UM10883

PN7462AU Quick Start Guide - Customer Board

Rev. 1.0 — 29 March 2016
319810

User manual
COMPANY PUBLIC

Document information

Info	Content
Keywords	PN7462AU, Customer board, Quick Start Guide, functional description of the customer board
Abstract	This document describes the required basic circuitry to operate the PN7462AU and it also describes how to setup and use the PN7462AU Customer Demo board V2.1

Revision history

Rev	Date	Description
1.0	20160329	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

1. Introduction

This document describes the basic requirements to power up the PN7462AU Customer Demo board. This document does not replace the PN7462AU datasheet. Please refer to the datasheet to read more information on each chapter. This document also gives an overview of the PN7462AU Customer Demo board V2.1 and describes the function and first steps of using the board and the PN7462AU.

2. Hardware overview of the PN7462AU Customer Demo Board

2.1 PN7462AU Customer Demo Board

The Fig 1 shows the PN7462AU Customer Demo board version 2.1.

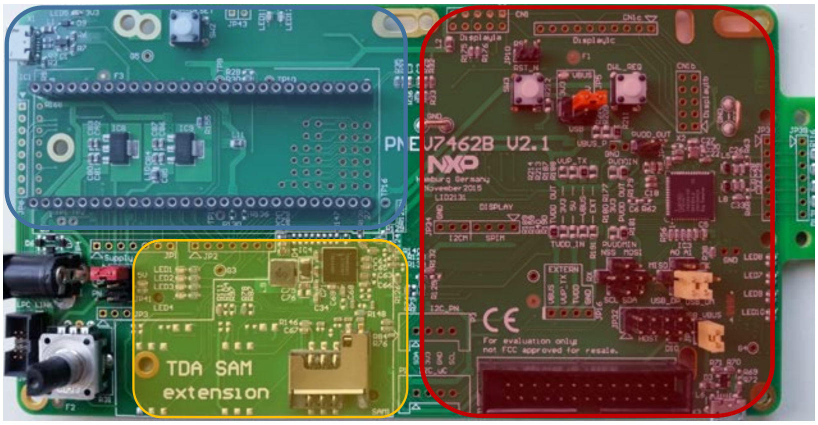


Fig 1. PN7462AU Customer board V2.1

The board is separated into 3 main blocks:

- PN7462AU (marked in red)
- LPCXpresso (marked in blue)
- TDA8026 (marked in yellow)

2.1.1 PN7462AU Block

This block containing the PN7462AU (Fig 2), the supply possibilities and the connections to all need pins of the IC.

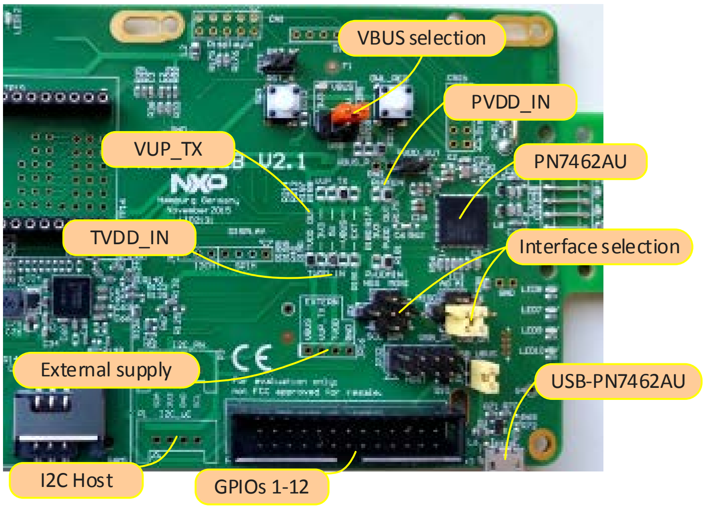


Fig 2. PN7462AU Customer Demo board v2.1

The settings shown in Fig 2 are the standard settings for using I2C and supplying the IC with 3.3 V. In these settings PVDD_IN, PVDD_M_IN, VBUS and VUP_TX are supplied with 3.3 V by using the onboard voltage regulator.

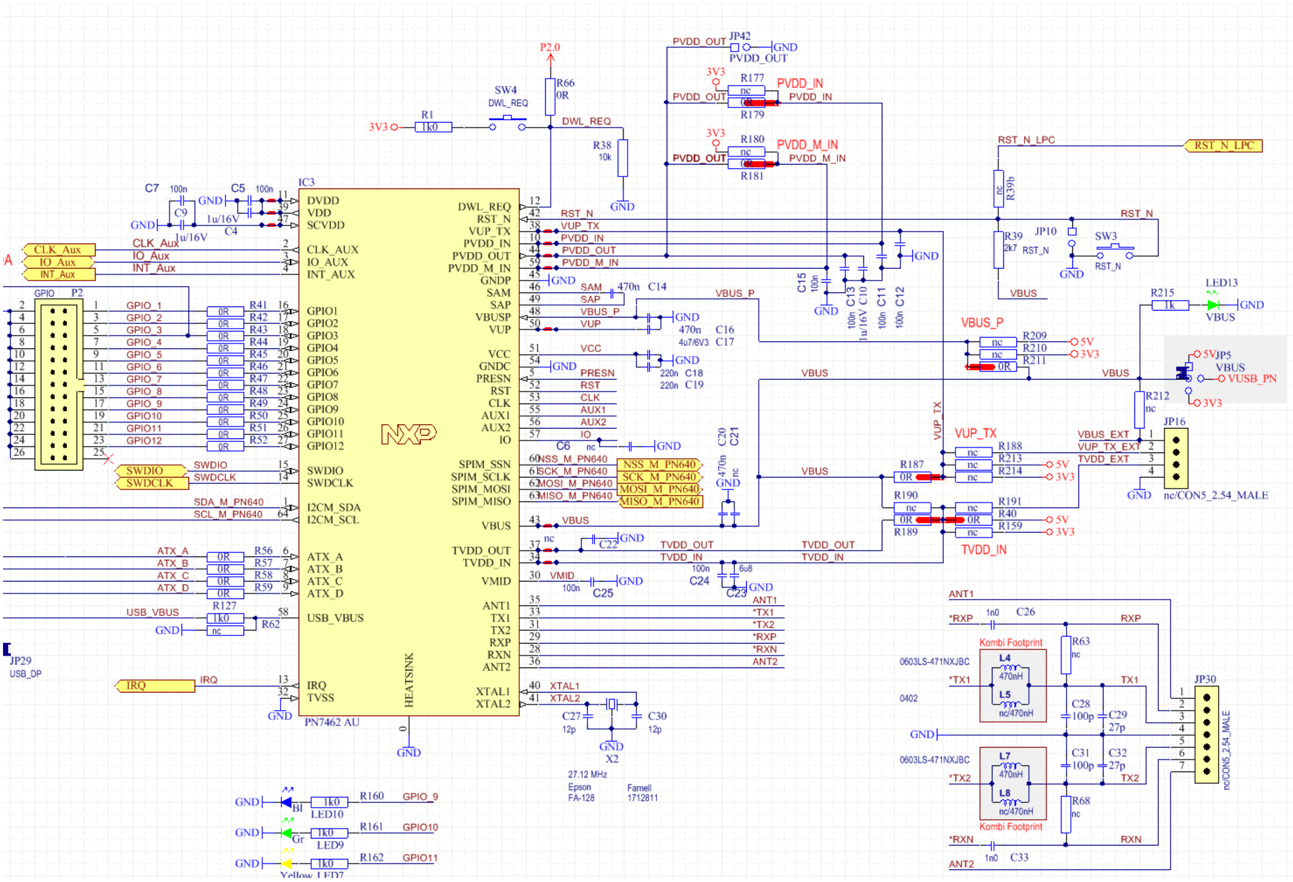


Fig 3. PN7462AU Customer Demo board V2.1 schematic

2.2 LPCXpresso development board – LPC1769

The LPC1769 is a Cortex-M3 microcontroller for embedded applications featuring a high level of integration and low power consumption at frequencies of 120 MHz. Features include 512 kB of flash memory, 64 kB of data memory, Ethernet MAC, USB Device/Host/OTG, 8-channel DMA controller, 4 UARTs, 2 CAN channels, 3 SSP/SPI, 3 I2C, I2S, 8-channel 12-bit ADC, 10-bit DAC, motor control PWM, Quadrature Encoder interface, 4 general purpose timers, 6-output general purpose PWM, ultra-low power Real-Time Clock with separate battery supply, and up to 70 general purpose I/O pins. The LPC1769 is pin-compatible to the 100-pin LPC2368 ARM7 MCU.

2.3 LCD

The PN7462AU customer board is prepared for using the connect EVE FT800 Breakout Board, for more information on the Display board please check www.nxp.com/redirect/mikroe.com/add-on-boards/display/connecteve/

The board can be connected in different ways to the PN7462AU Board.

2.4 Preparation of the HW

2.4.1 Interface type

The PN7462AU supports direct interfacing of various hosts as the SPI/M, I2C/M, SWD, USB and HSU.

2.4.2 PN7462AU Customer Demo board v2.1

The interface can be selected by jumper settings.

2.4.3 PN7462AU Customer Demo board v2.1 jumper settings

The following section describes the settings in the PN7462AU board V2.1 for each block and interface.

The jumper settings marked in yellow (Fig 4) are the default settings of the board. The settings shown in that figure are using external supply and 5 V VBUS supply.

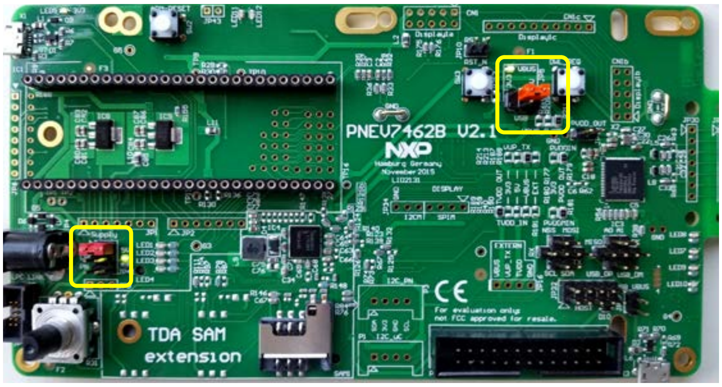


Fig 4. Default jumper settings

2.4.3.1 USB Host Interface

The yellow marked jumpers (Fig 5) shows how the board need to be set for using the USB host interface of the chip. The USB mini connector, which can be used in this selection, is on the lower right corner of the board.

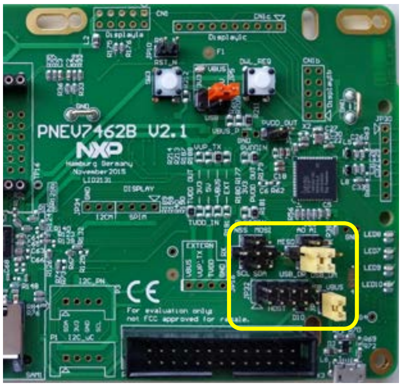


Fig 5. Host Interface selection – USB mode

2.4.3.2 I2C Host Interface

The yellow marked jumpers (Fig 6) need to be set for using the I2C host interface of the chip. This will connect the I²C SCL of the PN7462AU to the I/O P0(28) and also the SDA of the PN7642AU to the I/O P0(27) of the LPCXpresso 176x.

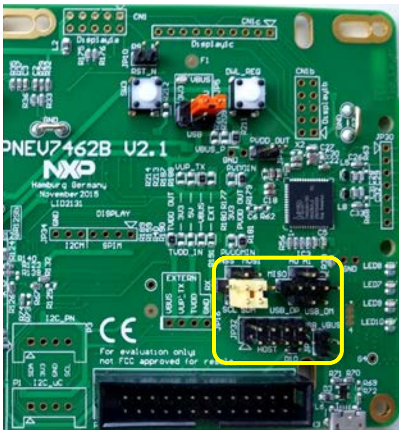


Fig 6. Host Interface selection - I2C mode

2.4.3.3 SPI Host Interface

The yellow marked jumpers (Fig 7) need to be set for using the SPI host interface of the chip. This will connect the SPI_MOSI of the PN7462AU to the I/O P0(18), SPI_MISO to the I/O P0(17),SCK to the I/O P0(15), and also the NSS of the PN7642AU to the I/O P0(16) of the LPCXpresso 176x.

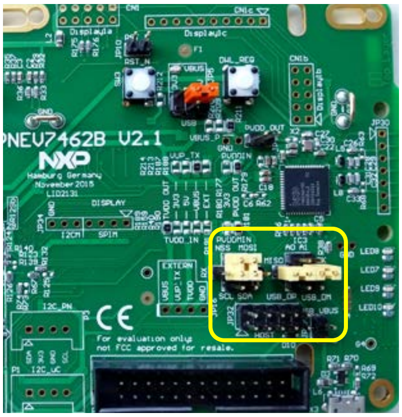


Fig 7. Host Interface selection - SPI

2.4.3.4 SWD settings

On the customer demo board the SWD connection is always enabled. The SWD/JTag connector is on the bottom left side of the board. LPC-Link2 board can be used to flash the PN7462AU. Fig 8 shows the position of the SWD/JTag connector on the customer demo board.

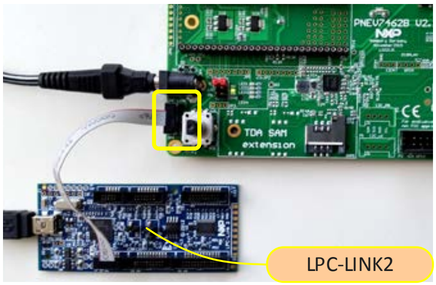


Fig 8. SWD/JTag connector

2.5 Antenna matching

The antenna matching of the PN7462AU customer board is separated in two parts. The matching with the damping resistors as well as the serial and parallel matching components and the EMC filter.

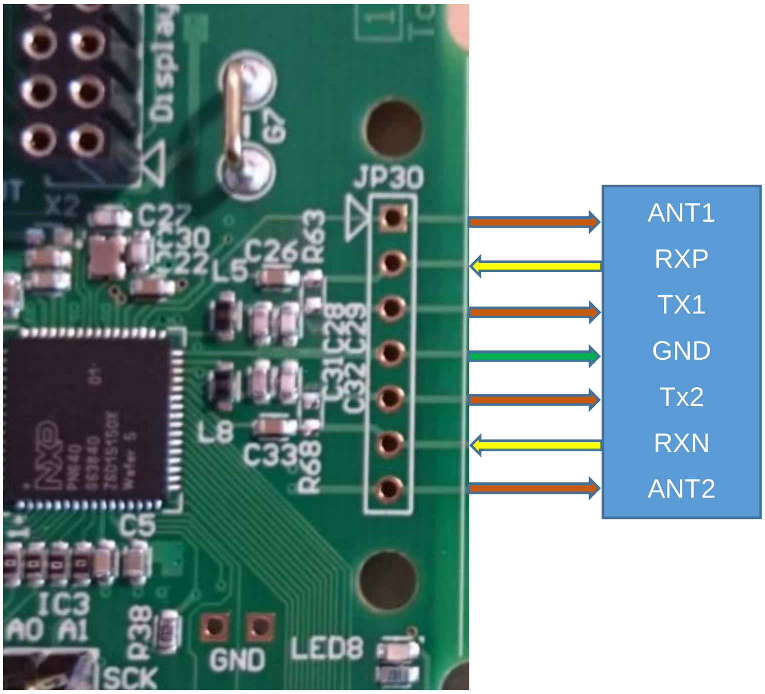


Fig 9. Pinning tor antenna connector

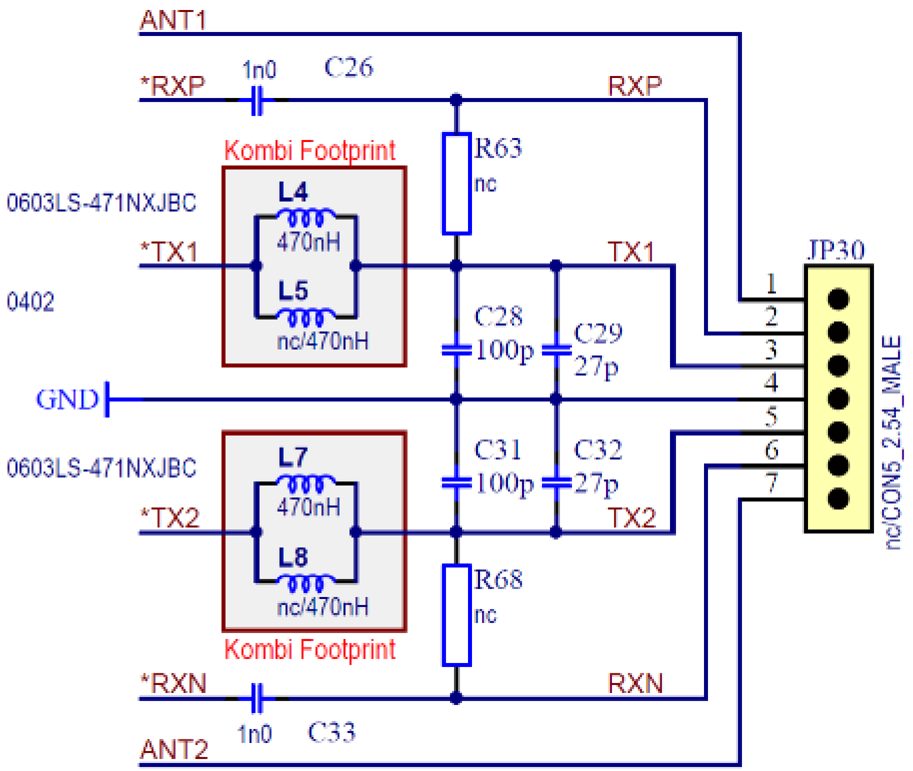


Fig 10. Antenna matching – EMC Filter

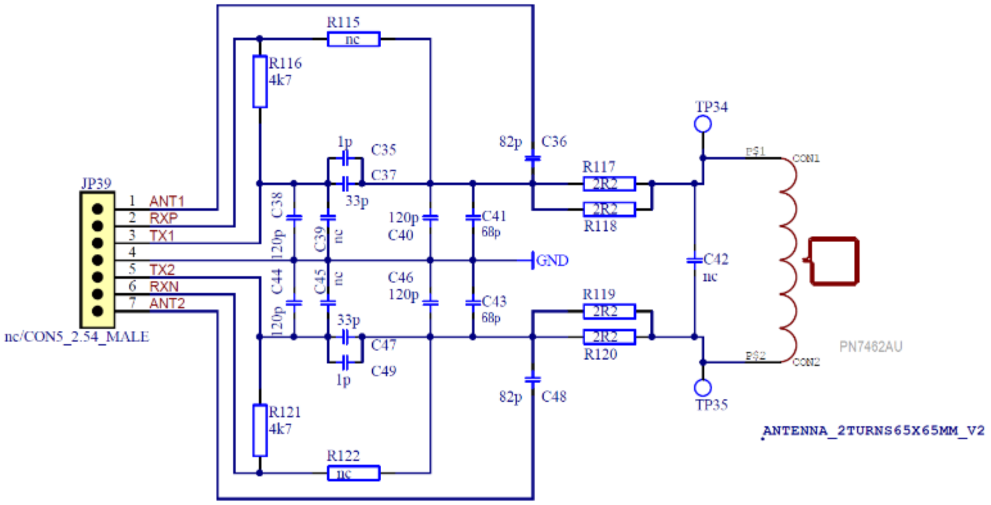


Fig 11. Antenna matching – Main Antenna

3. Configuration of the PN7462AU customer board

3.1 Board power setting

The PN7462AU customer board offers different possibilities of supplying the board and also the Pn7462AU.

For the board supply there are 3 options, external supply, LPC USB supply and PN7462AU USB supply that means the whole board can be supplied by one of these connectors.

Following jumper setting needs to be done to prepare the board for one of the three supplies. (The recommended supply is the external power supply!)

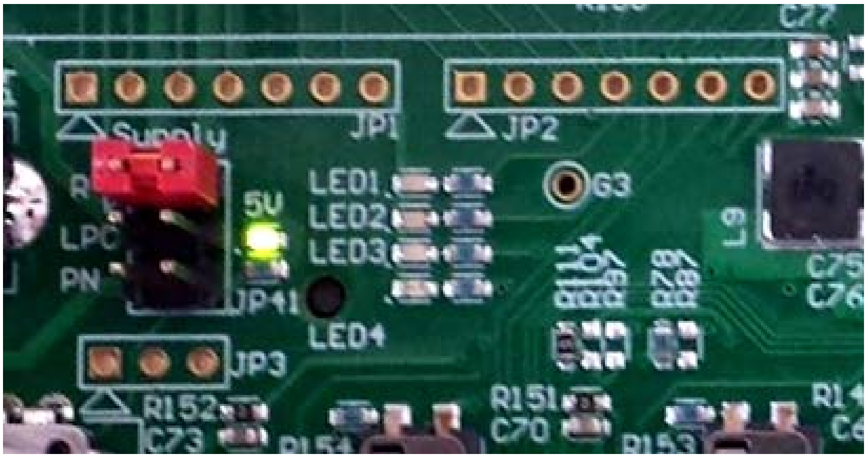


Fig 12. Board Supply Jumper

3.2 PN7462AU supply options

The boards offers several ways of supplying the PN7462AU

The default setting of the PN7462AU supply is set to use the internal supply for PVDD as well as TVDD. That means default setting is PVDD_IN connected to PVDD_OUT. TVDD_IN connected to TVDD_OUT.

The supply of the main chip supply (VBUS) can be set to 5 V 3.3 V or USB supply

The corresponding setting is described in Fig 13

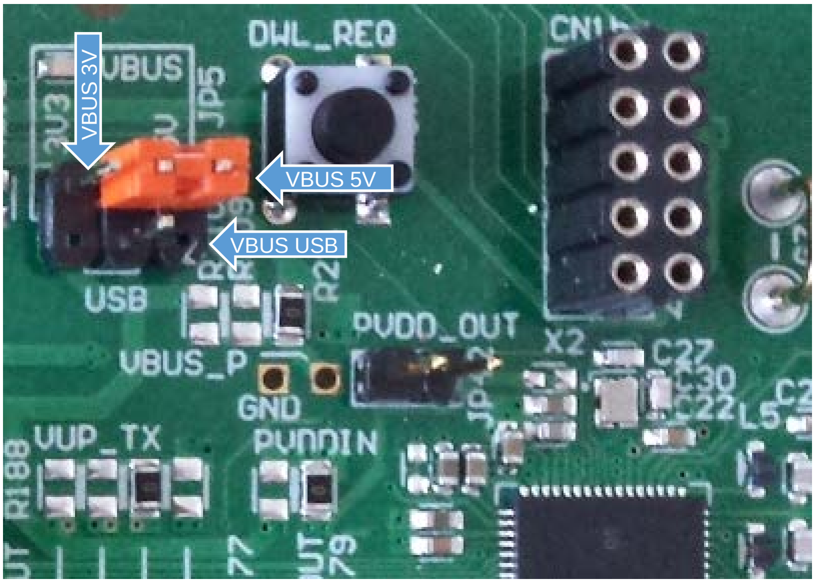


Fig 13. VBUS supply jumper setting

3.2.1 Supply status LED

If all jumpers are set correctly Following LED`s should light green,

3V3, 5 V and VBUS. In Fig 14 the location of the three different LED’s are shown.

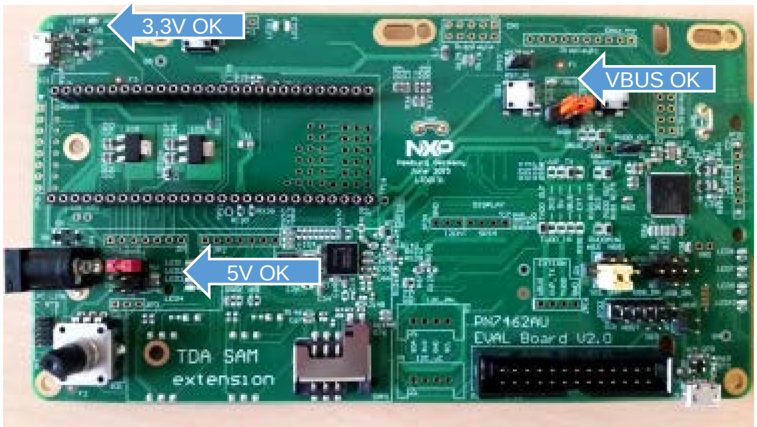
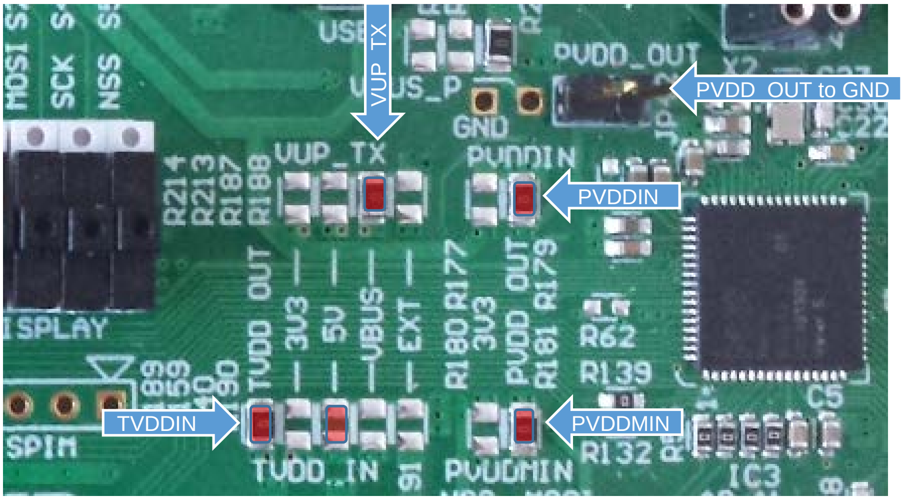


Fig 14. Supply indicator

3.2.2 Supply options for PVDD, VUP_TX and TVDD

The PN7462AU allows different options of supplying PVDDIN, PVDDMIN as well as for TVDDIN and VUP_TX.

The default setting on the customer Boards is marked in Fig 15 Jumper 42 needs to be open, to have PVDD_OUT activated by PN7462AU. (Fig 15)



(1) Red marked are the default settings

Fig 15. Default supply connection of the PN7462AU using all blocks

Added to the default settings the customer board offers following possible settings. To change one of the following supply inputs, the relevant (marked in Fig 15) needs to be set to the corresponding position. (Default settings are marked in green):

Table 1. Supply options

Supply options	
VUP_TX	3V3
	5V
	VBUS
	EXT
TVDD_IN	TVDD_OUT
	3V3
	5V
	VBUS
	EXT
PVDD_IN	3V3
	PVDD_OUT
PVDDM_IN	3V3
	PVDD_OUT

Note:

If PVDD is externally supplied the Jumper 42 (PVDD_OUT) needs to be set. By setting this Jumper the PVDD_OUT is shorted to GND and the PN7462AU turns off the PVDD LDO.

4. Software application stack

The PN7462AU Firmware is modular software written in C language, which provides an API that enables customers to create their own contact and contactless software stack and applications for the PN7462AU. This API facilitates all operations and commands required in contact and contactless applications such as reading or writing data to cards or tags, exchanging data with other NFC-enabled devices or allowing NFC reader ICs to emulate cards as well.

The PN7462AU software application stack consists of 4 main layers.

- Application & example layer
- Protocol abstraction layer – PAL
- Hardware abstraction layer – HAL
- OSAL (FreeRTOS) and utilities layer

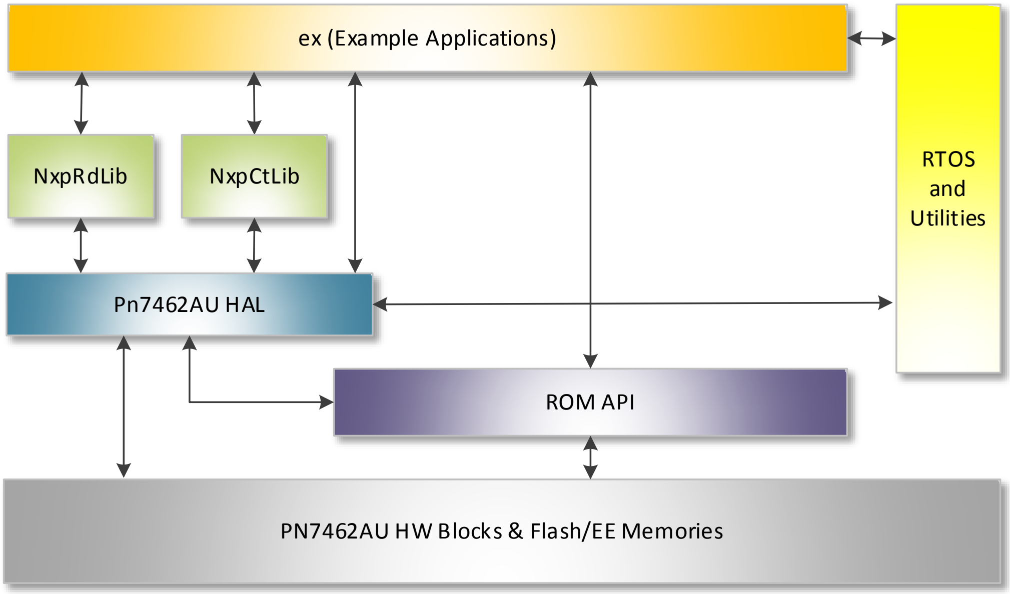

ex (Example Applications)
NxpRdLib
NxpCtLib
RTOS and Utilities
Pn7462AU HAL
ROM API
PN7462AU HW Blocks & Flash/EE Memories

Fig 16. Static overview of PN7462AU firmware

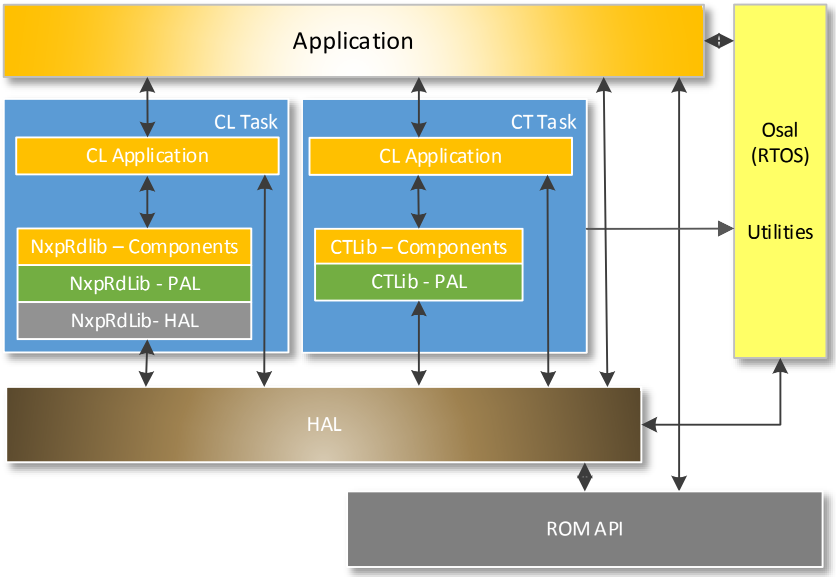


Fig 17. Architecture diagram

4.1 Hardware abstraction layer – HAL

Hardware abstraction layer – HAL is responsible for the CPU, communication, memory and utility peripherals. HAL composed of a set of HW functions, HW ISR and OSAL functions.

The HW functions can further be divided to:

1. Atomic functions: functions configuring the HW, but don´t result in any event from the HW, EEPROM, Flash, CRC, RNG, PMU/ PCR.
2. Blocking functions: functions configuring the HW and wait till one or more expected events occurs from the HW. CLIF HAL, CT HAL, I2CM/ SPIM HAL
3. Non-blocking functions: functions configuring the HW and expect one or more events, but don´t wait till it occurs. The events are notified to the caller of the function. Timer, Host interface.

The HW ISR handles HW events (interrupts) and signals of the blocking functions or notifies non-blocking functions. The HW ISR also handles time critical HW configuration or functions.

4.2 Protocol abstraction layer – PAL

Protocol abstraction layer – PAL implement HW independent communication protocols for contactless and contact interface and it is composed of two libraries.

NfcRdLib library implement contactless protocol and application components. Followed ISO/IEC contactless standards protocols are available:

- **ISO14443-3A**: Contactless proximity card air interface communication at 13.56MHz for the Type A and Jewel contactless cards.

- **ISO14443-3B:** Contactless proximity card air interface communication at 13.56MHz for the Type B contactless cards.
- **ISO14443-4:** Specifies a half-duplex block transmission protocol featuring the special needs of a contactless environment and defines the activation and deactivation sequence of the protocol.
- **ISO14443-4A**: Transmission protocol for Type A contactless cards.
- **MIFARE (R):** Contains support for MIFARE authentication and data exchange.
- **ISO15693:** Contactless protocol for vicinity RFID. It operates on 13.56MHz and uses magnetic coupling between the reader and transponder.
- **ISO18000-3M3:** Contactless protocol for vicinity RFID. It is especially suited for applications where reliable identification and high anti-collision rates are required.
- **FeliCa** (JIS: X6319): Contactless RFID smart card system from Sony.
- **ISO/IEC 18092:** NFC Interface and Protocol standard that enables NFC Data Exchange protocol.

The contact protocol library implement the components for the contactless protocol, such as EMV ATR Parser, T=0 protocol, T=1 protocol. This library also handles the timing compliance violations.

4.3 Application layer – AL

In the application layer customer applications shall be implemented and can directly use HAL APIs or APIs from the PAL libraries.

The contactless example (or application) is either NFC Forum Polling Loop or EMV Polling Loop that branches to dedicated examples depending on the card detected such as MIFARE Classic, MIFARE UL, MIFARE DF, EMV Paypass transactions (PPSE). There exists a compile time macro phExMain_Cfg.h to decide whether the example is NFC Forum or EMV Polling Loop.

The contact example (or application) is an EMV contact (PPSE application on JCOP card) application that uses the T=1 protocol and the ATR processing of the protocol library.

4.4 OSAL and utilities layer

The OSAL and Utilities layer is used to abstract Free-RTOS messages, to handle events, signals and messages between HW functions and to handle HW ISR.

Utilities layer includes a set of utilities which are grouped and encapsulated together in an independent set of functions. Utilities components provide an interface for protocol libraries to use HAL APIs such as CRC, RNG etc.

Note:

Detailed description how to use OSAL and utilities layer refer to the CHM help file.

4.5 Component view

4.5.1 Contactless component view

In contactless component view (Fig 18) for the “phExMain” example is shown.

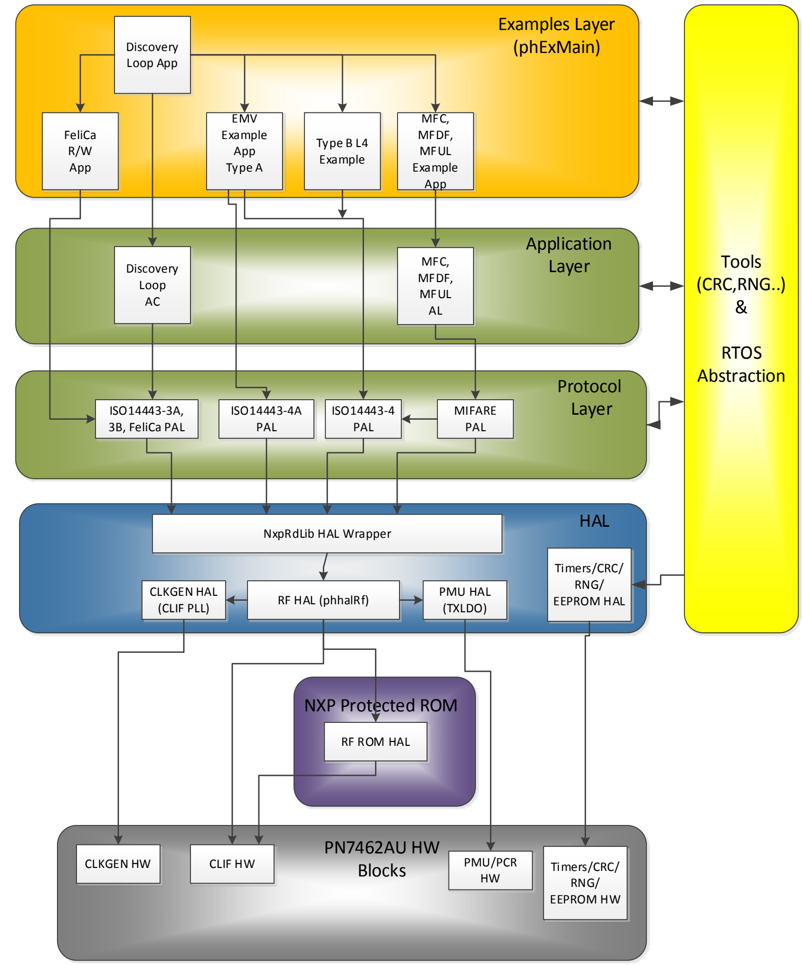


Fig 18. Contactless architecture view

4.5.2 Contact component view

In the Fig 19 contact component view for the “phExMain” example is shown.

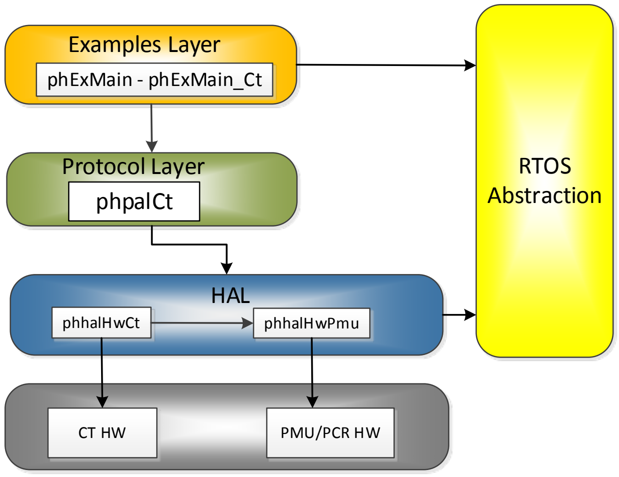


Fig 19. Contactless architecture view

4.6 Building a project from bottom to top

In order to use the PN7462AU firmware, a stack of components has to be initialized from bottom to top. Every component in the software stack has to be initialized before it can be used. The referred initialization of each layer generates a data context which feeds the immediate upper layer. Some of the components may need a data context coming from the same layer to be used as an entry point.

The Fig 19 illustrates the mentioned implementation for the initialization procedure of a “phExtMain” application.

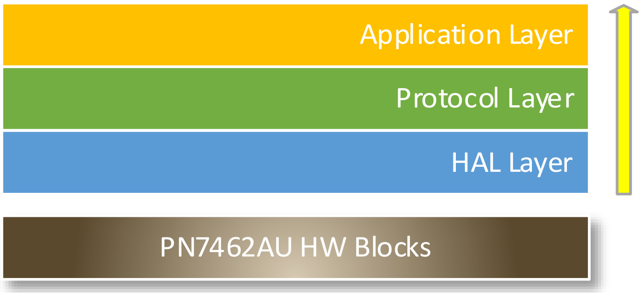


Fig 20. Project initialization order

4.7 RTOS and it´s usage

The PN7462AU FW is using FreeRTOS. The port.c file in the OpenRTOS source is modified to support disabling/enabling of scheduler (SysTick timer) and context switch (PendSV) during FW critical sections. The Cortex-M0 port is already available from FreeRTOS.

The FreeRTOS provides flexibility to develop multi-application environment. It provides the creation of multiple tasks. The FreeRTOS will handle multiple tasks with its scheduler. It is also possible to prioritize the tasks according to our requirement.

The FreeRTOS provides the message queues which are used to communicate between the tasks. The tasks can wait for the messages and if not available scheduler suspends these tasks which are waiting, and allow the other tasks to run.

The FreeRTOS provides the events which are used to communicate inside the tasks.

The tasks can go to suspended state waiting for the events as well. Whenever the events occur the scheduler wakes up that particular task and allows it to run.

For more information on FreeRTOS please refer the following link

www.nxp.com/redirect/freetos

The Fig 21 FreeRTOS Usage (below) provides the structure of FreeRTOS and its relation to PN7462AU FW Application.

The Flash boot performs the boot reason handling and initialization of common HALs.

See Below are the lists of examples available in current release to demonstrate the HW and FW features of PN7462AU IC.

In general, the FreeRTOS Scheduler has 2 default tasks running which are Idle task and

Timer task whose priority is kept lower than the application tasks.

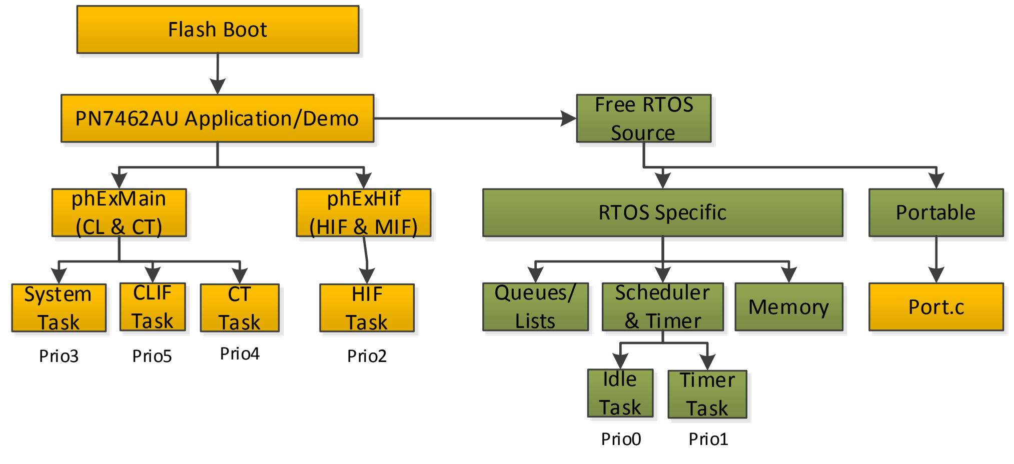


Fig 21. FreeRTOS usage diagram

Note:

Due to legal issues, NXP cannot provide source code for the FreeRTOS and the FreeRTOS source code should be integrated by users. Detailed description on how to integrate source code of the FreeRTOS to the project it is described in the “AN11784 PN7462AU How to integrate RTOS” document.

5. Managing the PN7462AU SW projects with LPCXpresso IDE

The PN7462AU SW projects are delivered in a zip package and can be extracted, edited, compiled and linked with LPCXpresso IDE.

The LPCXpresso IDE is a low-cost highly integrated software development environment for NXP's LPC microcontrollers and includes all the tools necessary to develop high-quality software solutions in a timely and cost effective fashion. LPCXpresso IDE is based on Eclipse and has many enhancements to simplify development with NXP LPC microcontrollers. It also features the industry-standard GNU tool chain, with a choice of a proprietary optimized C library or the standard “Newlib” library. The LPCXpresso IDE can build an executable of any size with full code optimization.

Designed for simplicity and ease of use, the LPCXpresso IDE provides software engineers a quick and easy way to develop their applications.

This tool can freely be 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.

5.1 Development environment

For developing PN7462AU firmware and customer applications all components listed in the Table 2 are required.

Table 2. Development environment

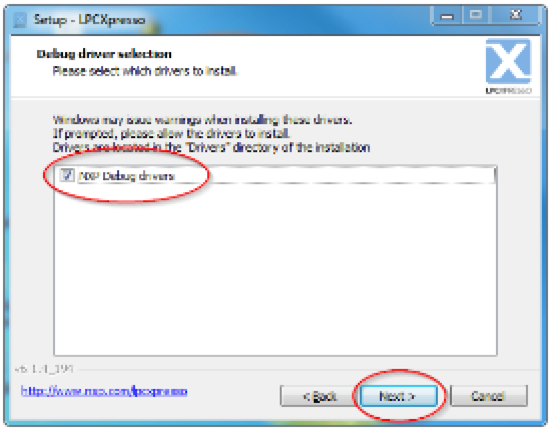
Item	Version	Purpose
PN7462AU customer board	2.1	Engineering development board
LPC Link 2	1.0	Standalone debug adaptor
LPCXpresso IDE	8.0.0 or higher	Development IDE
LPCXpresso PN7462AU plugin	com.nxp.pn7xxxxx.update-8.0.0-SNAPSHOT-150	Add PN7462AU reader to the LPCXpresso

5.2 Installation of the LPCXpresso IDE

The LPCXpresso IDE is installed into a single directory, of your choice. Unlike many software packages, the LPCXpresso IDE does not install or use any keys in the Windows Registry, or use or modify any environment variables (including PATH), resulting in a very clean installation that does not interfere with anything else on your PC. Should you wish to use the command-line tools, a command file is provided to set up the path for the local command window.

Multiple versions can be installed simultaneously without any issues.

The installation starts after double-clicking the installer file.



Setup - LPCXpresso

Debug driver selection

Please select which drivers to install.

Windows may issue warnings when installing these drivers.
If prompted, please allow the drivers to install.
Drivers are located in the "Drivers" directory of the installation

☑ NXP Debug drivers

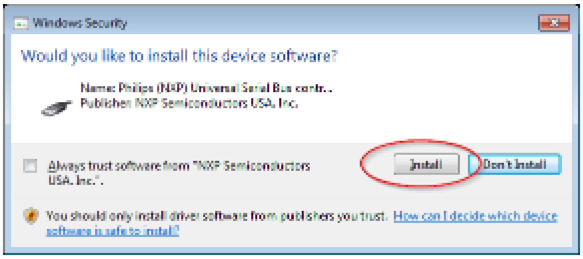
v8.1.4_190

http://www.nxp.com/lpcxpresso

Fig 22. LPCXpresso installation

Make sure, the checkbox for installing the NXP debug drivers is activated.

During the installation, the user will be asked to install some required drivers. The installation of these drivers shall be accepted.



Windows Security

Would you like to install this device software?

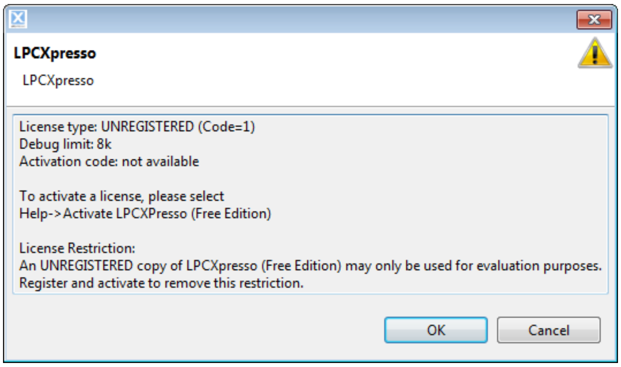
Name: Philips (NXP) Universal Serial Bus contr...
Publisher: NXP Semiconductors USA, Inc.

□ Always trust software from "NXP Semiconductors USA, Inc.".

You should only install driver software from publishers you trust. How can I decide which device software is safe to install?

Fig 23. Windows security dialog

After the setup wizard has finished, the newly installed IDE can be launched.



LPCXpresso

LPCXpresso

License type: UNREGISTERED (Code=1)
Debug limit: 8k
Activation code: not available

To activate a license, please select
Help->Activate LPCXPresso (Free Edition)

License Restriction:
An UNREGISTERED copy of LPCXpresso (Free Edition) may only be used for evaluation purposes.
Register and activate to remove this restriction.

Fig 24. LPCXpresso IDE

Directly after the first start of the LPCXpresso IDE, an info dialog will appear with the message of an unregistered copy of the LPCXpresso IDE. Confirm the dialog and follow the instructions on the Welcome Screen to get a registered version with the debug limit of 256k. The registration is free of a charge. The Link to the registration page is shown in the menu, Help → Activate LPCXpresso → Create Serial number and register.

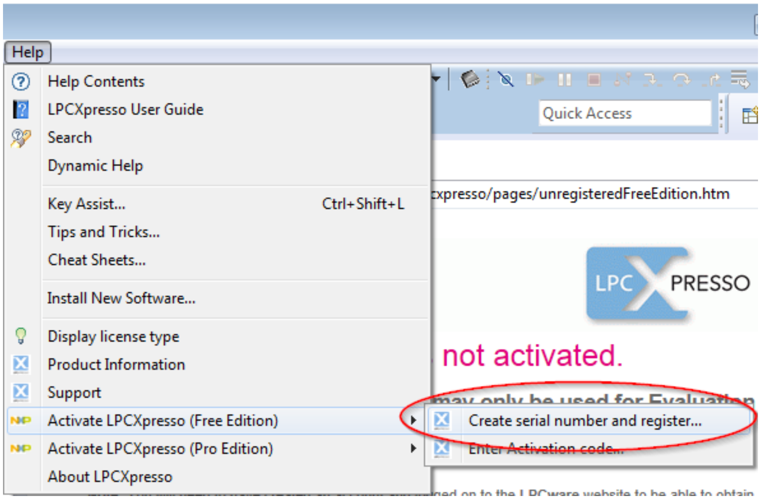
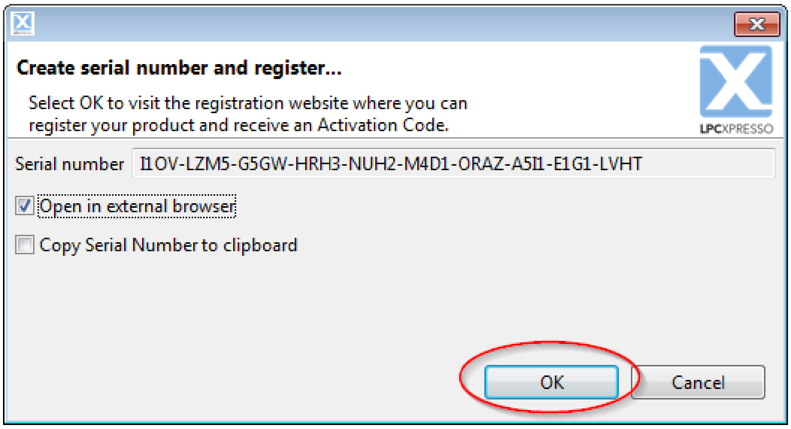


Fig 25. Product activation (1)



Create serial number and register...

Select OK to visit the registration website where you can register your product and receive an Activation Code.

Serial number I1OV-LZM5-G5GW-HRH3-NUH2-M4D1-ORAZ-A5I1-E1G1-LVHT

☑ Open in external browser

□ Copy Serial Number to clipboard

Fig 26. Product activation (2)

In case that you do not have an account on the www.nxp.com/lpcware, please sign up to get an activation code. The code will be sent to the provided e-mail address.

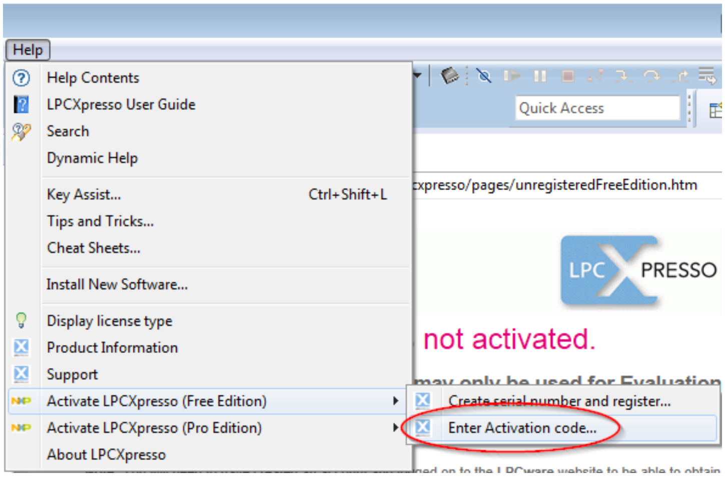


Fig 27. Product activation (3)

Once you receive the activation code open the activation window by pointing to Help → Activate LPCXpresso → Enter Activation code, and enter the code.

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

5.3 Adding PN7462AU Plugin

A separate PN7462AU Plugin is required for development of PN7462AU Firmware via LPCXpresso. With the plugin, a state-of-the-art development environment is available to the end user. PN7462AU Plugin is required for:

- Build PN7462AU Code in LPCXpresso
- Download PN7462AU Firmware via SWD+LPCLink2
- Access internal peripheral registers of the PN7462AU

To install the PN7462AU plugin, start LPCXpresso and follow the steps as shown below.

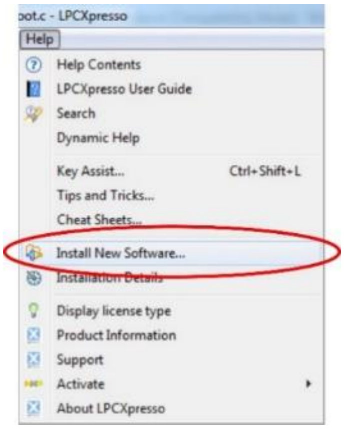
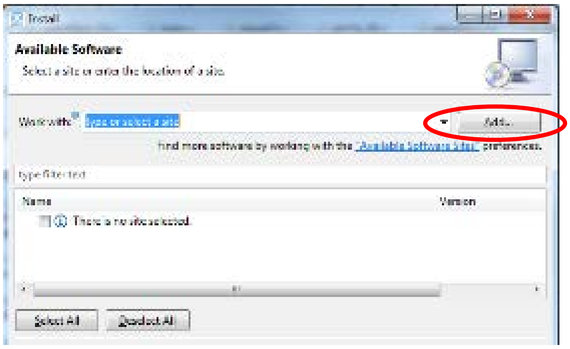


Fig 28. LPCXpresso – install new SW



Select a site or enter the location of a site.

Work with: type or select a site

Find more software by working with the "Available Software Sites" preferences.

type filter text

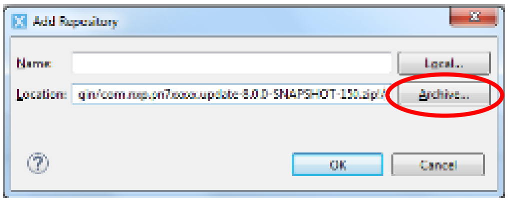
Name Version

There is no site selected.

(2) Click “Add”

Fig 29. LPCXpresso – installing LPC plugin/ adding installation repository

Plugin zip file is located in “.\PN7462AU Software\LPCXpresso Plugin” folder.



Name:

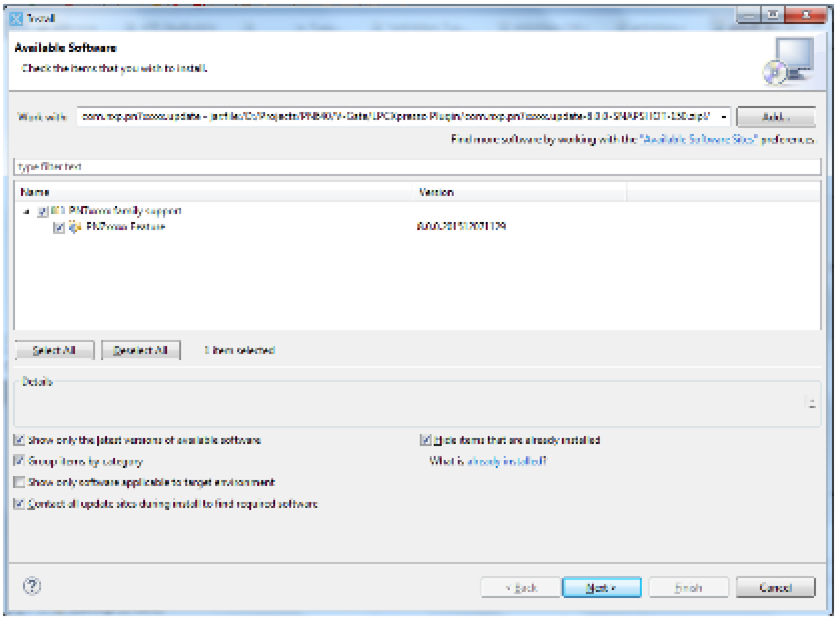
Location: gin/com.nxp.pn7xxxx.update-8.0.0-SNAPSHOT-150.zip!/

(3) Click “Archive”

(4) Browse to the provided LPCXpresso PN7462AU plugin zip file

(5) Select “com.nxp.pn7xxxxx.update-8.0.0-SNAPSHOT-150.zip” file

Fig 30. LPCXpresso – installing LPC plugin/ adding LPXpresso zip



Install

Available Software

Check the items that you wish to install.

Work with: com.nxp.pn7xxxx.update - jar:file:/D:/Projects/PN640/V-Gate/LPCXpresso Plugin/com.nxp.pn7xxxx.update-8.0.0-SNAPSHOT-150.zip!/

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
☑ PN7xxxx family support	
☑ PN7xxx Feature	8.0.0.201512071129

1 item selected

Details

☑ Show only the latest versions of available software

☑ Hide items that are already installed

☑ Group items by category

What is already installed?

□ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

(1) Select “PN7462AU Feature”

(2) Click “Next” and follow the standard installation flow

Fig 31. LPCXpresso – installing LPC plugin/ enable PN7462AU feature

5.4 Importing provided SW example projects

The use of quick start panel provides rapid access to the most commonly used features of the LPCXpresso IDE. Quick start panel allows easy import projects, create new projects, build and debug projects.

Provided FW and SW examples are part of the “PN7462AU Product Support Package”, this package have to be installed first on your PC.

To import the SW example projects please follow next steps.

- Start the LPCXpresso IDE and select new workspace
- 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.

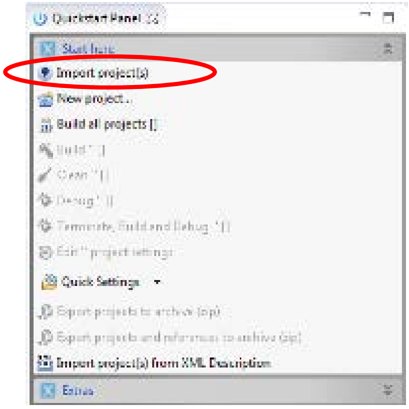
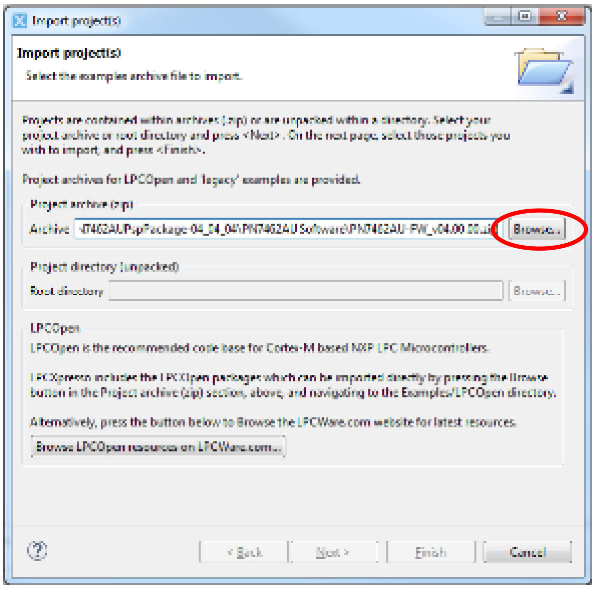


Fig 32. Importing project to LPCXpresso IDE (1)

In the “Quickstart Panel” window, which can be found on the left hand side, click on Import project(s).

The dialog for importing projects opens.



Import project(s)

Select the examples archive file to import.

Projects are contained within archives (.zip) or are unpacked within a directory. Select your project archive or root directory and press <Next>. On the next page, select those projects you wish to import, and press <Finish>.

Project archives for LPCOpen and 'legacy' examples are provided.

Project archive (zip)

Archive: \7462AU\PspPackage-04_04_04\PN7462AU Software\PN7462AU-FW_v04.00.00.zip

Project directory (unpacked)

Root directory:

LPCOpen

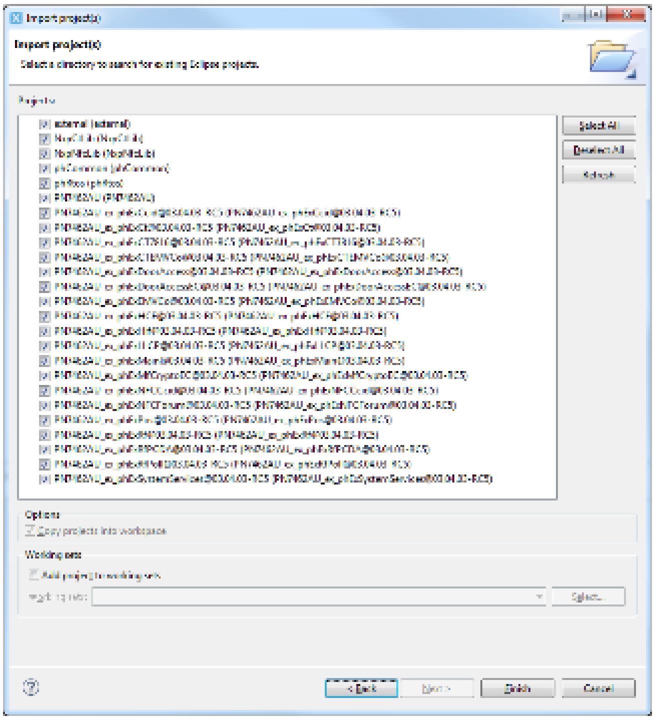
LPCOpen is the recommended code base for Cortex-M based NXP LPC Microcontrollers.

LPCXpresso includes the LPCOpen packages which can be imported directly by pressing the Browse button in the Project archive (zip) section, above, and navigating to the Examples/LPCOpen directory.

Alternatively, press the button below to Browse the LPCWare.com website for latest resources.

Fig 33. Importing project to LPCXpresso IDE (2)

Browse to the project zip file *“.\PN7462AU Software\PN7462AU-FW_v04.00.00.zip”* and click *“Next”.*



Import project(s)

Import project(s)

Select a directory to search for existing Eclipse projects.

Projects:

- ☑ external (external)
- ☑ NxpCtLib (NxpCtLib)
- ☑ NxpNfcLib (NxpNfcLib)
- ☑ phCommon (phCommon)
- ☑ phRtos (phRtos)
- ☑ PN7462AU (PN7462AU)
- ☑ PN7462AU_ex_phExCcid@03.04.03-RC5 (PN7462AU_ex_phExCcid@03.04.03-RC5)
- ☑ PN7462AU_ex_phExCT@03.04.03-RC5 (PN7462AU_ex_phExCT@03.04.03-RC5)
- ☑ PN7462AU_ex_phExCT7816@03.04.03-RC5 (PN7462AU_ex_phExCT7816@03.04.03-RC5)
- ☑ PN7462AU_ex_phExCTEMVCo@03.04.03-RC5 (PN7462AU_ex_phExCTEMVCo@03.04.03-RC5)
- ☑ PN7462AU_ex_phExDoorAccess@03.04.03-RC5 (PN7462AU_ex_phExDoorAccess@03.04.03-RC5)
- ☑ PN7462AU_ex_phExDoorAccessLCD@03.04.03-RC5 (PN7462AU_ex_phExDoorAccessLCD@03.04.03-RC5)
- ☑ PN7462AU_ex_phExEMVCo@03.04.03-RC5 (PN7462AU_ex_phExEMVCo@03.04.03-RC5)
- ☑ PN7462AU_ex_phExHIF@03.04.03-RC5 (PN7462AU_ex_phExHIF@03.04.03-RC5)
- ☑ PN7462AU_ex_phExHPP@03.04.03-RC5 (PN7462AU_ex_phExHPP@03.04.03-RC5)
- ☑ PN7462AU_ex_phExI2CP@03.04.03-RC5 (PN7462AU_ex_phExI2CP@03.04.03-RC5)
- ☑ PN7462AU_ex_phExMain@03.04.03-RC5 (PN7462AU_ex_phExMain@03.04.03-RC5)
- ☑ PN7462AU_ex_phExMfCryptoEC@03.04.03-RC5 (PN7462AU_ex_phExMfCryptoEC@03.04.03-RC5)
- ☑ PN7462AU_ex_phExNFCCcid@03.04.03-RC5 (PN7462AU_ex_phExNFCCcid@03.04.03-RC5)
- ☑ PN7462AU_ex_phExNFCForum@03.04.03-RC5 (PN7462AU_ex_phExNFCForum@03.04.03-RC5)
- ☑ PN7462AU_ex_phExPos@03.04.03-RC5 (PN7462AU_ex_phExPos@03.04.03-RC5)
- ☑ PN7462AU_ex_phExRHPP@03.04.03-RC5 (PN7462AU_ex_phExRHPP@03.04.03-RC5)
- ☑ PN7462AU_ex_phExRfPCTA@03.04.03-RC5 (PN7462AU_ex_phExRfPCTA@03.04.03-RC5)
- ☑ PN7462AU_ex_phExRfPoll@03.04.03-RC5 (PN7462AU_ex_phExRfPoll@03.04.03-RC5)
- ☑ PN7462AU_ex_phExSystemServices@03.04.03-RC5 (PN7462AU_ex_phExSystemServices@03.04.03-RC5)

Options

☑ Copy projects into workspace

Working sets

□ Add project to working sets

Working sets:

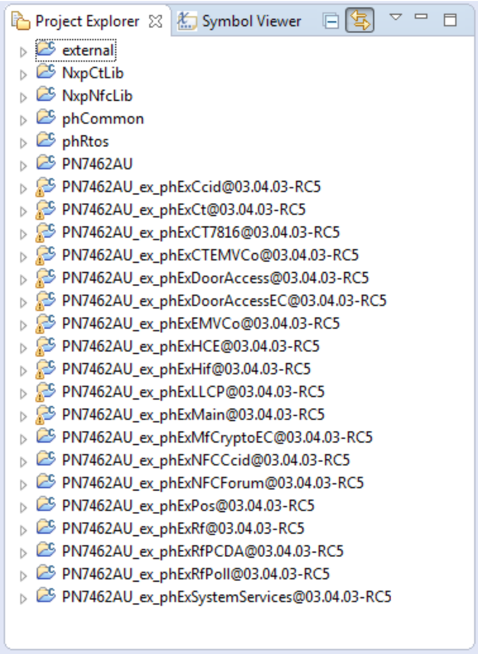
Fig 34. Select project

Select projects to be imported and then click “Finish”. Selected examples will be imported to the workspace.

To import only one examples, it is mandatory to import also projects in the list:

- external
- NxpCtLib
- NxpNfcLib
- phCommon
- phRtos

When the import process is finished, you can start with the development and editing the code.



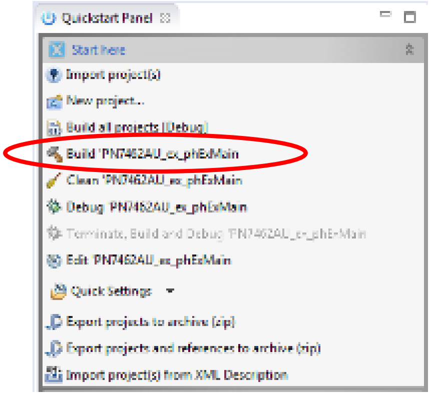
external
NxpCtLib
NxpNfcLib
phCommon
phRtos
PN7462AU
PN7462AU_ex_phExCcid@03.04.03-RC5
PN7462AU_ex_phExCt@03.04.03-RC5
PN7462AU_ex_phExCT7816@03.04.03-RC5
PN7462AU_ex_phExCTEMVCo@03.04.03-RC5
PN7462AU_ex_phExDoorAccess@03.04.03-RC5
PN7462AU_ex_phExDoorAccessEC@03.04.03-RC5
PN7462AU_ex_phExEMVCo@03.04.03-RC5
PN7462AU_ex_phExHCE@03.04.03-RC5
PN7462AU_ex_phExHif@03.04.03-RC5
PN7462AU_ex_phExLLCP@03.04.03-RC5
PN7462AU_ex_phExMain@03.04.03-RC5
PN7462AU_ex_phExMfCryptoEC@03.04.03-RC5
PN7462AU_ex_phExNFCCcid@03.04.03-RC5
PN7462AU_ex_phExNFCForum@03.04.03-RC5
PN7462AU_ex_phExPos@03.04.03-RC5
PN7462AU_ex_phExRf@03.04.03-RC5
PN7462AU_ex_phExRfPCDA@03.04.03-RC5
PN7462AU_ex_phExRfPoll@03.04.03-RC5
PN7462AU_ex_phExSystemServices@03.04.03-RC5

Fig 35. Project Workspace will all examples

5.5 Building projects

Building projects in a workspace is a simple case of using the Quick start Panel - ‘Build all projects’. Alternatively a single project can be selected in the “Project Explorer View” and built separately. Note that building a single project may also trigger a build of any associated library projects.

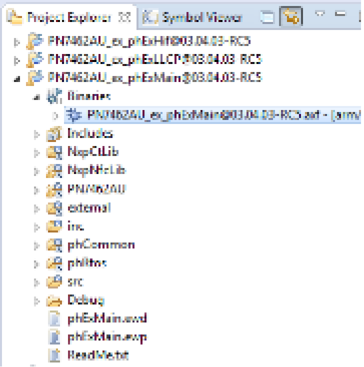
The project can be built as shown in the Fig 36.



Quickstart Panel
Start here
Import project(s)
New project...
Build all projects [Debug]
Build 'PN7462AU_ex_phExMain
Clean 'PN7462AU_ex_phExMain
Debug 'PN7462AU_ex_phExMain
Terminate, Build and Debug 'PN7462AU_ex_phExMain
Edit 'PN7462AU_ex_phExMain
Quick Settings
Export projects to archive (zip)
Export projects and references to archive (zip)
Import project(s) from XML Description

Fig 36. Building a project

As a part of the build output, the binary file for Flash is created. This binary file can be used to update PN7462AU Flash via USB mass storage interface or by using Flash tool or debug in LPCXpresso IDE. In case that “Binaries” folder is not visible in the project structure, refresh the project (right click on project and select “Refresh”).



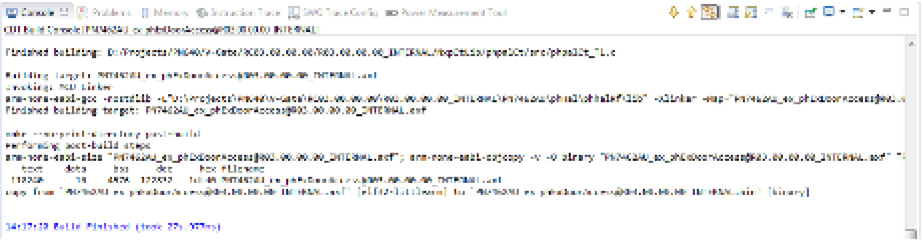
```
▹ PN7462AU_ex_phExHif@03.04.03-RC5
▹ PN7462AU_ex_phExLLCP@03.04.03-RC5
◢ PN7462AU_ex_phExMain@03.04.03-RC5
  ◢ Binaries
    ▹ PN7462AU_ex_phExMain@03.04.03-RC5.axf - [arm/
  ▹ Includes
  ▹ NxpCtLib
  ▹ NxpNfcLib
  ▹ PN7462AU
  ▹ external
  ▹ inc
  ▹ phCommon
  ▹ phRtos
  ▹ src
  ▹ Debug
    phExMain.ewd
    phExMain.ewp
    ReadMe.txt
```

Fig 37.   Build output - flash binary file

The project settings, compiler and link flags can be changed in the project properties dialog. To open the project properties dialog, select appropriate project in the “Project Explorer View” and click “Edit ‘selected-project’ project settings”.

Note:
Due to legal issues, NXP cannot provide source code for the FreeRTOS and the FreeRTOS source code should be integrated by users. Detailed description on how to integrate source code of the FreeRTOS to the project it is described in the “AN11784 PN7462AU How to integrate RTOS” document.

After successful integration of the FreeRTOS source code, project build should be compiled without errors as shown on the picture below.



```
CDT Build Console [PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL]

Finished building: D:/Projects/[illegible]/NxpCtLib/[illegible]/src/phhalCt_[illegible].c

Building target: PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.axf
Invoking: MCU Linker
arm-none-eabi-gcc -nostdlib -L"D:\Projects\[illegible]" -Xlinker -Map="PN7462AU_ex_phExDoorAccess@[illegible]
Finished building target: PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.axf

make --no-print-directory post-build
Performing post-build steps
arm-none-eabi-size "PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.axf"; arm-none-eabi-objcopy -v -O binary "PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.axf" "
   text    data     bss     dec     hex filename
 [illegible]      [illegible]    [illegible]  [illegible]   [illegible] PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.axf
copy from `PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.axf' [elf32-littlearm] to `PN7462AU_ex_phExDoorAccess@[illegible]_INTERNAL.bin' [binary]


[illegible] Build Finished (took [illegible])
```

Fig 38.   Successful build

5.6 Running and debugging a project

This description shows how to run the PN7462AU “PN7462AU_ex_*phExMain*” example application for the PN7462AU customer development board. The same basic principles will apply for all other examples. In cases where example will need additional configuration this will be detailed described in the example description.

First of all you need to ensure that your PN7462AU customer board is connected to your computer via LPC-LINK2, as shown in Fig 39.

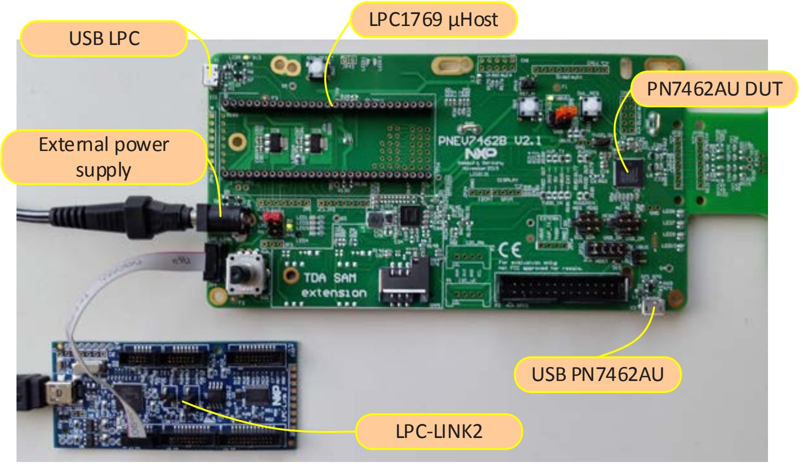


Fig 39. Important PINs on the PN7462AU Customer Board 2.1

When debug is started, the application is automatically downloaded to the target and it´s programmed to the FLASH memory; a default breakpoint is set on the first instruction in main (), the application is started (by simulating a processor reset), and code is executed until the default breakpoint is hit.

To start debugging your application on the PN7462AU, simply highlight the project in the Project Explorer and then in the Quick start Panel click Debug, as shown in Fig 40. The LPCXpresso IDE will first build application and then start debugging.

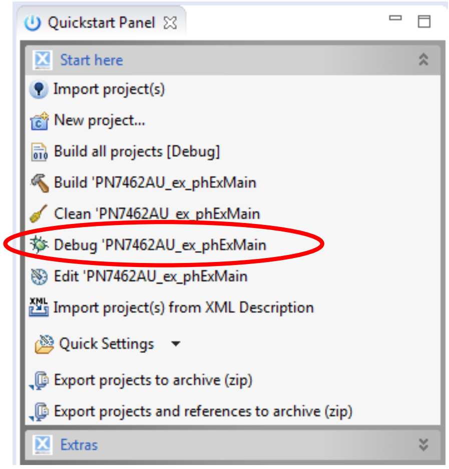
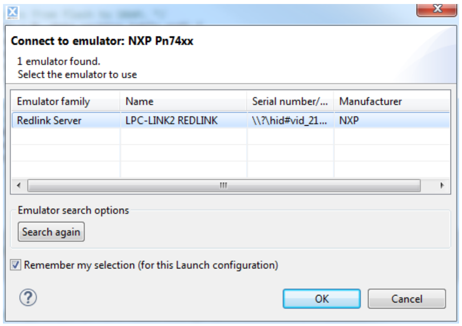

Quickstart Panel
Start here
Import project(s)
New project...
Build all projects [Debug]
Build 'PN7462AU_ex_phExMain
Clean 'PN7462AU_ex_phExMain
Debug 'PN7462AU_ex_phExMain
Edit 'PN7462AU_ex_phExMain
Import project(s) from XML Description
Quick Settings
Export projects to archive (zip)
Export projects and references to archive (zip)
Extras

Fig 40. Launch debug session



Connect to emulator: NXP Pn74xx

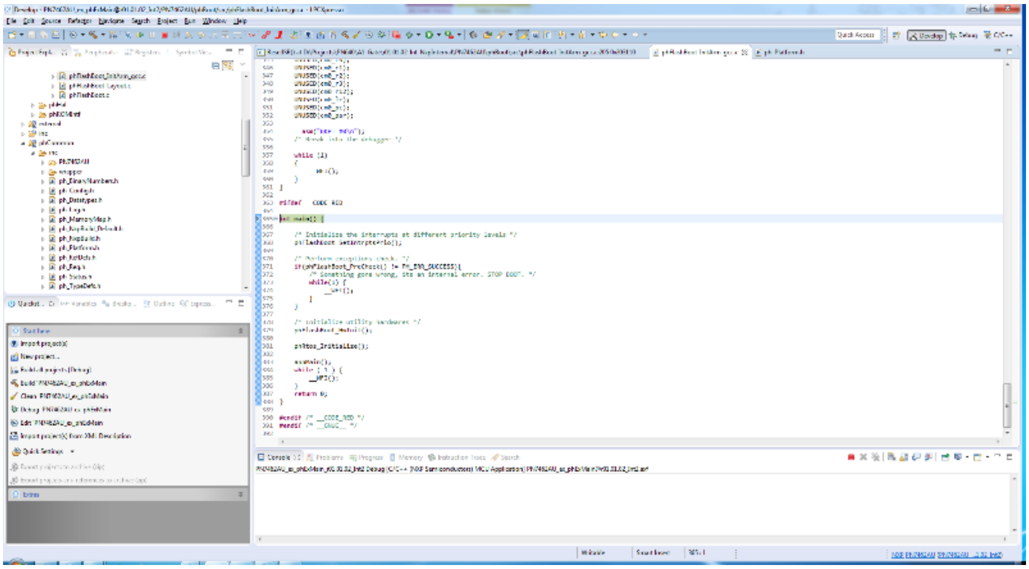
1 emulator found.
Select the emulator to use

Emulator family	Name	Serial number/...	Manufacturer
Redlink Server	LPC-LINK2 REDLINK	\\?\hid#vid_21...	NXP

Emulator search options

☑ Remember my selection (for this Launch configuration)

Fig 41. Select the launch configuration



```
    UNUSED(cm0_r1);
    UNUSED(cm0_r2);
    UNUSED(cm0_r3);
    UNUSED(cm0_r12);
    UNUSED(cm0_lr);
    UNUSED(cm0_pc);
    UNUSED(cm0_psr);

      asm("bkpt #0\n");
    /* Break into the debugger */

    while (1)
    {
        __WFI();
    }
}

#ifdef   CODE_RED

int main() {

    /* Initialize the interrupts at different priority levels */
    phFlashBoot_SetIntrptsPrio();

    /* Perform exceptions check. */
    if(phFlashBoot_PreCheck() != PH_ERR_SUCCESS){
        /* Something gone wrong, its an internal error. STOP BOOT. */
        while(1) {
            __WFI();
        }
    }

    /* initialize utility hardware */
    phFlashBoot_HwInit();

    phRtos_Initialize();

    asmain();
    while ( 1 ) {
        __WFI();
    }
    return 0;
}

#endif /* __CODE_RED */
#endif /* __GNUC__ */
```

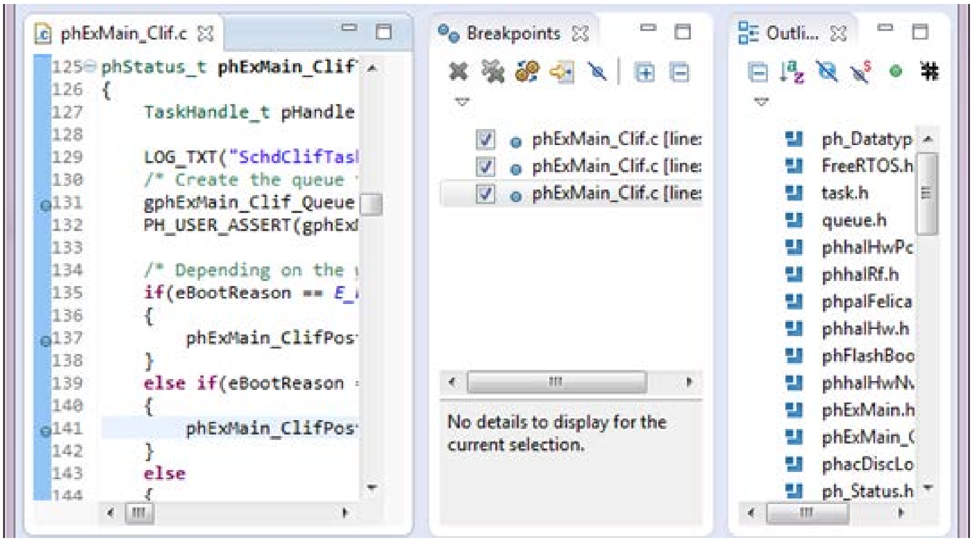
PN7462AU_ex_phExMain_v03.01.02_Int2 Debug [C/C++ (NXP Semiconductors) MCU Application] PN7462AU_ex_phExMain_v03.01.02_Int2.axf

Fig 42. Debug project

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

5.6.1 Break points

PN7462AU supports 4 breakpoints and 2 watch points. In usual way, double click on the left of the editor to set the break points. The execution of the application will be stopped when the break point is reached.



```
phStatus_t phExMain_Clif
{
    TaskHandle_t pHandle

    LOG_TXT("SchdClifTasl
    /* Create the queue
    gphExMain_Clif_Queue
    PH_USER_ASSERT(gphExl

    /* Depending on the
    if(eBootReason == E_
    {
        phExMain_ClifPos
    }
    else if(eBootReason
    {
        phExMain_ClifPos
    }
    else
    {
```

Breakpoints

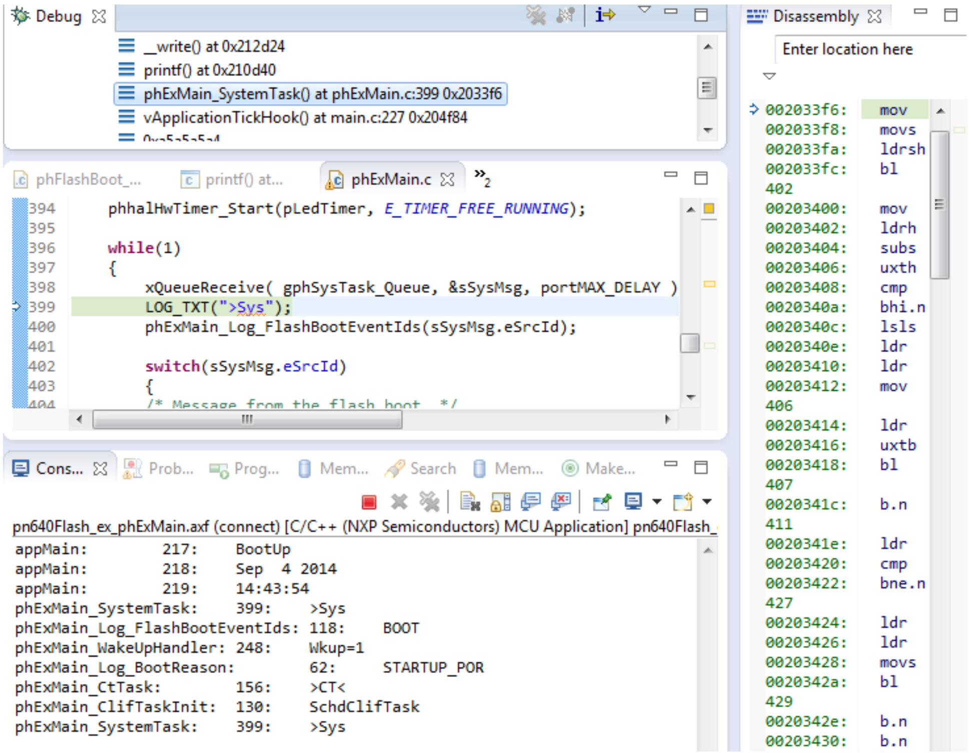
☑ phExMain_Clif.c [line:
☑ phExMain_Clif.c [line:
☑ phExMain_Clif.c [line:

No details to display for the current selection.

Fig 43. Inserting breakpoints

5.6.2 Debug traces

The debug traces can be seen on console as shown in Fig 44.



```
_write() at 0x212d24
printf() at 0x210d40
phExMain_SystemTask() at phExMain.c:399 0x2033f6
vApplicationTickHook() at main.c:227 0x204f84
0xa5a5a5a4
```

```
    phhalHwTimer_Start(pLedTimer, E_TIMER_FREE_RUNNING);

    while(1)
    {
        xQueueReceive( gphSysTask_Queue, &sSysMsg, portMAX_DELAY )
        LOG_TXT(">Sys");
        phExMain_Log_FlashBootEventIds(sSysMsg.eSrcId);

        switch(sSysMsg.eSrcId)
        {
        /* Message from the flash boot  */
```

pn640Flash_ex_phExMain.axf (connect) [C/C++ (NXP Semiconductors) MCU Application] pn640Flash_

```
appMain:          217:    BootUp
appMain:          218:    Sep  4 2014
appMain:          219:    14:43:54
phExMain_SystemTask:    399:    >Sys
phExMain_Log_FlashBootEventIds: 118:    BOOT
phExMain_WakeUpHandler: 248:    Wkup=1
phExMain_Log_BootReason:        62:     STARTUP_POR
phExMain_CtTask:        156:    >CT<
phExMain_ClifTaskInit:  130:    SchdClifTask
phExMain_SystemTask:    399:    >Sys
```

```
002033f6:   mov
002033f8:   movs
002033fa:   ldrsh
002033fc:   bl
402
00203400:   mov
00203402:   ldrh
00203404:   subs
00203406:   uxth
00203408:   cmp
0020340a:   bhi.n
0020340c:   lsls
0020340e:   ldr
00203410:   ldr
00203412:   mov
406
00203414:   ldr
00203416:   uxtb
00203418:   bl
407
0020341c:   b.n
411
0020341e:   ldr
00203420:   cmp
00203422:   bne.n
427
00203424:   ldr
00203426:   ldr
00203428:   movs
0020342a:   bl
429
0020342e:   b.n
00203430:   b.n
```

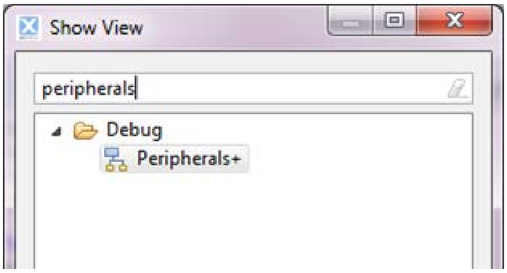
Fig 44. Debug view and debugger traces of PN7462AU

5.6.3 Peripheral view

After installing PN7462AU Plugin to the LPCXpresso, (See chapter 5.3), we get direct access to all the peripheral registers of the PN7462AU.

As shown below, we can see the fields and description of the EEPROM Controller on the PN7462AU IC.

To see the peripheral registers, follow the steps as shown below.



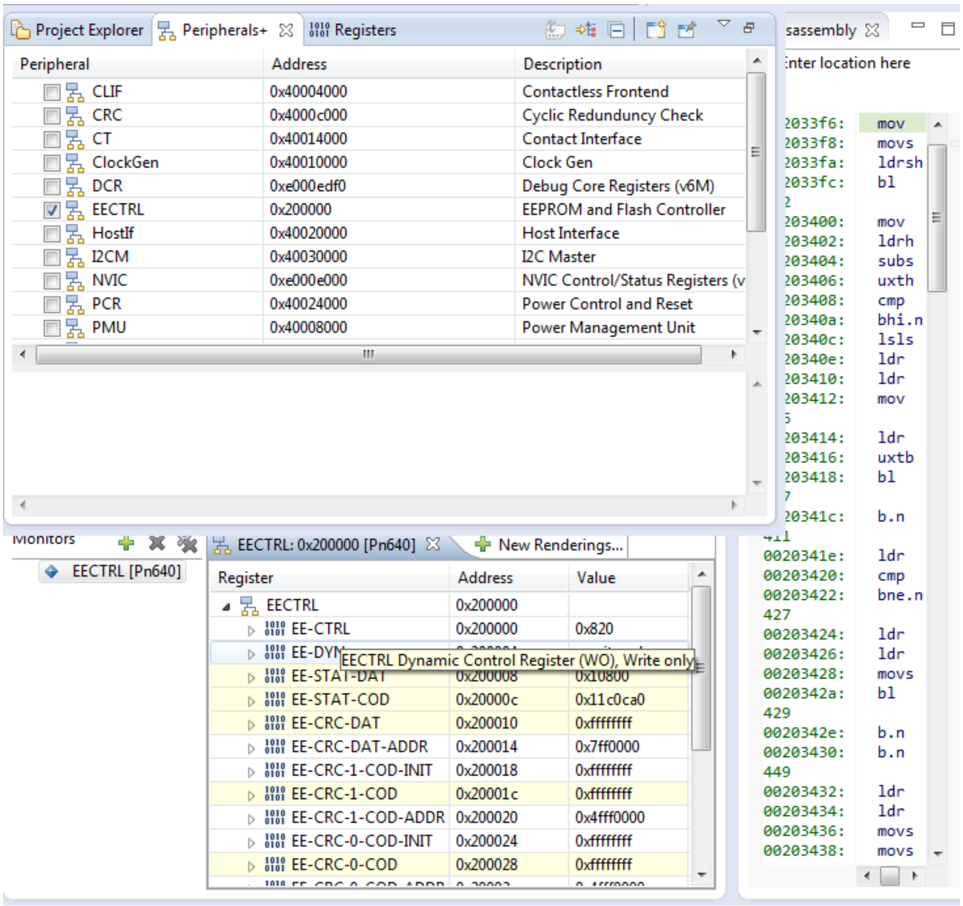
(1) Go to window → show view → other

(2) Select “Peripherals+”

Fig 45. Peripheral view

Select appropriate register or IP to watch or change the register values.

Note:
If CT IP is accessed without being enabled, PN7462AU crashes internally and hence cannot be used any further. To avoid falling into this trap/limitation never enable CT Peripheral register view on boot up. And enable only after the CT IP has been initialized during the code execution (i.e. after `phhalHwCt_Init()` is invoked).



Peripheral	Address	Description
□ CLIF	0x40004000	Contactless Frontend
□ CRC	0x4000c000	Cyclic Redundancy Check
□ CT	0x40014000	Contact Interface
□ ClockGen	0x40010000	Clock Gen
□ DCR	0xe000edf0	Debug Core Registers (v6M)
☑ EECTRL	0x200000	EEPROM and Flash Controller
□ HostIf	0x40020000	Host Interface
□ I2CM	0x40030000	I2C Master
□ NVIC	0xe000e000	NVIC Control/Status Registers (v
□ PCR	0x40024000	Power Control and Reset
□ PMU	0x40008000	Power Management Unit

Register	Address	Value
EECTRL	0x200000	
EE-CTRL	0x200000	0x820
EE-DYN	[illegible]	[illegible]
EE-STAT-DAT	0x200008	0x10800
EE-STAT-COD	0x20000c	0x11c0ca0
EE-CRC-DAT	0x200010	0xffffffff
EE-CRC-DAT-ADDR	0x200014	0x7ff0000
EE-CRC-1-COD-INIT	0x200018	0xffffffff
EE-CRC-1-COD	0x20001c	0xffffffff
EE-CRC-1-COD-ADDR	0x200020	0x4fff0000
EE-CRC-0-COD-INIT	0x200024	0xffffffff
EE-CRC-0-COD	0x200028	0xffffffff

EECTRL Dynamic Control Register (WO), Write only

```
2033f6:    mov
2033f8:    movs
2033fa:    ldrsh
2033fc:    bl
2
203400:    mov
203402:    ldrh
203404:    subs
203406:    uxth
203408:    cmp
20340a:    bhi.n
20340c:    lsls
20340e:    ldr
203410:    ldr
203412:    mov
5
203414:    ldr
203416:    uxtb
203418:    bl
7
20341c:    b.n
411
0020341e:  ldr
00203420:  cmp
00203422:  bne.n
427
00203424:  ldr
00203426:  ldr
00203428:  movs
0020342a:  bl
429
0020342e:  b.n
00203430:  b.n
449
00203432:  ldr
00203434:  ldr
00203436:  movs
00203438:  movs
```

(1) Select the relevant peripheral

(2) You will see the register, bit fields and description in the relevant view

Fig 46. Debug view and debugger traces of PN7462AU

6. Downloading, executing and debugging flash FW

The firmware on PN7462AU can be downloaded either Using USB Mass Storage mode or using LPC-LINK2 via SWD interface.

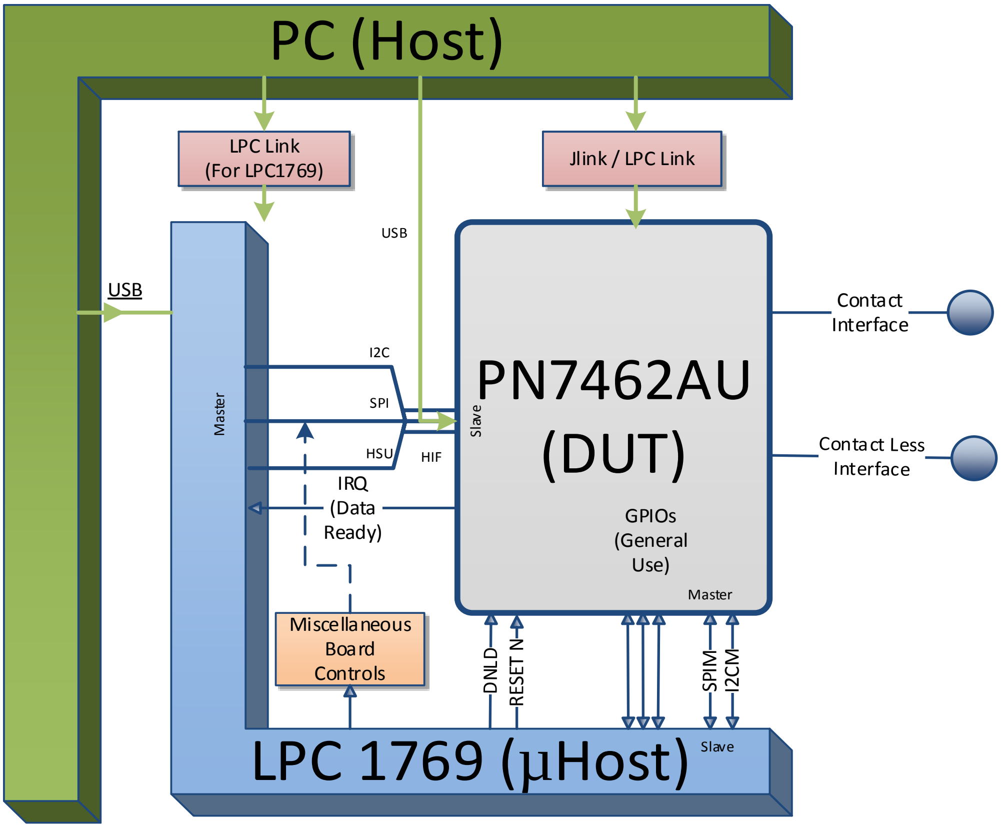


Fig 47. Overview of PN7462AU development environment

6.1 Downloading FLASH via LPC Link2

Ensure that LPC-LINK2 is connected to PN7462AU Fig 48 and follow the steps shown below.

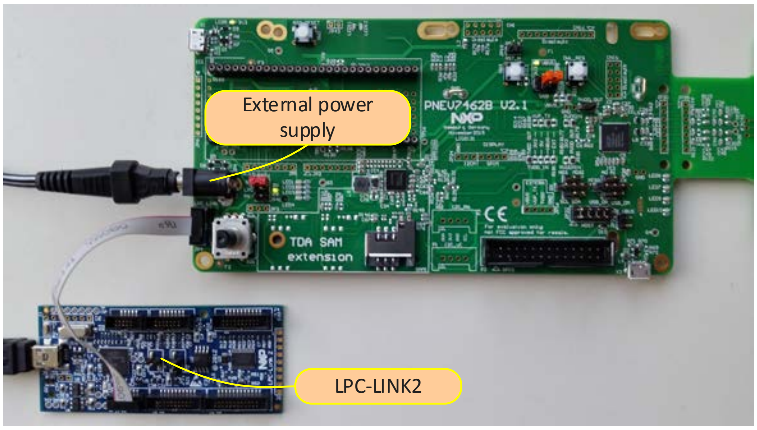
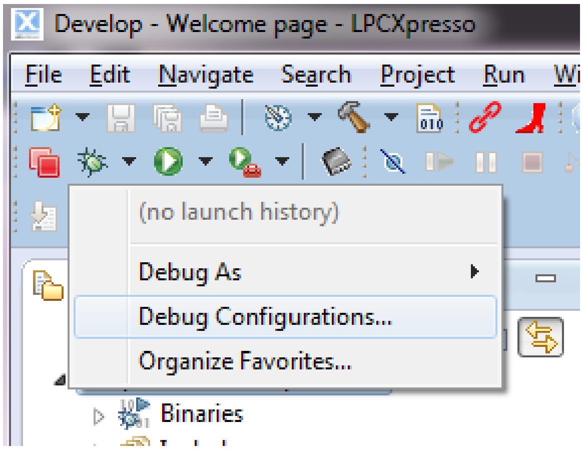
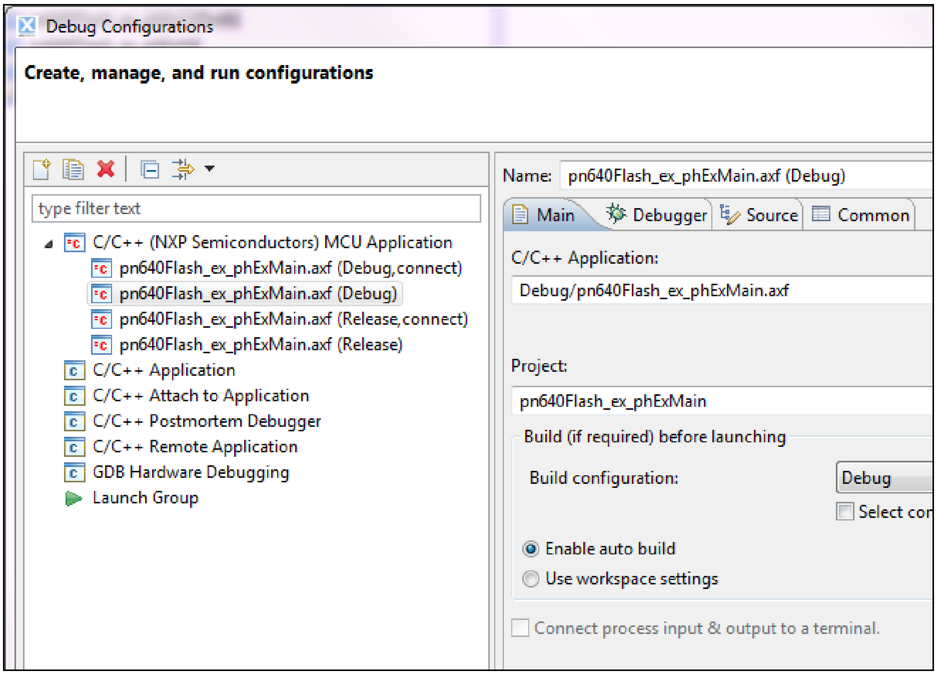


Fig 48. HW setup for the flashing via LPC-Link2



(1) For first time debug session, select Debug configurations

Fig 49. Selection debug configuration



Debug Configurations

Create, manage, and run configurations

type filter text

- C/C++ (NXP Semiconductors) MCU Application
  - pn640Flash_ex_phExMain.axf (Debug,connect)
  - pn640Flash_ex_phExMain.axf (Debug)
  - pn640Flash_ex_phExMain.axf (Release,connect)
  - pn640Flash_ex_phExMain.axf (Release)
- C/C++ Application
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- GDB Hardware Debugging
- Launch Group

Name: pn640Flash_ex_phExMain.axf (Debug)

Main | Debugger | Source | Common

C/C++ Application:
Debug/pn640Flash_ex_phExMain.axf

Project:
pn640Flash_ex_phExMain

Build (if required) before launching

Build configuration: Debug

□ Select cor

● Enable auto build
○ Use workspace settings

□ Connect process input & output to a terminal.

(1) * (Debug, connect) → Configuration to debug the Debug flavor of the Firmware but skip build/download

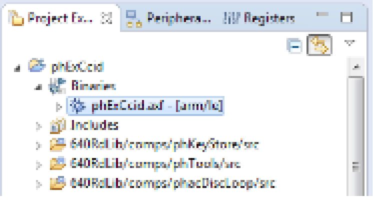
(2) * (Debug) → Configuration to build, download and debug the Debug flavor of the firmware

(3) * (Release, connect) → Configuration to debug the Release flavor of the Firmware but skip build/download

(4) * (Release) → Configuration to build, download and debug the Release flavor of the firmware

Fig 50. Debugging PN7462AU FW

As a part of the build output, the binary for Flash file is created. This binary file can be used to update PN7462AU Flash via USB mass storage interface or by using Flash tool.

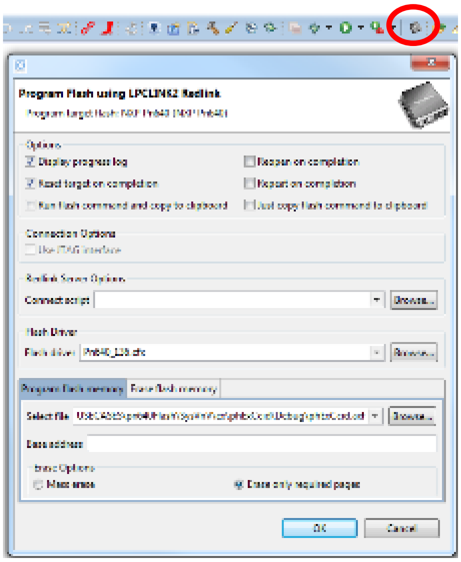


Project Ex... | Periphera... | Registers

- phExCcid
  - Binaries
    - phExCcid.axf - [arm/le]
  - Includes
  - 640RdLib/comps/phKeyStore/src
  - 640RdLib/comps/phTools/src
  - 640RdLib/comps/phacDiscLoop/src

(1) Select build output binary file

Fig 51. Build output



Program Flash using LPCLINK2 Redlink

Program target flash: NXP PN640 (NXP PN640)

Options

☑ Display progress log

☐ Reopen on completion

☑ Reset target on completion

☐ Repeat on completion

☐ Run flash command and copy to clipboard

☐ Just copy flash command to clipboard

Connection Options

☐ Use JTAG interface

Redlink Server Options

Connect script:

Flash Driver

Flash driver: PN640_128.cfx

Select file: [illegible]

Base address:

Erase Options

○ Mass erase

● Erase only required pages

(1) Select build output binary file

(2) Click on the “Program Flash Option”.

(3) Ensure that all the options are set properly and Click OK

Fig 52. Build output - flash binary file

6.2 Mounting PN7462AU as a USB mass storage device

PN7462AU can update FLASH contents via USB Mass Storage interface.

To mount PN7462AU IC as USB Mass storage drive (The steps mentioned here refer to the 2.4.3.1)

- Ensure that “HIF selection” is USB, see Fig 53
- USB Port of PC is connected to USB-PN74642AU
- Press “RST_N” switch.
- Press “DWL_REQ” switch.
- Release “RST_N” and keep holding “DWL_REQ”.
- Release “DWL_REQ” after some seconds.

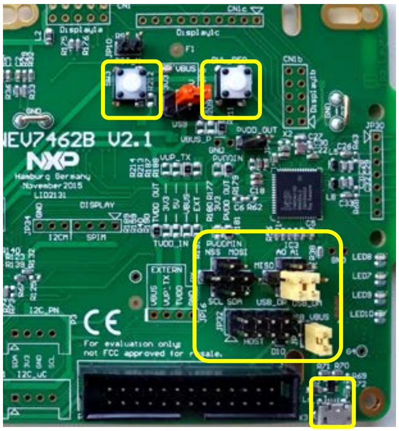
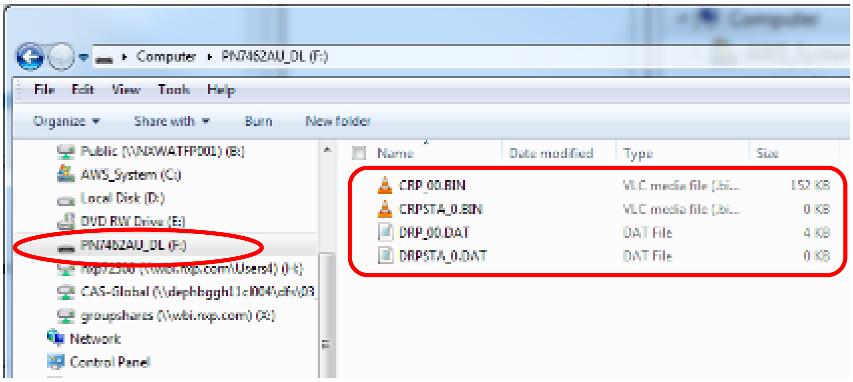


Fig 53. PN7462AU as USB mass storage device

You should see that a new Mass Storage device is detected by PC.


Computer ▸ PN7462AU_DL (F:)
File
Edit
View
Tools
Help
Organize ▾
Share with ▾
Burn
New folder
Public (\\NXWATFP001) (B:)
AWS_System (C:)
Local Disk (D:)
DVD RW Drive (E:)
PN7462AU_DL (F:)
nxp72300 (\\wbi.nxp.com\Users4) (H:)
CAS-Global (\\dephbgghl1c004\dfs\03
groupshares (\\wbi.nxp.com) (X:)
Network
Control Panel

Name	Date modified	Type	Size
CRP_00.BIN		VLC media file (.bi...	152 KB
CRPSTA_0.BIN		VLC media file (.bi...	0 KB
DRP_00.DAT		DAT File	4 KB
DRPSTA_0.DAT		DAT File	0 KB

Fig 54. PN7462AU IC detected as USB mass storage device for FW/ EEPROM upgrade

6.3 USB mass storage files

When the PN7462AU is mounted as a USB mass storage device files listed in the table below are visible in the device root folder.

Table 3. Files found in USB mass storage

File	Description
CRP_<nn>.BIN	PN7462AU´s user flash code (see Table 4 for description of <nn>)
CRPSTA_<s>.BIN	Status of previous write operation to user flash (see Table 5 for description of <s>)
DRP_<nn>.DAT	PN7462AU´s user EEPROM date (see Table 4 for description of <nn>)
DRPSTA_<s>.DAT	Status of previous write operation to user EEPROM (see Table 5 for description of <s>)

PN7462 USB mass storage supports various data/ code protection levels.

Table 4. Code and data protection level

<nn>	Description
00	Read and write allowed
01	Cannot read. Write allowed. Only applicable sectors erased before writing
02	Cannot read. All sectors of the applicable memory are erased before writing.
03	Cannot read. Cannot write via USB mass storage.

Table 5. Status of read write operating code

<s>	Description
0	Last write operation was successful
1	Memory region formatted
2 – or anything else	Failed
3	Fresh memory (FLASH/ EEPROM has never been downloaded via USB mass storage)

6.4 Updating user flash by USB

To update the user flash by USB, first of all the IC must to be in USB Mass storage mode. To enter to the USB Mass storage mode please follow steps described in chapter 6.2. After that follow the instructions listed below:

- Go to the newly mounted Drive.
- Delete CRP_<nn>.bin file
- Copy the Flash Binary file to the new Drive. As part of Firmware Build via LPCXpresso, the user Flash Binary is created. (See Fig 51).
- PN7462AU should automatically un-mount and re-enumerate itself. (In other words, the new drive automatically gets disconnected but re-connects/re-appears after some time).
- If the status file CRPSTA_00.bin is present, the download was successful.
- Press the “RST_N” switch.
- PN7462AU should be executing the new Flash code now.

7. Associated SW projects

7.1 General overview

For a detailed description of examples, please refer to the “UM10913 - PN7462AU Software User Manual”.

Note:

Due to legal issues, NXP cannot provide source code for the FreeRTOS and the FreeRTOS source code should be integrated by users. Detailed description on how to integrate source code of the FreeRTOS to the project it is described in the “AN11784 PN7462AU How to integrate RTOS” document.

7.1.1 Application messages – debug printouts

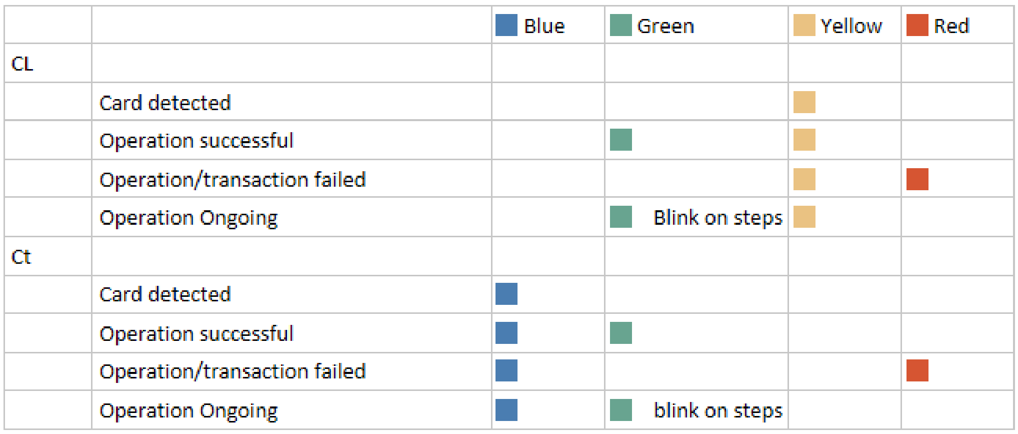
Examples support simple debug messages printouts via LPC-Link2 board. LPC-Link2 board needs to be connected to the customer board by SWD (Fig 48) and example must be in a debug build configuration.

Debug build configuration enables printout application messages. Printout messages are displayed in the Debug Messages Console View. Console view can be opened in menu “Window → Show View → Console” or by clicking the shortcut keys “Alt+Shift+Q C”.

To see application messages the application needs to be built in debug configuration and flashed via LPC-Link2 board.

7.1.2 LEDs status specifications

All example are prepared in way that LEDs on the board shows the current status of the running example. At the polling all LEDs are turning on in a circular sequence and when any card is detected the Fig 55 represent the meaning of the LED pattern.



		🟦 Blue	🟩 Green	🟨 Yellow	🟥 Red
CL					
	Card detected			🟨	
	Operation successful		🟩	🟨	
	Operation/transaction failed			🟨	🟥
	Operation Ongoing		🟩 Blink on steps	🟨	
Ct					
	Card detected	🟦			
	Operation successful	🟦	🟩		
	Operation/transaction failed	🟦			🟥
	Operation Ongoing	🟦	🟩 blink on steps		

Fig 55. LEDs status

7.2 phExMain – main example (CLIF + CTIF functionality)

The “phExMain” is an example which implements the polling for contact and contactless cards and it´s the right application to start working with the PN7462AU customer demo board. The “phExMain” is the root of many sub examples described below. For task and interface managing, the application can be configured to use FreeRTOS or not.

The example can switch between EMVCo polling loop and NFC Forum Mode Polling Loop via runtime flag and provide implementation for standby mode.

Application is based on the NFC Reader Library and CT Library.

7.2.1 Features

“*phExMain*” example is covering next features:

Table 6. “phExMain” Example features

Feature	supported
CLIF Interface	Yes
CT Interface	Yes
NXP NFC Reader Library	Yes
CT Reader Library	Yes
FreeRTOS	Yes
Non RTOS	Yes
Standby mode	Yes
HIF/MIF Interface	No

7.3 phExEMVCo example (CLIF + CTIF functionality)

The “*phExEMVCo*” is an example which implements the polling for the EMVCo contact and contactless cards and implement reference EMV transaction.

Application is based on the NFC Reader Library, CT Library and be run with or without FreeRTOS.

7.3.1 Features

“*phExEMVCo*” example is covering next features:

Table 7. “phExMain” Example features

Feature	supported
CLIF Interface	Yes
CT Interface	Yes
NXP NFC Reader Library	Yes
CT Reader Library	Yes
FreeRTOS	Yes
Non RTOS	Yes
Standby mode	No
HIF/MIF Interface	No

CLIF Features

- EMV Polling Loop
    - Passive A → Passive B
- EMV Transactions
    - SELECT (PPSE)
    - SELECT command
    - GET PROCESSING OPTIONS
    - READ RECORD
    - GENERATE AC

CT Features

- EMV Functionality
    - ATR Parsing accd EMV
    - Send Different AIDs to identify card
        - Master Card : Credit or Debit (tested)
        - Visa Card : Credit or Debit (tested)
        - Master Card : Maestro(debit card)
        - Master Card : Cirrus(interbank network)
        - Master Card : Maestro UK
        - Visa Card : Electron card
        - Visa Card : V PAY card
        - Visa Card : VISA Plus card
        - Amex Card (tested)
    - Class A (DCDC always in doubler mode)

7.4 phExRF example (CL functionality)

The “*phExRf*” is an example which implements the polling for contactless cards without NFC Reader Library support. Application use only HAL APIs and perform same CLIF functionality as 0 “*phExMain*” example with the only difference that for the transaction static predefined packets are used.

Application does not implement CT and “Standby” functionality and it is not based on the FreeRTOS.

7.4.1 Features

“*phExRF*” example is covering next features:

Table 8. “phExRf” Example features

Feature	supported
CLIF Interface	Yes
CT Interface	No
NXP NFC Reader Library	No
CT Reader Library	No
FreeRTOS	No
Non RTOS	Yes
Standby mode	No
HIF/MIF Interface	No

7.5 phExCT example (CT functionality)

This example implements simple polling for contact cards. Application use only HAL APIs and perform same CTIF functionality as 0 “*phExMain*” example with the only difference that for the transaction static predefined packets are used and example is not using any library.

Application does not implement CLIF and “Standby” functionality and it is not based on the FreeRTOS.

7.5.1 Features

“*phExCT*” example is covering next features:

Table 9. “phExRf” Example features

Feature	supported
CLIF Interface	No
CT Interface	Yes
NXP NFC Reader Library	No
CT Pal Library	No
FreeRTOS	No
Non RTOS	Yes
Standby mode	No
HIF/MIF Interface	No

7.6 phExCTEMVCo example (CT functionality)

The “*phExCtEMVCo*” is an example which implements the CT functionality with CT Pal library support. Application use CT Pal library APIs and perform same CT functionality as 0 “*phExEMVCo*” example with the only difference that for the transaction static predefined packets are used.

Application does not implement CLIF and “Standby” functionality and it is not based on the FreeRTOS.

7.7 phExCT7816 example (CT functionality)

The “*phExCt7816*” is an example which implements the CT functionality with CT library support. Application use CT library APIs and perform same CT functionality as 0 “*phExMain*” example with the only difference that for the transaction static predefined packets are used.

Application does not implement CLIF and “Standby” functionality and it is not based on the FreeRTOS.

7.8 phExHif example

This example is demonstrating the host interface loop back functionality for I2C, SPI, HSU and master interface functionality for I2CM, SPIM. Beside that it demonstrates secondary downloader functionality to EEPROM and FLASH memory over SPI Host interface. For Host Interface Frames “FREE Format” is used.

Application is implementing CT functionality with SPI Host interface and it is not based on the FreeRTOS.

Application consist from two projects. First application is executed on the PN7464AU and second application needs to be executed on the LPC1769 board.

7.9 phExHCE example

This example demonstrate how to use PN7462AU in Host Card Emulation mode (HCE). Purpose of the example is to show how to configure PN7462AU for HCE and to act as a contactless card.

When the NFC Device is tabbed application send a NDEF message to the device using LLCP/SNEP protocol. After a successful transfer www.nxp.com url shall be detected.

Application is based on the NFC Reader Library and use FreeRTOS to support HCE task. Example is emulating T4T.

7.10 phExLLCP example

This example is demonstrating Peer-To-Peer (P2P) communication between PN7462AU and NFC Device. Purpose of the example is to show how to use PN7462AU to exchange NDEF message over the P2P communication. Application can work in an Active Initiator or Active Target mode, depend on the discovered technologies.

7.11 POS use case demo application

POS use-case shows how to use PN7462AU in combination with second application hosted on the µController. In our case we will use LPC1769 µC and connection will be established through SPI host interface. POS use-case demonstrate the Pay pass transaction on the contact and contactless frontend.

Note

Detailed description and how to use example is described in “POS Use Case Demo Setup Manual”.

7.12 PC USB reader demo application

The PC USB reader application demonstrate how to use the PN7462AU Customer Demo board as a CCID reader and shows how connected PN7462AU via USB interface to a PC and provide the CCID protocol implementation on the top of the physical link.
The PC USB reader example is hosted on the PN7462AU and can be tested with any PC/SC application running on the PC with Windows OS.

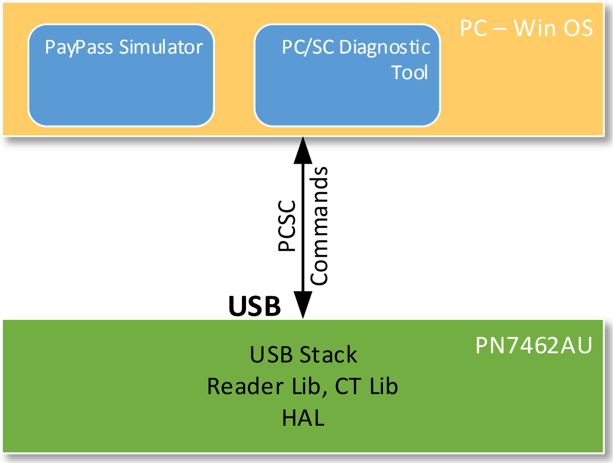


Fig 56. PC USB reader example block diagram

The USB stack and CCID class is implemented in the PN7462AU. The default CCID driver present in PC with Windows OS is used for operation.

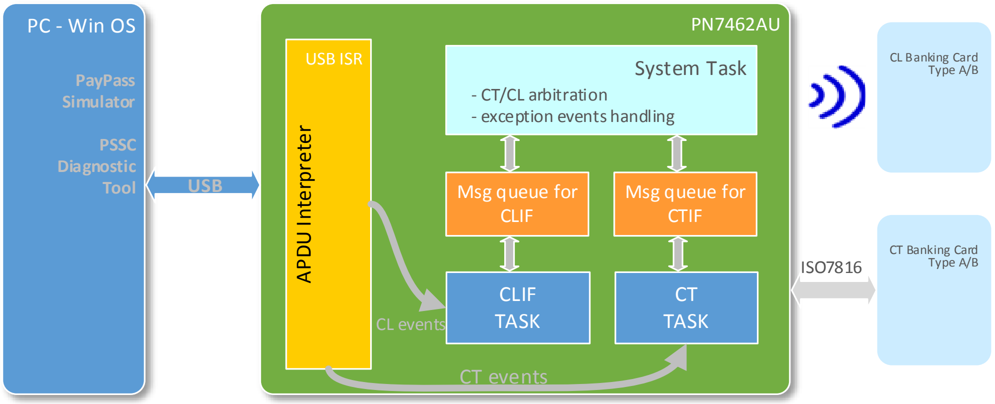


Fig 57. Example architecture

Note:
Detailed description and how to use example is described in “PC Reader Demo Setup Manual”.

8. References

[1] LPCXpresso webpage

www.nxp.com/redirect/lpcware.com/lpcxpresso/download

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 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 NFC technology

Purchase of an NXP Semiconductors IC that complies with one of the Near Field Communication (NFC) standards ISO/IEC 18092 and ISO/IEC 21481 does not convey an implied license under any patent right infringed by implementation of any of those standards. Purchase of NXP Semiconductors IC does not include a license to any NXP patent (or other IP right) covering combinations of those products with other products, whether hardware or software.

Purchase of NXP ICs with ISO 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 Semiconductors N.V.

I²C-bus logo — is a trademark of NXP Semiconductors N.V

ICODE and I•CODE — are trademarks of NXP Semiconductors N.V.

10. List of figures

11. List of tables

12. Contents

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

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

Date of release: 29 March 2016

319810

Document identifier: UM10883

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

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

Date of release: 29 March 2016
319810
Document identifier: UM10883