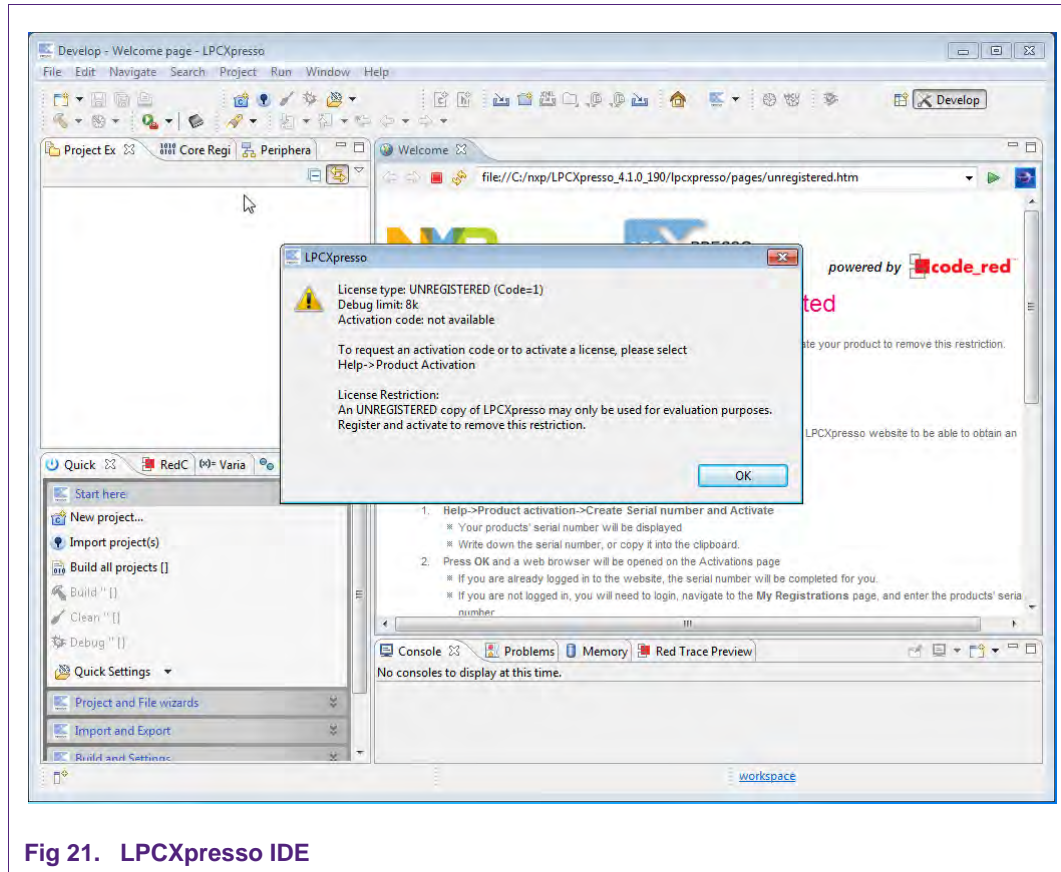


Fig 21. LPCXpresso IDE

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...

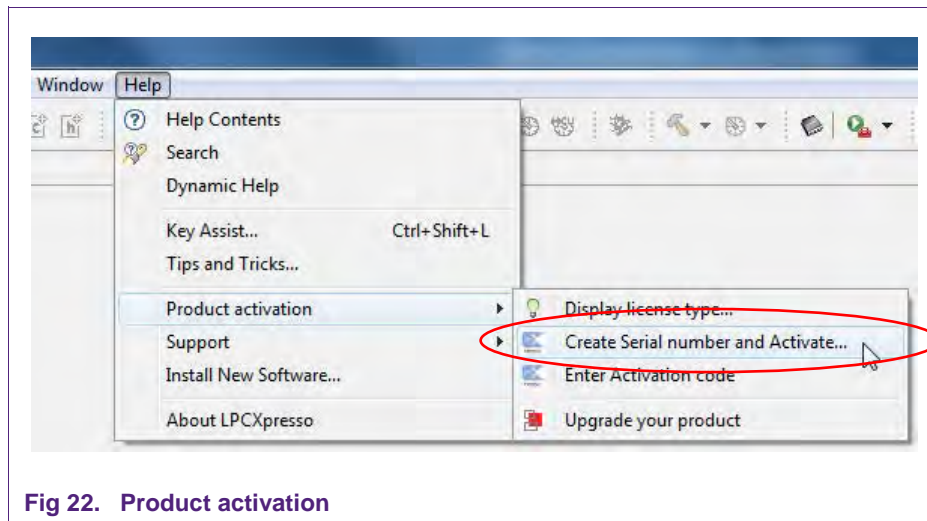


Fig 22. Product activation

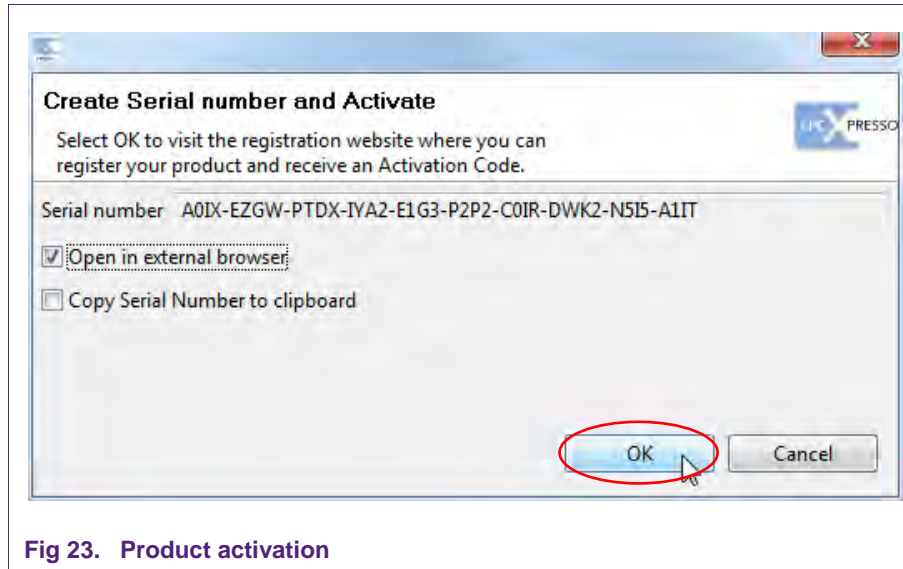


Fig 23. Product activation

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.

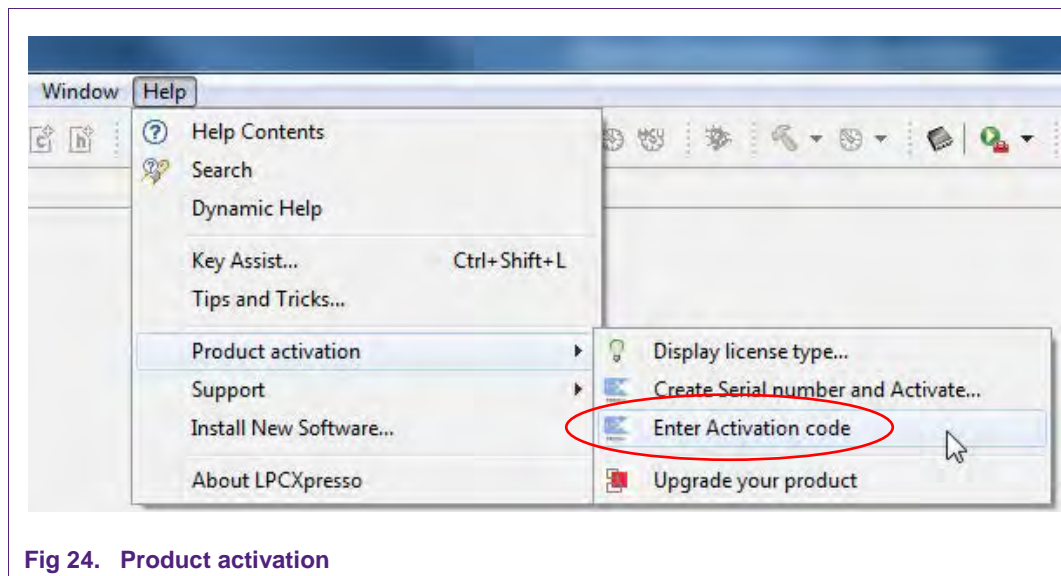


Fig 24. Product activation

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.

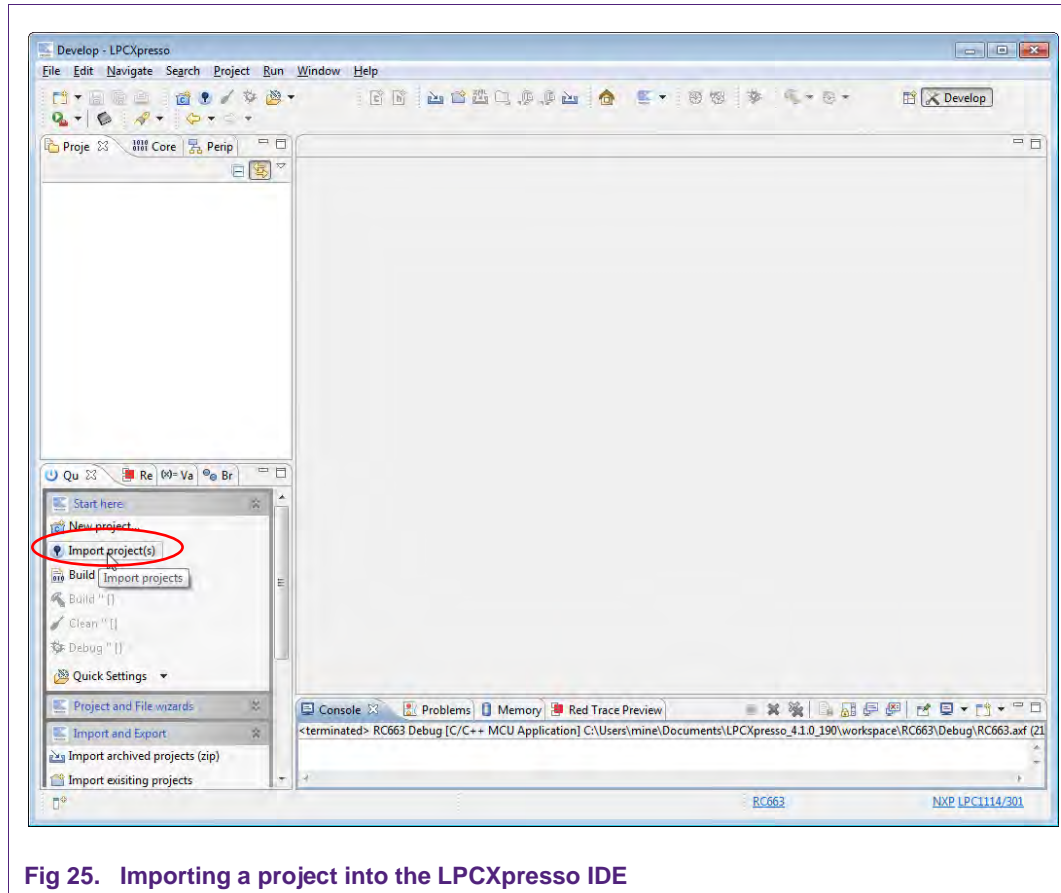
## 4.2 Extraction of the demo reader project

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.



- The software package is ready for use.



**Fig 25. Importing a project into the LPCXpresso IDE**

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

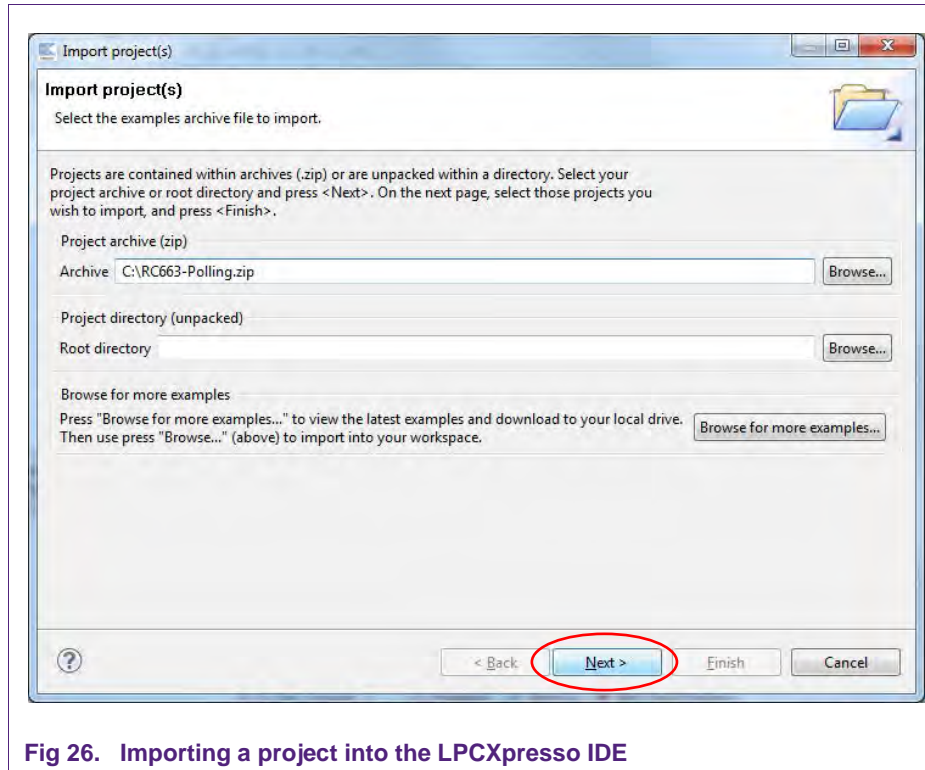


Fig 26. Importing a project into the LPCXpresso IDE

Browse the desired project and click “Next”.

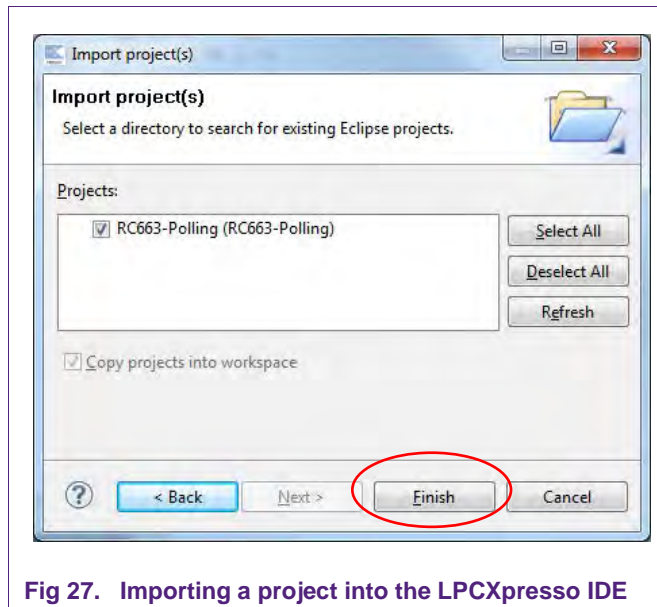


Fig 27. Importing a project into the LPCXpresso IDE

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 Blueboard with the RC663 needs to be connected to the computer. Wait until the adequate drivers have been installed.

### 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 LPC1114/301 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.

**Remember:** If one is using the SLRC610 Blueboard, the code needs to get edited as described in 2.1.1 before one can run the code.

#### 4.3.1 Run the project

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

Although the project names include the string LPC1115, they are preconfigured for the use with the LPCXpresso LPC1114.

Please ensure that the used LPCXpresso LPC1114 board has the label LPC1114/301 on its processor. If it has the label LPC1114/302, you need to change the MCU type in the LPCXpresso IDE setting. Please refer to chapter 7.7 for the steps.

**If one is using the projects with the Silica Tusa Board instead of the RC663 Blueboard, please find the needed modifications in the code at section 0.**

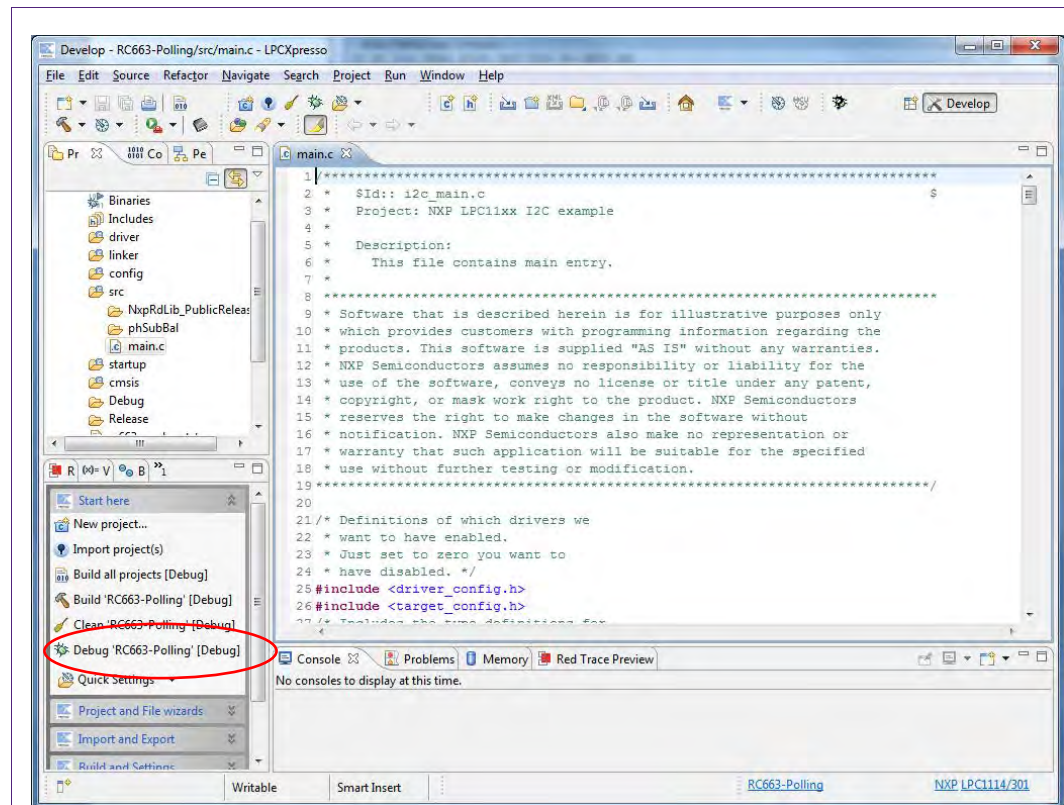


Fig 28. Run the project

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

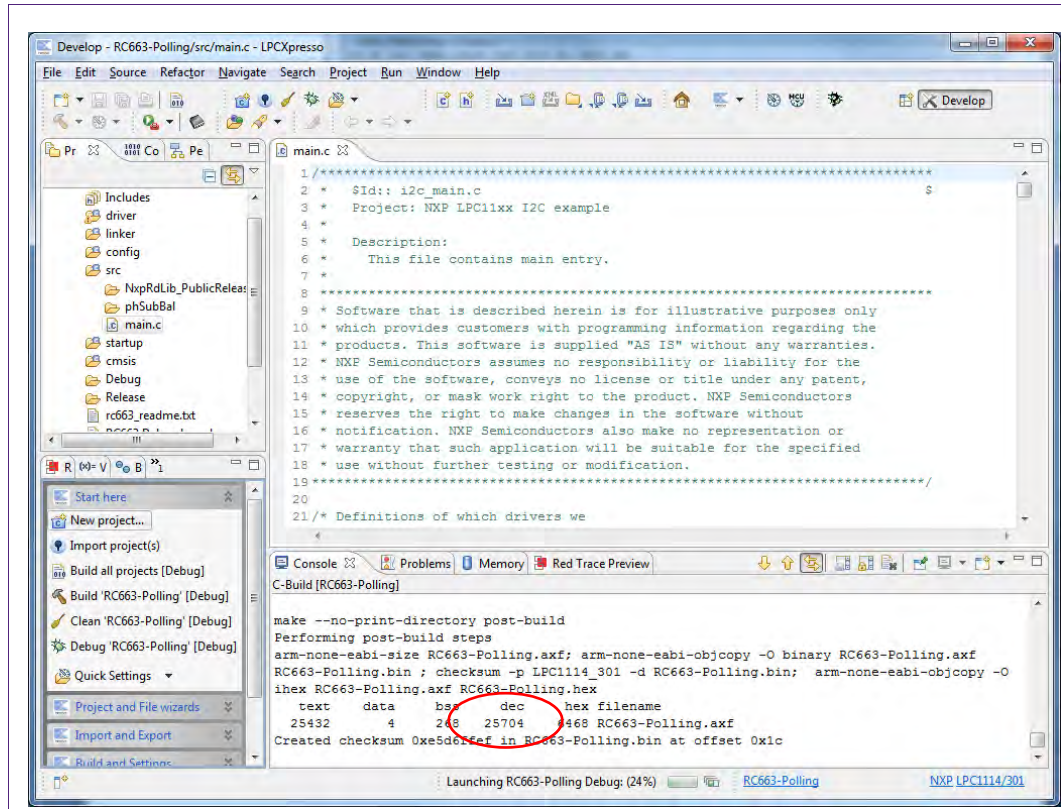


Fig 29. Run the project

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

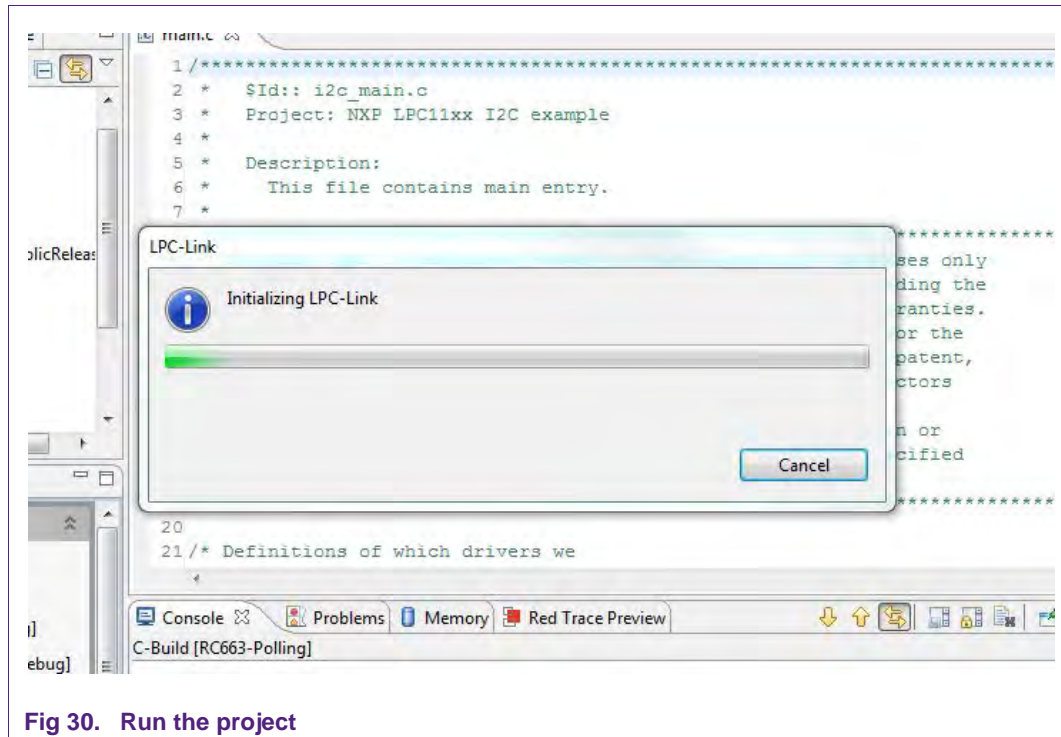
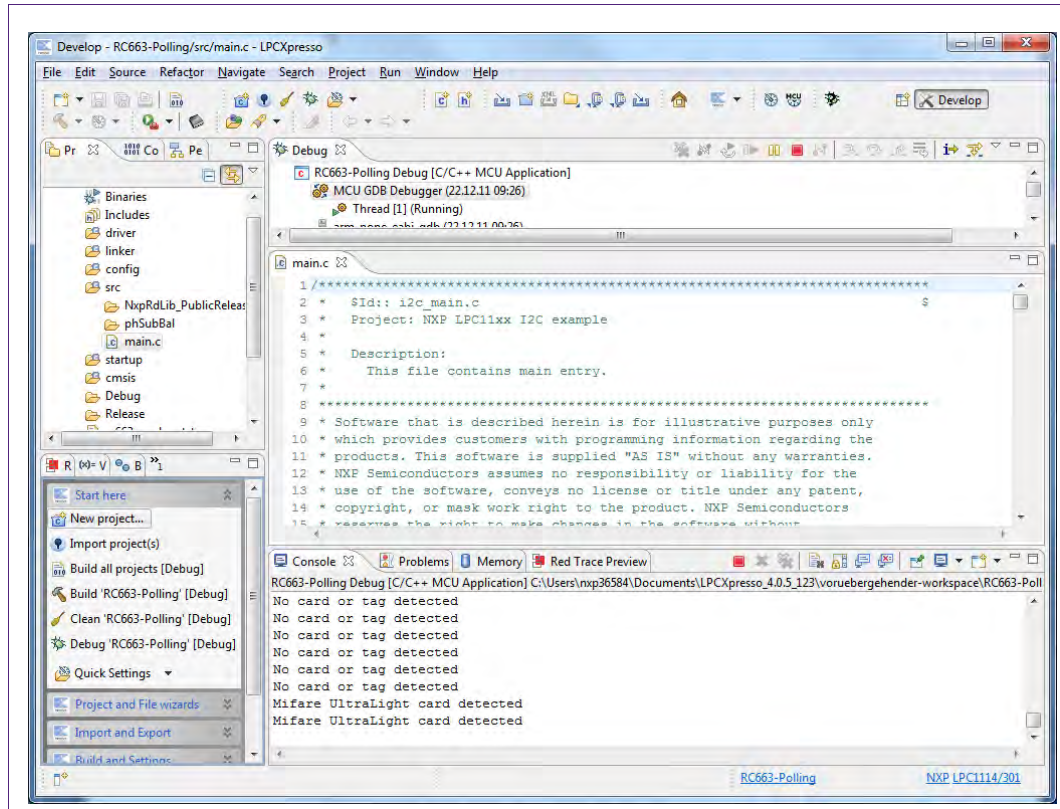


Fig 30. Run the project

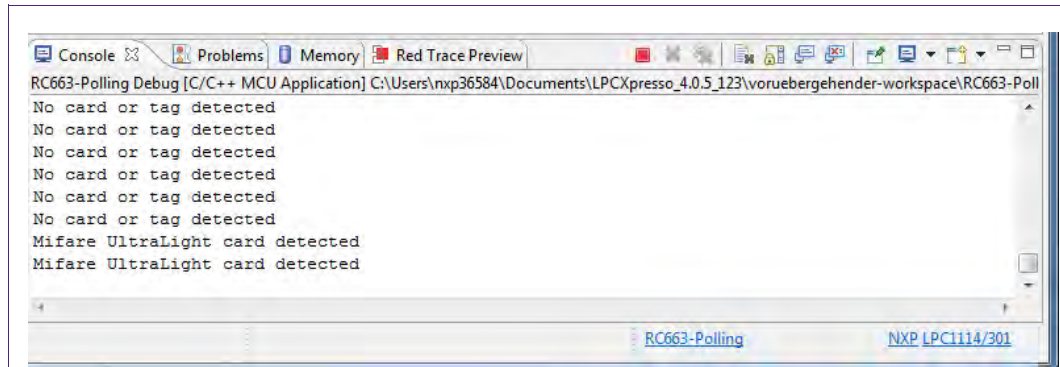
The initialization of the LPC-Link can take a few seconds.





**Fig 31. Run the project**

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



**Fig 32. Run the project**

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

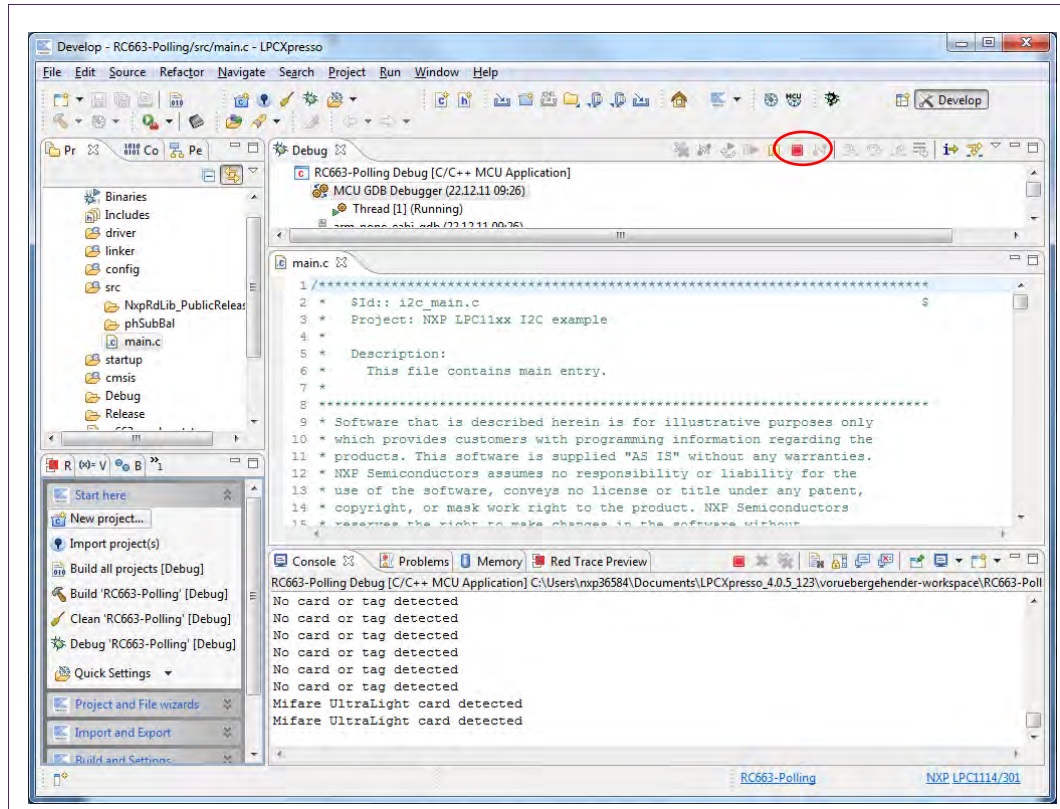


Fig 33. Stop the project

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 34. Debug Buttons

## 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.

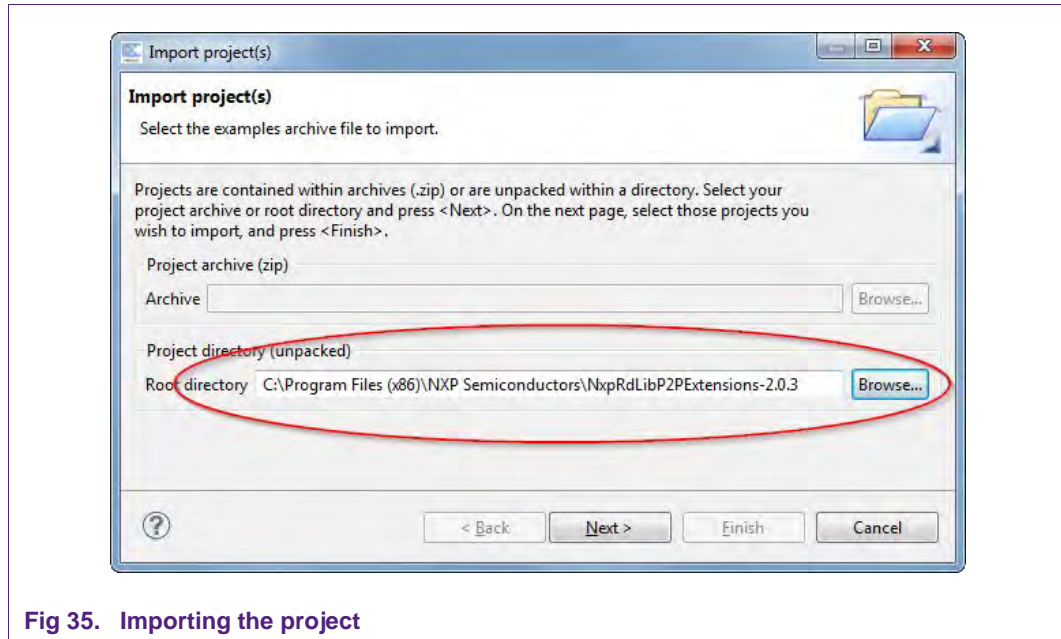


Fig 35. Importing the project

After the import there are two new projects in the workspace - one for the use with the RC663 Blueboard 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 LPC1227 with RC663 Blueboard attached behaves like SNEP client with performing the Put request functionality only [10]. 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 configuration in section 7.8.
2. Hold your NFC device (smartphone) approximately 5 centimeters above antenna of the PNEV512B Blueboard.
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 the RC663 Blueboard’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



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 [11].
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 [9] in chapter “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)},
```

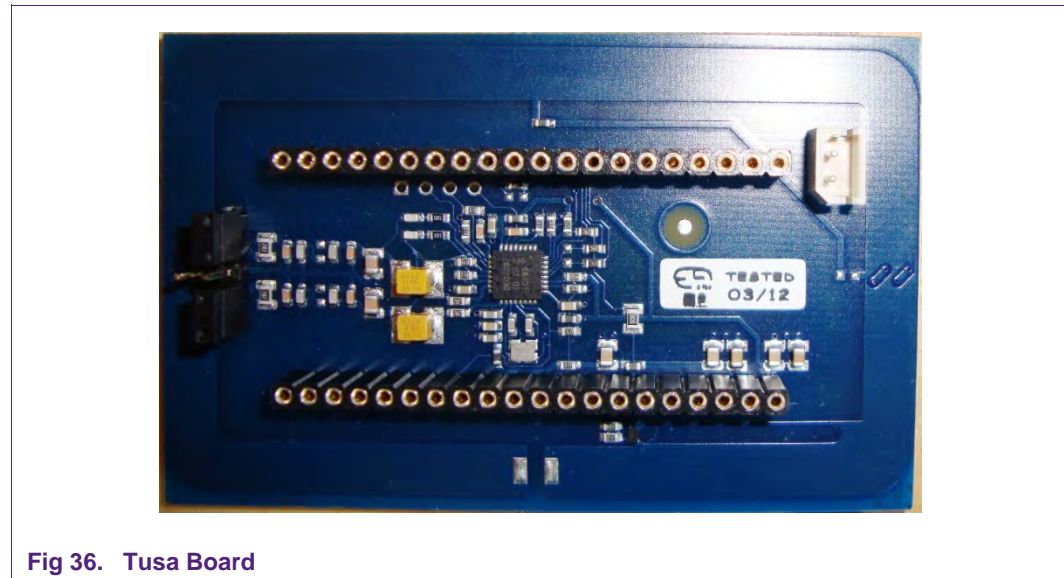
**Table 1. Table of files prepared for NDEF message transmission**

*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	<code>c_tablepng.h</code>	NDEF_TYPE_IMAGE	NDEF_IMAGE_PNG
QR code of NXP	<code>c_tableQR.h</code>	NDEF_TYPE_IMAGE	NDEF_IMAGE_PNG
Image of NXP logo	<code>c_tablenxp.h</code>	NDEF_TYPE_IMAGE	NDEF_IMAGE_JPEG
Long text message	<code>c_tabletxt.t</code>	'T'	LANG_EN

## 6. Other supported hardware by the projects

It's also possible to use the provided projects with the Tusa Board (Manufacturer: Silica). This board is a 3d party alternative to the RC663 Blueboard. It also uses the RC663 reader IC.



**Fig 36. Tusa Board**

To get this board work with the LPC1114 or LPC1115 controller boards you will have to do a small hardware modification. Because the Tusa Board gets its power from the controller board, we need to solder a wire from the Capacitor C28 to the Pin 29 on the LPCXpresso Board. See figure:

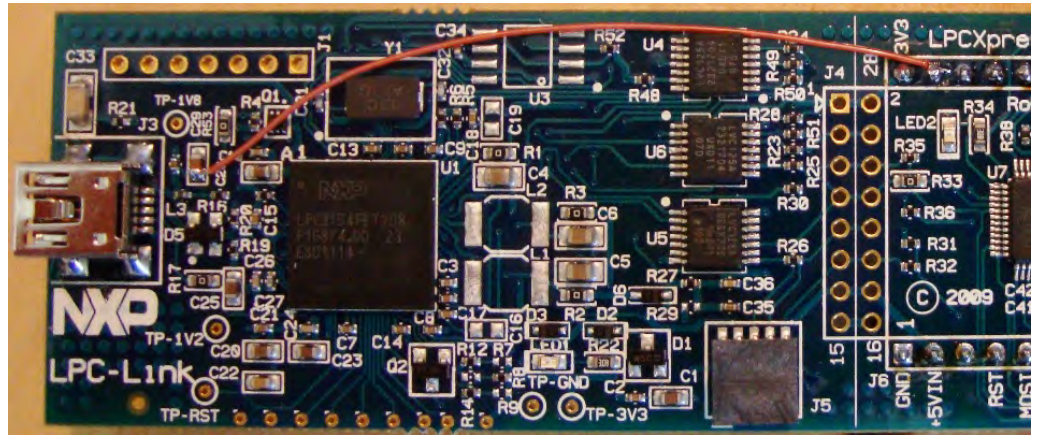


Fig 37. Modification on the LPCXpresso Board

To get a more detailed description of the Tusa Board and the modifications in the LPCXpresso controller board, please visit the product website at Silica [7].

After doing the modifications the hardware is ready and one can put the boards together like shown on the following figure.

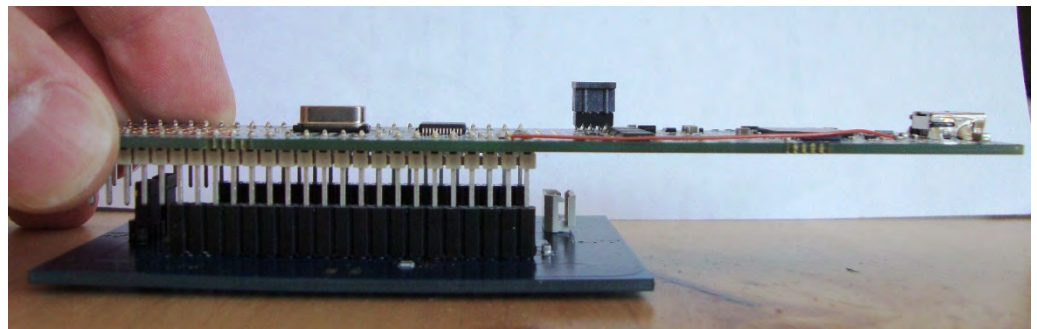


Fig 38. Combine the Tusa Board and the LPCXpresso Board

For information about the needed changes in the projects to work with the Tusa board, please see section 0.

## 7. Supplementary Notes

### 7.1 Software architecture

The software of the reference reader is based on the NXP reader library [6]. 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)

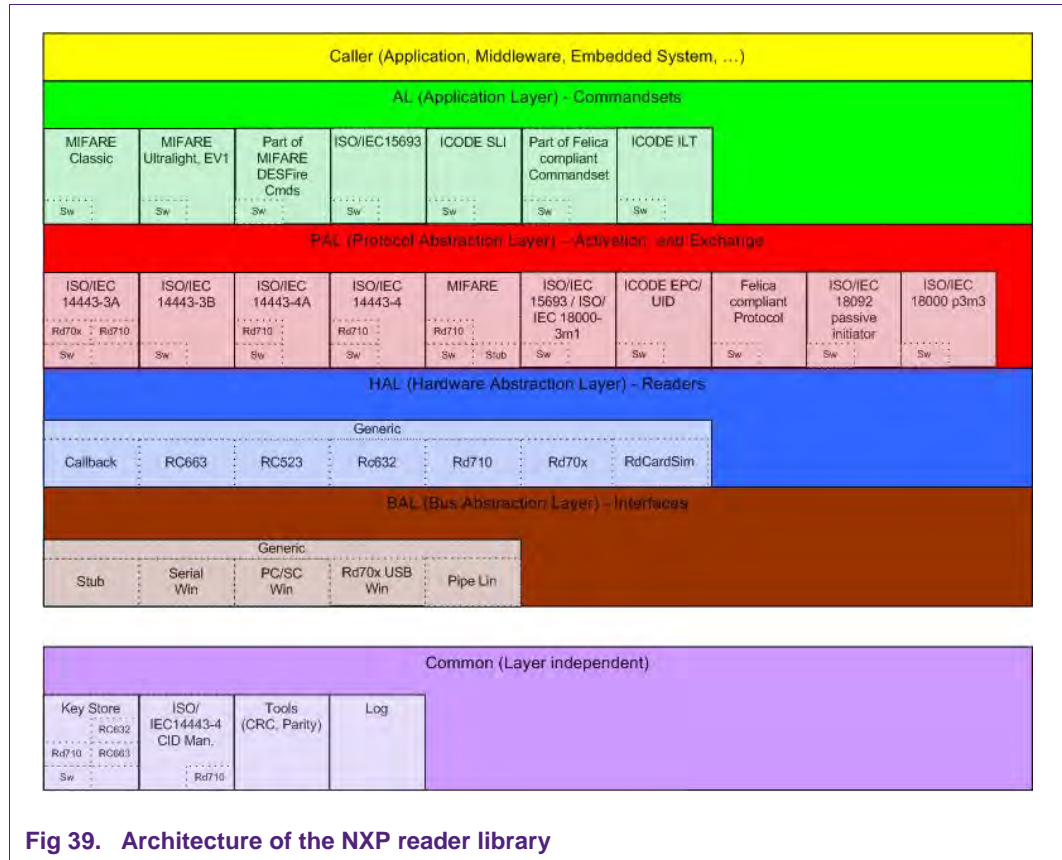


Fig 39. Architecture of the NXP reader library

For a detailed description of the NXP Reader Library please refer to the user manual **UM10663 - NXP Reader Library User Manual based on CLRC663 and PN512 Blueboard Reader projects**. It can be downloaded web site of the CLRC663 Blueboard [8].

### 7.1.1 Bus abstraction layer

This layer offers functions to abstract the hardware parts of the LPC11XX microcontroller.

These functions use the specific libraries available for the LPC11XX 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 LPC11XX family and therefore cannot be ported to other microcontrollers.

### 7.1.2 Hardware abstraction layer

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

### 7.1.3 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

#### 7.1.4 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 [1].

### 7.2 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 RC663 in the project window of the LPCXpresso™ IDE,
- Right click of the mouse → Select build configuration,
- Set active Debug build (or Release build).



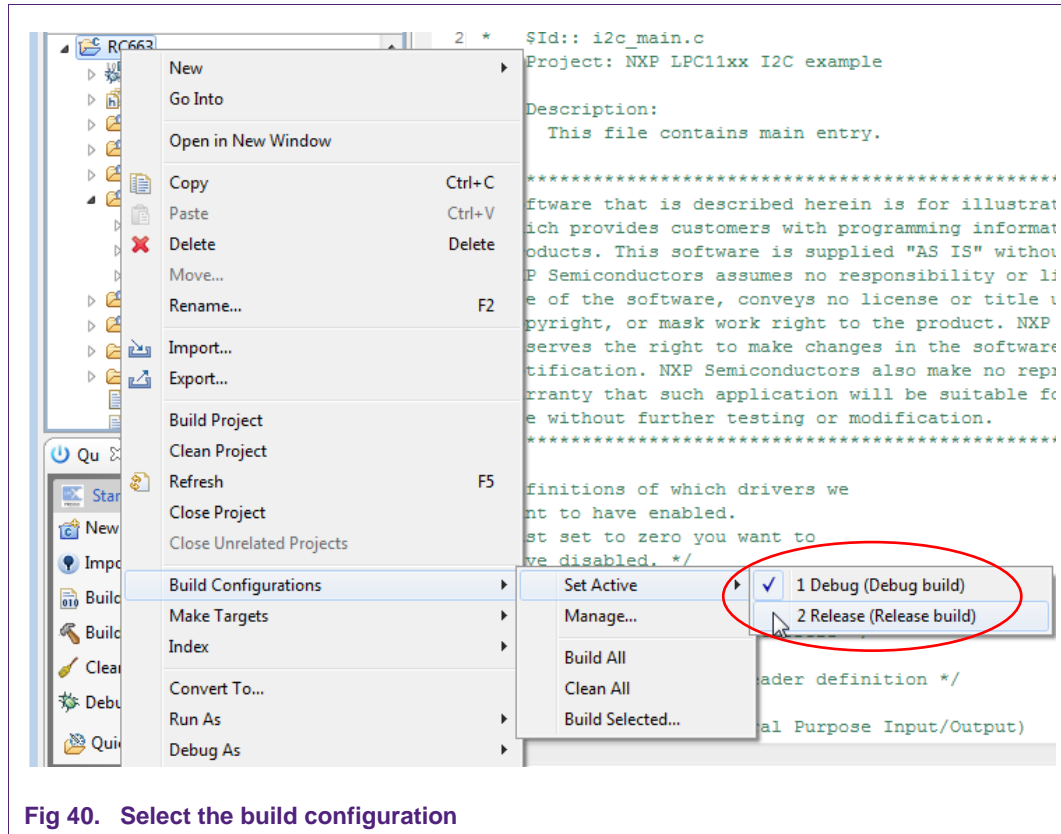


Fig 40. Select the build configuration

### 7.3 Level of compiler optimization

When the code size at the current compiler level overloads the FLASH size of the target board (32K for the ARM-based microcontroller LPC 1114), 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 RC663 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.

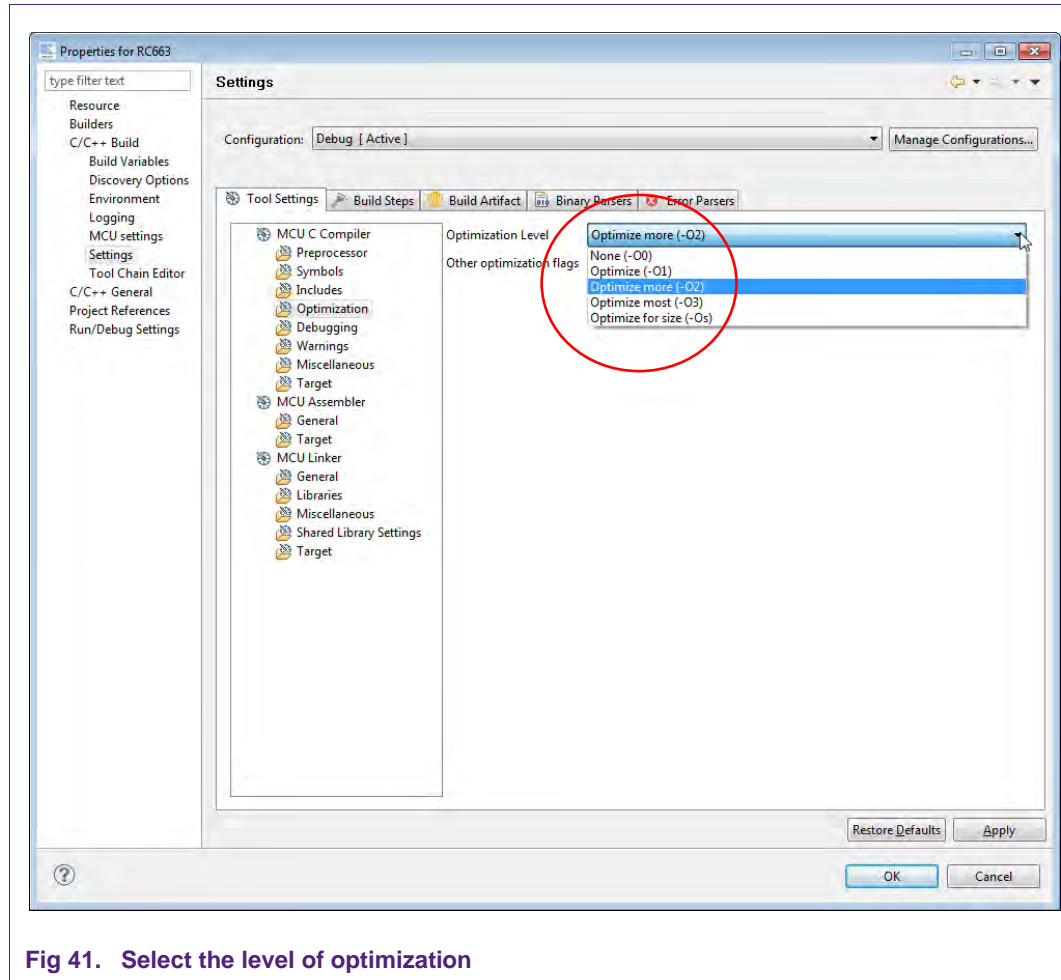


Fig 41. Select the level of optimization

### 7.3.1 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.

### 7.4 Optimizing the code size of the NXP Reader Library

A detailed description on how to optimize the code size of the NXP Reader Library for the use with one specially defined reader IC and card type please refer to the attached documents on the product page of the CLEV663B [8]. On that page one can also find an exemplary project for the use of the MIFARE Classic card in conjunction with the CLRC663 reader IC.

## 7.5 Preparing the projects for the use of the Blueboard in SPI configuration

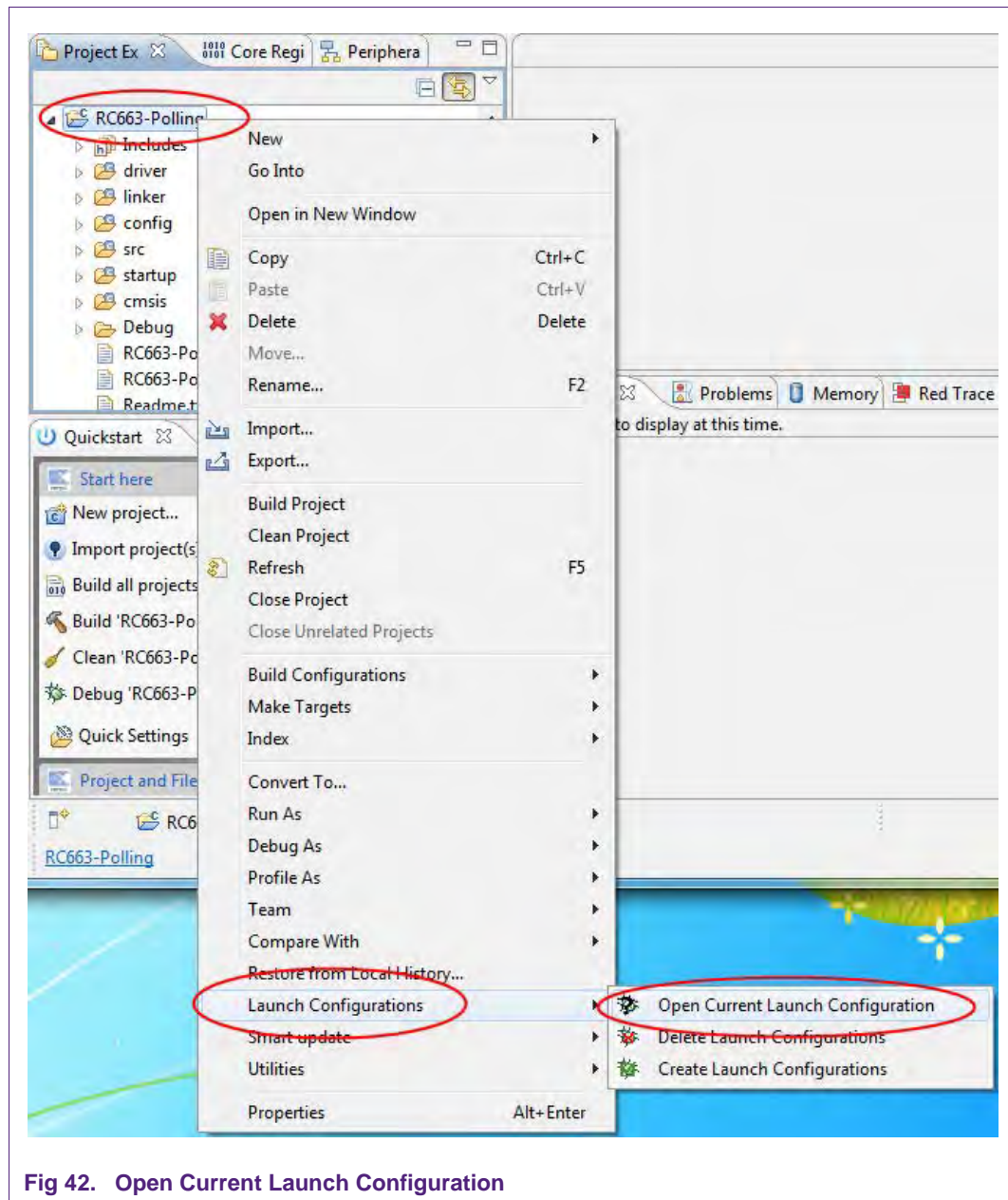
To use the projects in SPI configuration one has to do some small adaptations in the file `hw_config.h` located in `..\src\`.

1. Open the file `hw_config.h` and
  - a. comment the line `#define I2C_USED`.
  - b. uncomment the line `#define SPI_USED`.

## 7.6 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 42. 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.



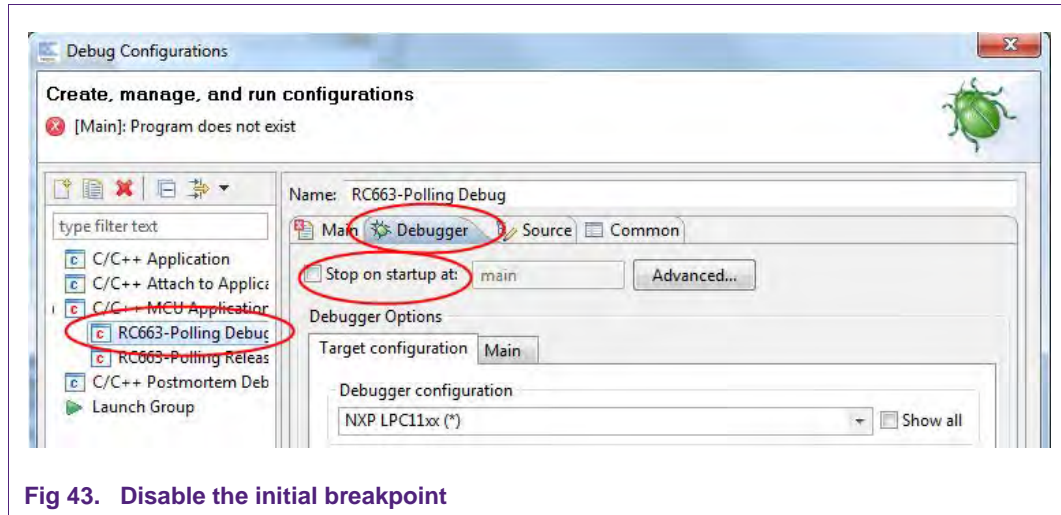


Fig 43. Disable the initial breakpoint

### 7.7 Replacing the LPCXpresso LPC1114 with the LPCXpresso LPC1115

Solder the multipoint connector onto the LPCXpresso LPC1115 in the same way as shown in Chapter 2.5. At this point there is no difference between the LPC1114 and the LPC1115.

In the IDE please do the following changes:

1. In the menu of the LPCXpresso IDE choose Project → Properties.
2. Choose C/C++ Build → MCU settings.
3. In the list choose the entry LPC1115/303 and click OK.

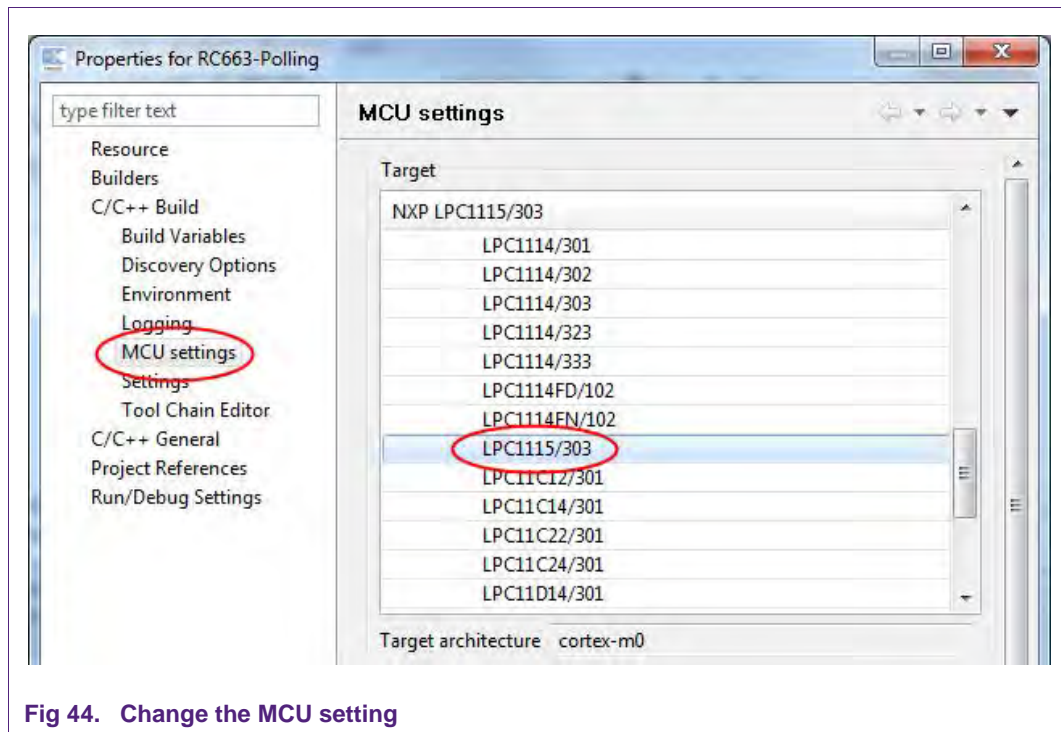


Fig 44. Change the MCU setting

Now, one can start working with the LPCXpresso LPC1115. There is no need to change anything in the code.

## 7.8 Replacing the LPCXpresso LPC1114/LPC1115 with the LPCXpresso LPC1227

Solder the multipoint connector onto the LPCXpresso LPC1227 in the same way as shown in Chapter 2.5. At this point there is no difference between the LPC1114/LPC1115 and the LPC1227. The projects for the LPC1114/LPC1115 are fully compatible with the LPCXpresso LPC1227 board by easily changing IDE configuration. One has to do only two subtle changes in IDE build configuration.

Change MCU in LPCXpresso IDE

1. In the menu of the LPCXpresso IDE choose Project → Properties.
2. Choose C/C++ Build → MCU settings.
3. In the list choose the entry LPC1277/301 and click OK.

Set either Build Configuration either Debug or Release

- Click on the project RC663 in the project window of the LPCXpresso™ IDE,
- Right click of the mouse → Select Build Configuration,
- Set active Debug1227 build (or Release1227 build) for LPC 1227.

## 7.9 Preparing the projects for the use with the 3d party Tusa Board

One needs to make two modifications in the code to get the Tusa Board working:

1. Point to the file ...\\src\\hw\_config.h and uncomment the line  
`#define TUSA // If using the Silica TUSA Board, set TUSA.`
2. Configure the project to use SPI communication. The needed steps are described in section 7.5.

## 8. References

---

- [1] **RC663 and NXP Reader Library**  
[http://www.nxp.com/documents/application\\_note/AN11021.pdf](http://www.nxp.com/documents/application_note/AN11021.pdf)
- [2] **LPCXpresso website**  
[www.nxp.com/redirect/lpcware.com/lpcxpresso/downloads/older](http://www.nxp.com/redirect/lpcware.com/lpcxpresso/downloads/older)
- [3] **RC663 data sheet**  
[http://www.nxp.com/documents/data\\_sheet/CLRC663.pdf](http://www.nxp.com/documents/data_sheet/CLRC663.pdf)
- [4] **LPC11XX family User Manual**  
[http://www.nxp.com/documents/data\\_sheet/LPC111X.pdf](http://www.nxp.com/documents/data_sheet/LPC111X.pdf)
- [5] **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](http://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](http://www.nxp.com/redirect/conrad.at/ce/de/product/736427/BUCHSENLEISTE-EINREIHIG-36-POLIG-RM254)
- [6] **Direct link to the NXP Reader Library**  
<http://www.nxp.com/documents/software/200310.zip>
- [7] **Tusa Board at the Silica website**  
[www.nxp.com/redirect/silica.com/products/highlight/product/silica-tusa-board.html](http://www.nxp.com/redirect/silica.com/products/highlight/product/silica-tusa-board.html)
- [8] **CLEV663B demo board site**  
<http://www.nxp.com/demoboard/CLEV663B.html>
- [9] **NXP Reader Library P2P user manual**  
[http://www.nxp.com/documents/user\\_manual/UM10721.pdf](http://www.nxp.com/documents/user_manual/UM10721.pdf)
- [10] **Technical Specification** – Simple NDEF Exchange Protocol, NFCForum-TS-SNEP\_1.0  
[www.nxp.com/redirect/nfc-forum.org/specs/spec\\_license](http://www.nxp.com/redirect/nfc-forum.org/specs/spec_license)
- [11] **Technical Specification** Logical Link Control Protocol, NFCForum-TS-LLCP\_1.1  
[www.nxp.com/redirect/nfc-forum.org/specs/spec\\_license](http://www.nxp.com/redirect/nfc-forum.org/specs/spec_license)

## 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.

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 of 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 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

### 9.4 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.



## 10. List of figures

Fig 1.	Picture of RC663 demo board.....	3	Fig 22.	Product activation.....	15
Fig 2.	Picture of solder bridges .....	4	Fig 23.	Product activation.....	16
Fig 3.	Picture of LPCXpresso LPC1114 development board.....	6	Fig 24.	Product activation.....	16
Fig 4.	Multipoint Connectors we used .....	6	Fig 25.	Importing a project into the LPCXpresso IDE..	17
Fig 5.	LPCXpresso with the Multipoint Connectors .....	7	Fig 26.	Importing a project into the LPCXpresso IDE..	18
Fig 6.	Connect the two boards .....	7	Fig 27.	Importing a project into the LPCXpresso IDE..	18
Fig 7.	Picture of RC663 demo board with the connectors joined alternatively.....	7	Fig 28.	Run the project.....	19
Fig 8.	LPCXpresso with the Multipoint Connectors used in the alternative way.....	7	Fig 29.	Run the project.....	20
Fig 9.	Connect the two boards the alternative way .....	8	Fig 30.	Run the project.....	20
Fig 10.	Pin RXP .....	8	Fig 31.	Run the project.....	21
Fig 11.	Pin RXN .....	9	Fig 32.	Run the project.....	21
Fig 12.	Pin TVDD .....	9	Fig 33.	Stop the project .....	22
Fig 13.	Blueboard in SPI configuration.....	10	Fig 34.	Debug Buttons .....	22
Fig 14.	Relevant solder bridges for the SPI and I <sup>2</sup> C configuration of the Blueboard version 3.0.....	10	Fig 35.	Importing the project .....	24
Fig 15.	Enumeration of the LPCXpresso Board in Device Manager Window .....	11	Fig 36.	Tusa Board.....	26
Fig 16.	LPCXpresso installation setup wizard 1 .....	12	Fig 37.	Modification on the LPCXpresso Board.....	27
Fig 17.	LPCXpresso installation setup wizard 2 .....	13	Fig 38.	Combine the Tusa Board and the LPCXpresso Board.....	27
Fig 18.	LPCXpresso installation setup wizard 3 .....	13	Fig 39.	Architecture of the NXP reader library.....	28
Fig 19.	Windows Security dialog.....	14	Fig 40.	Select the build configuration .....	30
Fig 20.	LPCXpresso installation setup wizard 4 .....	14	Fig 41.	Select the level of optimization.....	31
Fig 21.	LPCXpresso IDE .....	15	Fig 42.	Open Current Launch Configuration .....	33
			Fig 43.	Disable the initial breakpoint .....	34
			Fig 44.	Change the MCU setting .....	34

## 11. Contents

<b>1.</b>	<b>Introduction .....</b>	<b>3</b>	7.3.1	Optimization issues .....	31
<b>2.</b>	<b>Hardware overview of the Demo Reader .....</b>	<b>3</b>	7.4	Optimizing the code size of the NXP Reader Library .....	31
2.1	RC663 demo board (Blueboard) .....	3	7.5	Preparing the projects for the use of the Blueboard in SPI configuration .....	32
2.1.1	Derivates of the RC663 demo board (Blueboard) .....	4	7.6	Removing the initial breakpoint on debug startup .....	32
2.2	CE certification of the Blueboard .....	5	7.7	Replacing the LPCXpresso LPC1114 with the LPCXpresso LPC1115 .....	34
2.3	LPCXpresso LPC1114 development board .....	5	7.8	Replacing the LPCXpresso LPC1114/LPC1115 with the LPCXpresso LPC1227 .....	35
2.4	Alternative to the LPCXpresso LPC1114 .....	6	7.9	Preparing the projects for the use with the 3d party Tusa Board .....	35
2.5	Preparation of the hardware .....	6	<b>8.</b>	<b>References .....</b>	<b>36</b>
2.6	Interesting points of measurement .....	8	<b>9.</b>	<b>Legal information .....</b>	<b>37</b>
2.6.1	RXP - receiver input pin for the received RF signal .....	8	9.1	Definitions .....	37
2.6.2	RXN - receiver input pin for the received RF signal .....	8	9.2	Disclaimers .....	37
2.6.3	TVDD - transmitter voltage supply .....	9	9.3	Licenses .....	37
2.7	Preparing the Blueboard for the use with SPI or I <sup>2</sup> C .....	9	9.4	Trademarks .....	37
2.7.1	Blueboard version 2.1 and below .....	9	<b>10.</b>	<b>List of figures .....</b>	<b>38</b>
2.7.2	Blueboard version 3.0 and above .....	10	<b>11.</b>	<b>Contents .....</b>	<b>39</b>
<b>3.</b>	<b>Installation of the LPCXpresso Board .....</b>	<b>11</b>			
<b>4.</b>	<b>Managing the Demo Reader project with LPCXpresso IDE .....</b>	<b>11</b>			
4.1	Installation of LPCXpresso IDE .....	12			
4.2	Extraction of the demo reader project .....	16			
4.3	Start the project .....	19			
4.3.1	Run the project .....	19			
<b>5.</b>	<b>Associated Projects .....</b>	<b>23</b>			
5.1	Communication with MIFARE Ultralight .....	23			
5.2	Communication with MIFARE Classic .....	23			
5.3	Communication with MIFARE DESFire .....	23			
5.4	Polling .....	23			
5.5	Peer to Peer functionality .....	23			
5.5.1	Installation .....	23			
5.5.2	SNEP client project .....	24			
5.5.2.1	What is going on inside? .....	24			
5.5.2.2	Choosing the NDEF message .....	25			
<b>6.</b>	<b>Other supported hardware by the projects .....</b>	<b>26</b>			
<b>7.</b>	<b>Supplementary Notes .....</b>	<b>27</b>			
7.1	Software architecture .....	27			
7.1.1	Bus abstraction layer .....	28			
7.1.2	Hardware abstraction layer .....	28			
7.1.3	Protocol abstraction layer .....	28			
7.1.4	Application layer .....	29			
7.2	Build configuration .....	29			
7.3	Level of compiler optimization .....	30			

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