

# UM10883

## PN7462AU Quick Start Guide - Customer Board

Rev. 1.2 — 16 February 2017  
319812

User manual  
COMPANY PUBLIC

### Document information

Info	Content
<b>Keywords</b>	PN7462AU, Customer board, Quick Start Guide, functional description of the customer board
<b>Abstract</b>	This document describes the required basic circuitry to operate the PN7462AU and it also describes how to setup and use the PNEV7462B Customer Demo board



**Revision history**

Rev	Date	Description
1.2	20170216	PNEV7462B customer evalboard V2.2 added SW examples description updated. Guidelines how to upgrade firmware are updated. Figures updated
1.1	20161124	SW examples description updated. Guidelines how to import projects are updated. Figures updated
1.0	20160329	First release

**Contact information**

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

## 1. Introduction

---

This document describes the PNEV7462B (PN7462 evaluation board), which provides an easy evaluation of the features and functions of the PN7462.

It provides the first steps to operate the board, using the NFC Cockpit (PN7462 GUI Version 3.6.0) or software examples.

The default antenna is a 65mm x 65mm antenna with some metal layer inside the antenna area. This antenna is not an optimum antenna as such, but intends to demonstrate the performance and register settings of the PN7462 under typical design constraints like LCD or some metal (e.g. PCB) inside the antenna area.

### 1.1 PNEV7462B concept

The basic **concept of the PNEV7462B** is to enable the user to perform a quick evaluation of the PN7462, and also connect his own antenna to the PNEV7462 board. In addition, dedicated boards which allow to solder custom matching components are available. The NFC Cockpit can be used to optimize the PN7462 antenna tuning, to perform the DPC calibration and the related TX and Rx optimization without touching any source code.

All the relevant PN7462 registers can be modified and fine-tuned using the NFC Cockpit. After successful register optimization the found settings can be stored in the PN7462 EEPROM as well as user\_ee.bin. The user\_ee.bin is the binary file that contains all eeprom values and can be loaded using mass storage mode.

The NFC Cockpit also allows a dump of the complete user EEPROM content into an XML file. This file then can be loaded again into the EEPROM or the user\_ee.bin can be generated. That allows to manage and exchange different user or antenna configurations. In addition, the register settings found to work well using the NFC Cockpit, can be used during user code development as well.

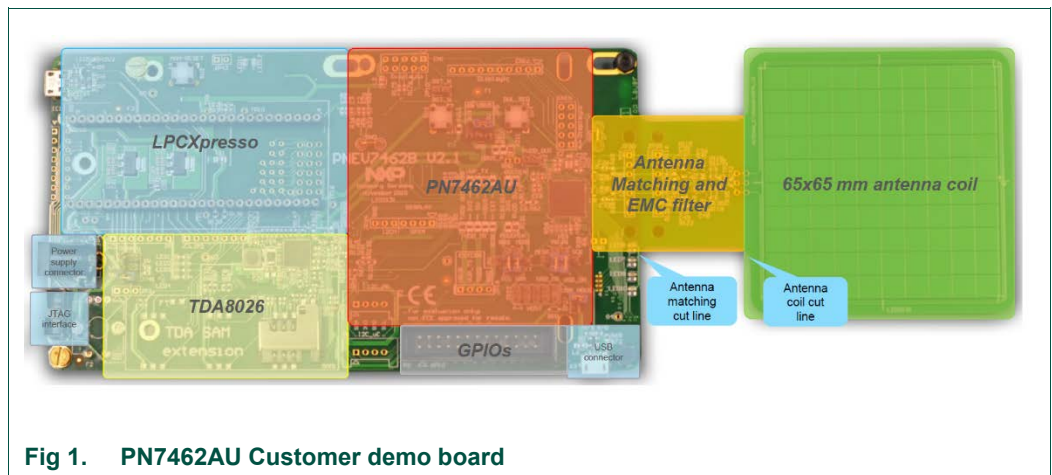
## 2. Hardware overview of the PN7462AU Customer Demo Board

The PN7462 is supplied with a voltage, which can be chosen between: internal and external supply. For the internal supply either 5V or 3.3V can be used. The external power supply can be an AC or DC supply (polarity does not matter) with at least 7.5V, since the board provide a rectifier and LDO to supply the circuit with 5V and 3.3V. In addition the board can be supplied via USB, which reduces the performance due to USB power restrictions.

### 2.1 PN7462AU Customer Demo Board

PN7462 Customer demo board (PNEV7462B) is development board for evaluating software and hardware features of the PN7462AU. It features easy application development with full NFC Forum compliant and contact software libraries.

The Fig 1 shows the PN7462AU Customer Demo board (PNEV7462B)



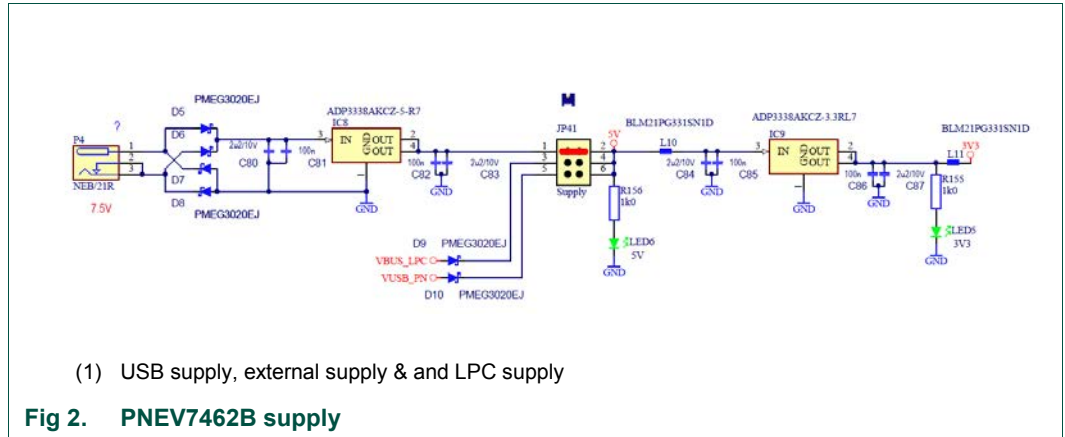
The board consists of four main blocks:

- PN7462AU (marked in red)
- LPCXpresso interface (marked in blue)
- TDA8026 (marked in yellow)
- Antenna coil and related matching circuit (marked in green and orange)

#### 2.1.1 Power supply

The default setting of the power supply is external power supply. For the maximum performance and a better test capability the external power supply should be connected. The AC or DC power input can cover any power supply providing an AC or DC voltage between 7.5 and 12V (PNEV7462B V2.2). For the development purposes the USB power supply can be used by setting jumper J41.

Note:  
 PN7462B v2.1: power supply 7.5V max.  
 PN7462B v2.2: power supply 12V max.



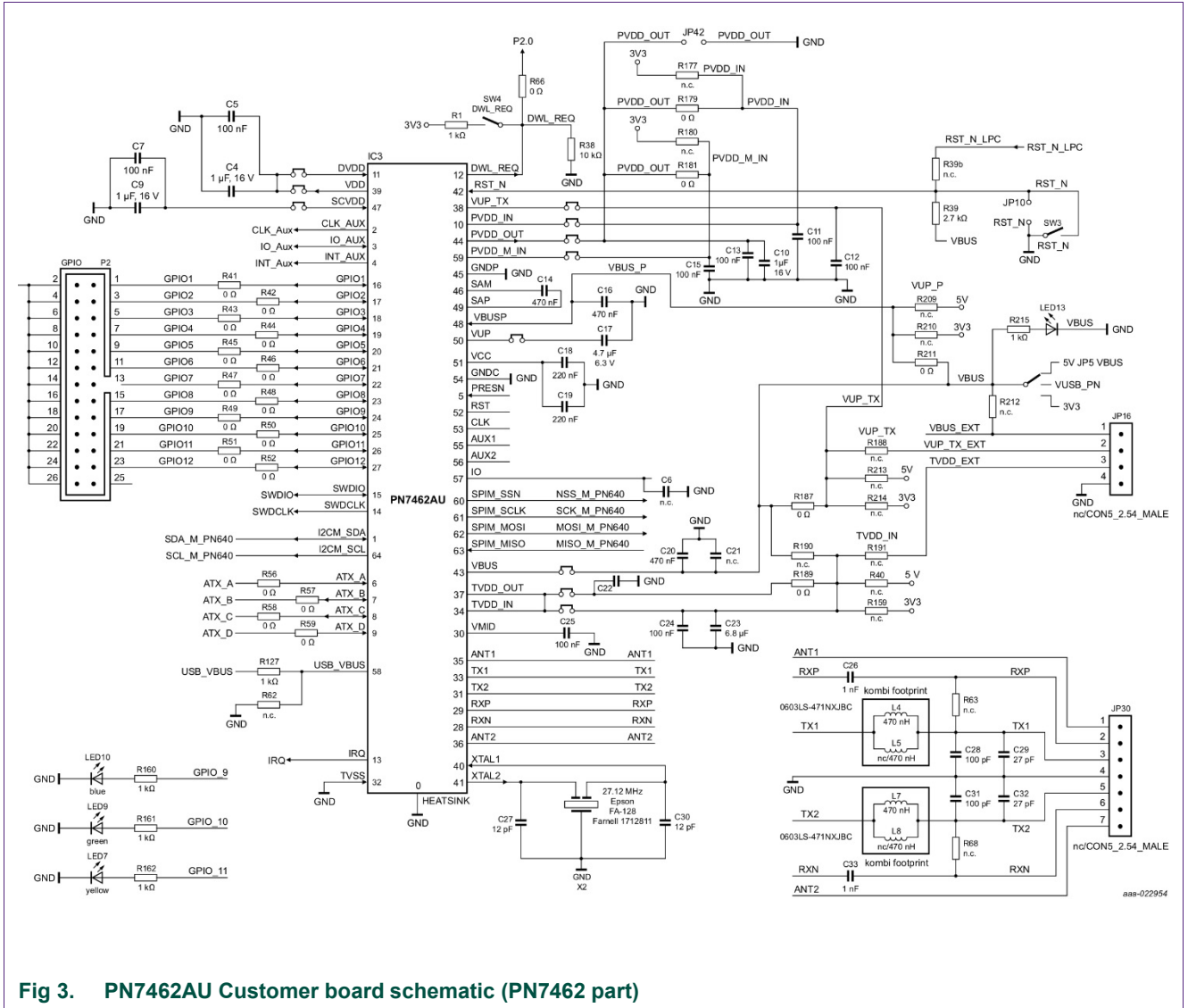
As soon as the board is supplied with power, the red LED6, LED5 and LED VBUS must be on. For more details on the supply options and settings see chapter 3.

### 2.1.2 PN7462AU block

PN7462AU is 32-bit ARM Cortex-M0-based NFC microcontroller offering a one chip solution to build contact and contactless applications.

The block features PN7462AU power supply settings, peripheral's interface headers and the host interface selection jumpers. PN7462UA chip board supply options are 3.3V or 5V and the selection is done through jumper settings as described in 3.1. One out of four host interface options can be configured as described in 3.2.

- Debug option
  - Serial Wire Debug (SWD) interface
- Peripherals
  - Host interface:
    - USB 2.0 full speed with USB 3.0 hub connection capability
    - HSUART for serial communication, supporting standards speeds from 9600 baud to 115200 baud, and faster speed up to 1.288 Mbit/s
    - SPI with half duplex and full duplex capability with speeds up to 7 Mbit/s
    - I2C supporting standard mode, fast mode and high-speed mode with multiple address support
  - Master interface:
    - SPI with half duplex capability from 1 Mbit/s to 6.78 Mbit/s
    - I2C supporting standard mode, fast mode, fast mode plus and clock stretching



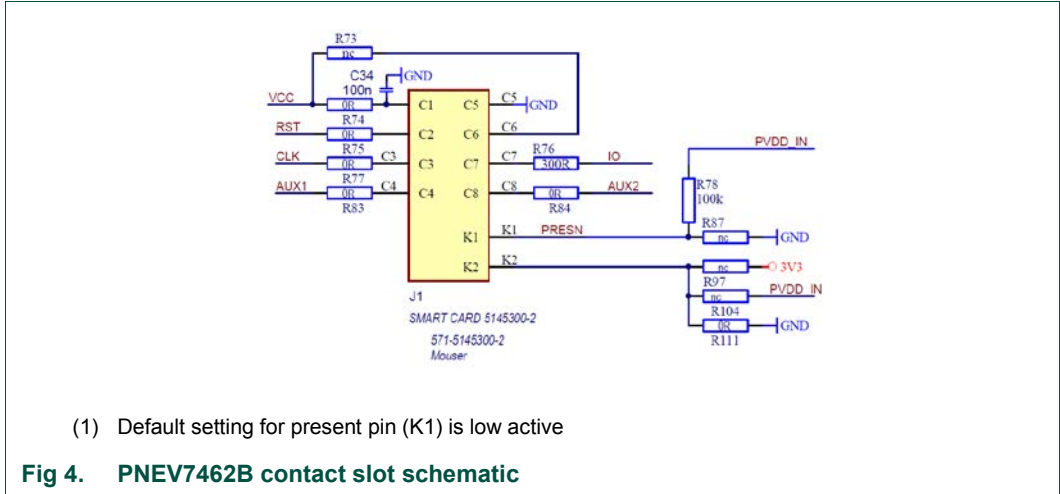
### 2.1.3 LPCXpresso block

This block of the PNEV7462B evaluation board () provides optional interface for the additional LPCXpresso board through the standard LPCXpresso/mbed expansion connector (2x27 pin). Onboard communication between the LPCXpresso MCU board and PN7462AU in this configuration is possible through the I2C or SPI interface.

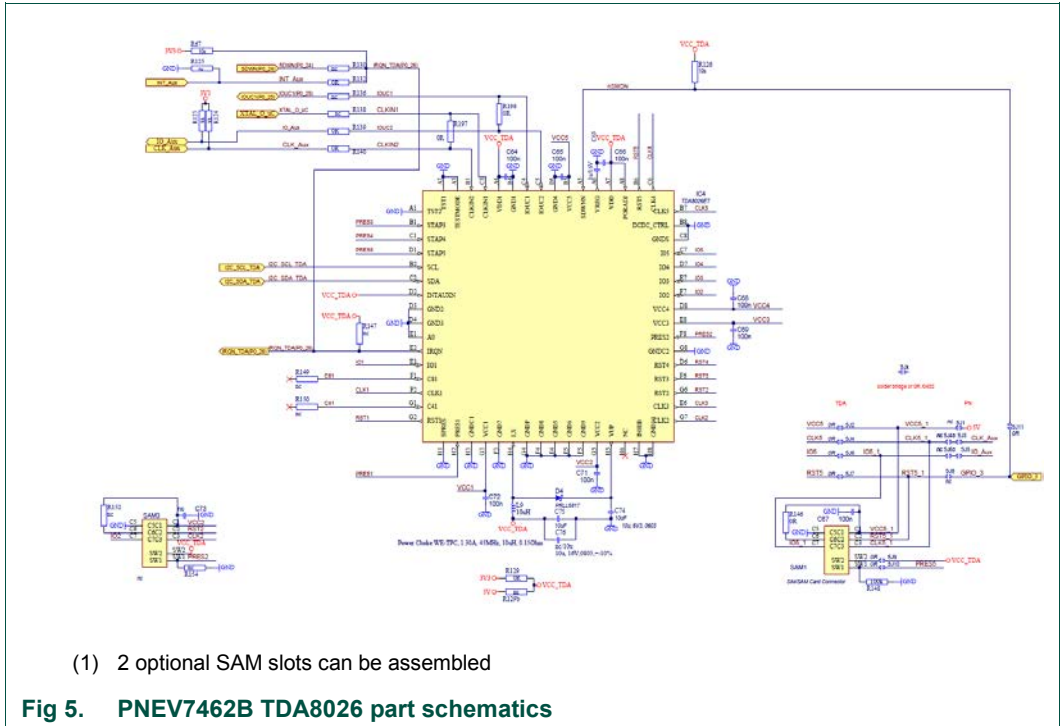
Block features LPCXpresso board USB interface and the LPC board reset circuit.

2.1.4 Smartcard reader and SAM slots extensions

This block provides a SAM slot connected to the TDA8026 (Fig 5) and the smartcard slot (bottom side) connected to the PN7462 contact pins (Fig 4).

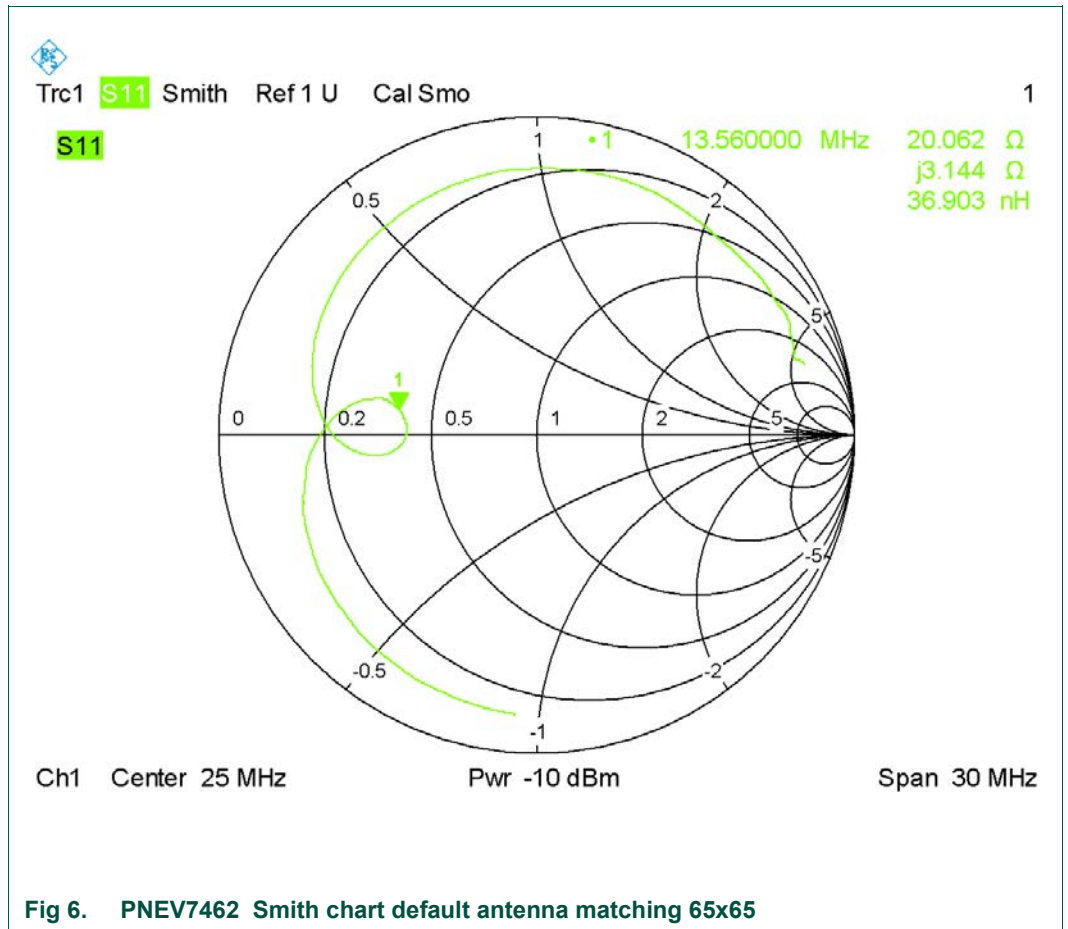


The TDA8026 is connected to the i2C master and AUX pins of the PN7462. So the TDA8026 can be configured using the PN7462.



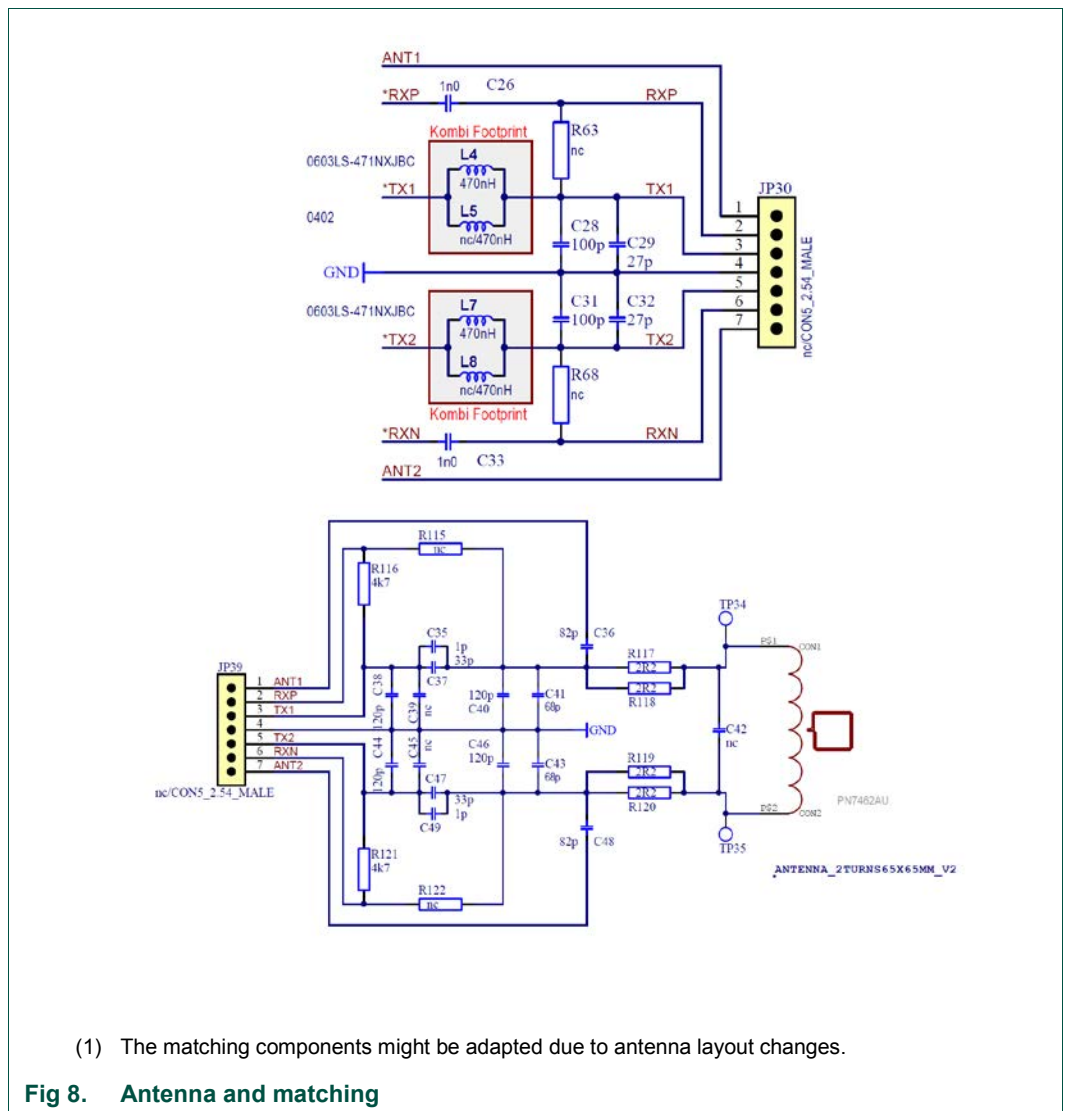
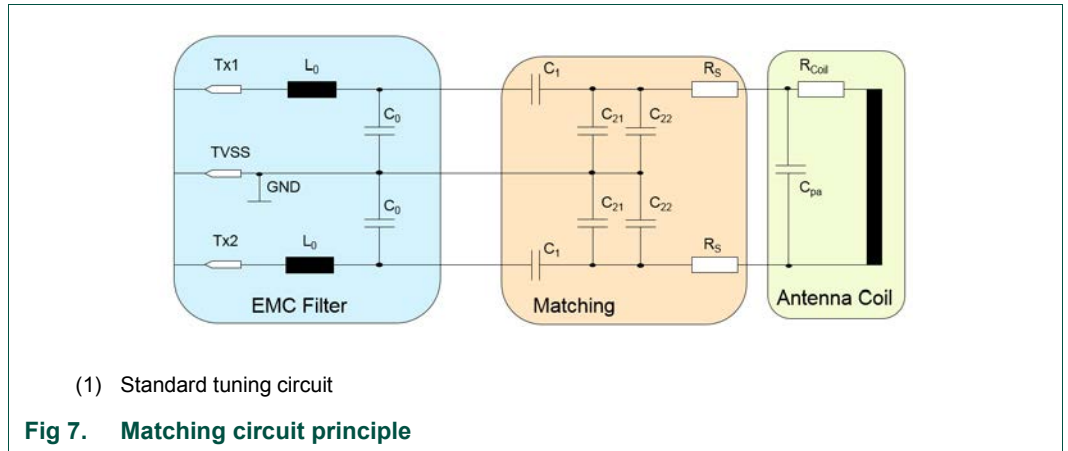
2.1.5 Antenna matching and EMC filter

The PNEV7462B provides a 65x65mm antenna with “symmetrical” tuning (see Fig 6).



The antenna connection uses the standard tuning circuit. The EMC filter is designed with a cut off frequency of  $f_{EMC} = 14,8$  MHz, and the antenna impedance is tuned to  $Z = 20\Omega$ .





In Table 1 the assembled components for the “symmetric” matching are listed.

**Table 1. Assembled matching components**

General component	Component PNEV7462B	Value	comment
L0	L4/ L7	470nH	<a href="#">PNEV7462B V2.1-&gt;0603LS-471NXJBC</a> <a href="#">PNEV7462B V2.2-&gt;36502AR47JTDG</a>
C0	C28/ C31	100pF	C0 split in 3 parallel capacitors
	C29/ C32	27pF	
	C38/ C44	120pF	
C1	C35/ C49	33pF	C1 split in 2 parallel capacitors
	C37/ C47	1pF	
C2 <sub>1</sub>	C40/ C46	120pF	
C2 <sub>2</sub>	C41/ C43	68pF	
R <sub>s</sub>	R117/ R119	2,2Ω	R <sub>s</sub> split in 2 parallel resistors
	R118/ R120	2,2Ω	

## 2.2 Customer demo board available versions

Following Versions of the PNEV7462B are available

- PNEV7462B V2.1
- PNEV7462B V2.2

### 2.2.1 PNEV7462B V2.1

The V2.1 of the customer evaluation board is the initial version of the board that comes with the launch of the PN7462 chip.

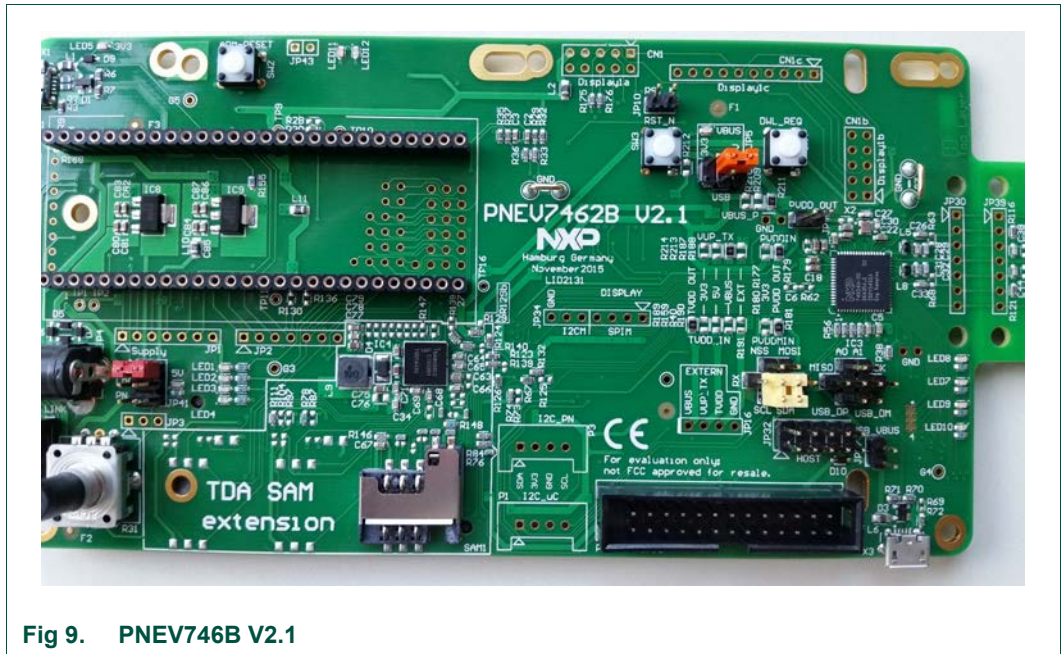


Fig 9. PNEV746B V2.1

### 2.2.2 PNEV7462B V2.2

The V2.2 of the customer evaluation board is the replacement and latest version of the customer evaluation board incl. FCC certification. Functionality of the V2.2 is the same as of V2.1.

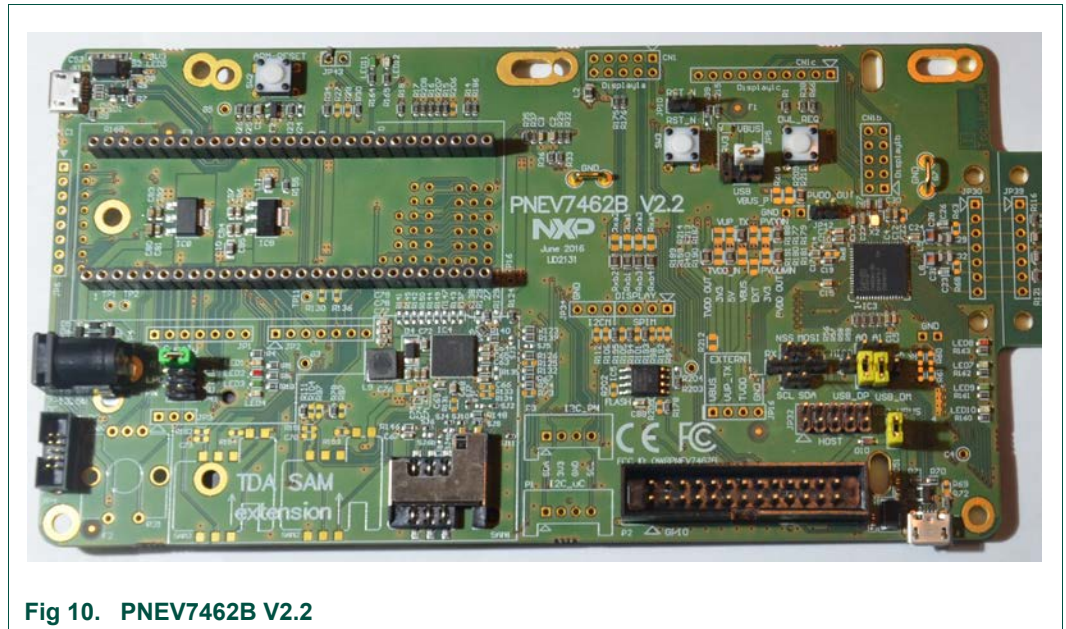


Fig 10. PNEV7462B V2.2

#### 2.2.2.1 Design changes V2.1 to V2.2:

- External supply from 7.5V to 12V
- Different routing (PNEV7462B V2.1 stays the board reference design which can be obtained from the NXP DocStore). Layout recommendations for NFC readers can be found in AN11090.
- Changed EMC filter components

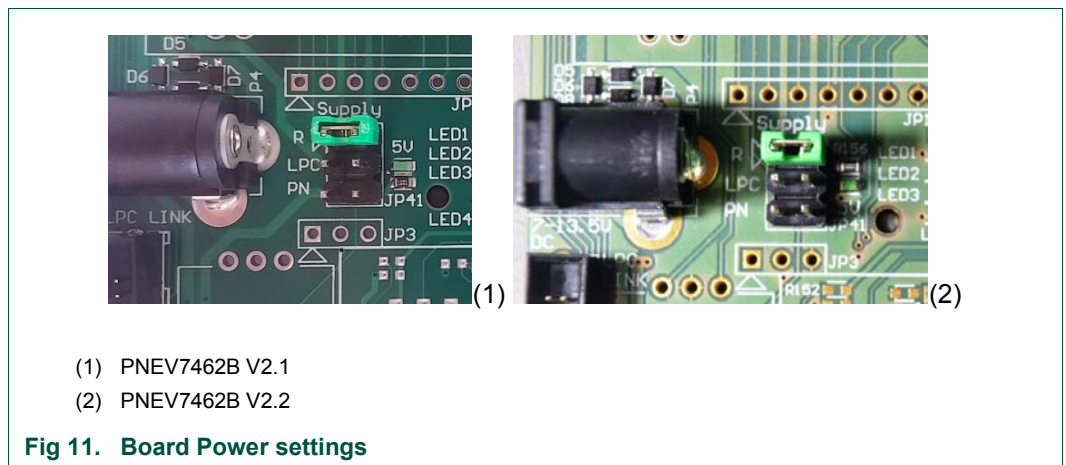
### 3. Configuration of the PN7462AU Customer board

#### 3.1 Board power settings

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

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

Jumper JP41 setting (Fig 2) needs to be done to prepare the board for one of the three supplies. (The recommended supply is the external power supply!)



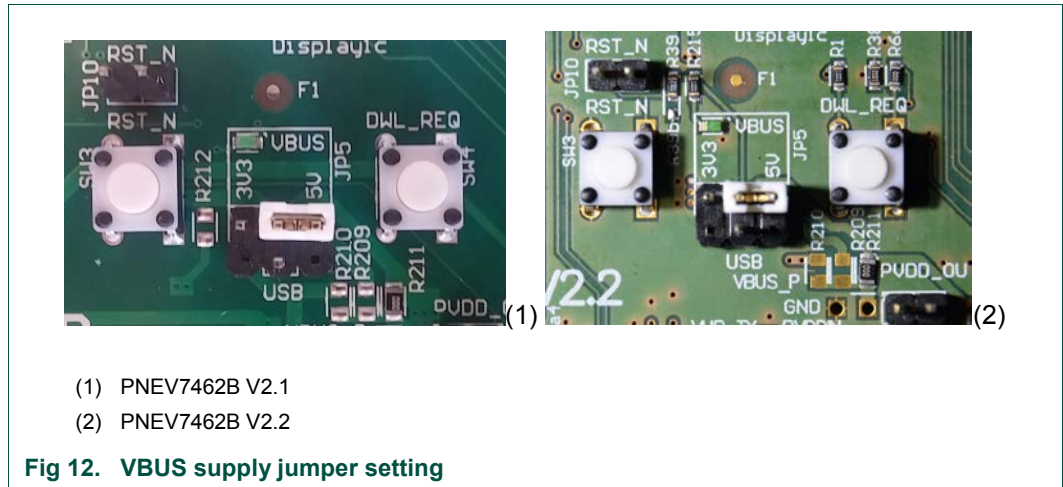
#### 3.1.1 PN7462AU supply options

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.

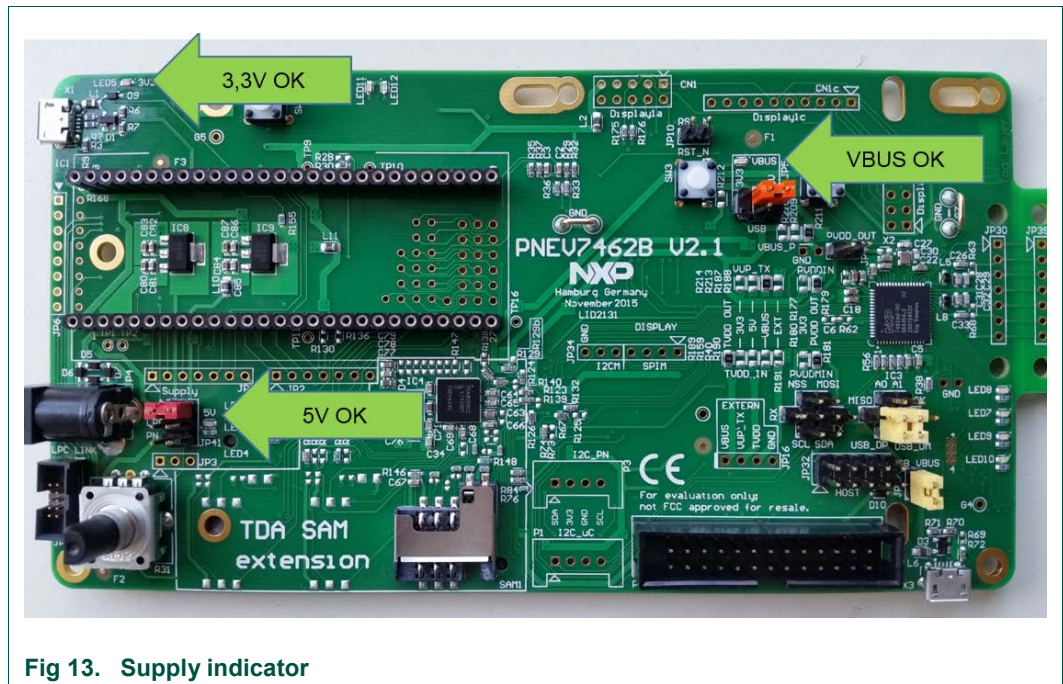
The supply of the main chip supply (VBUS) can be set to 5 V 3.3 V or USB supply

The corresponding setting is described in Fig 12



### 3.1.2 Power supply status LED

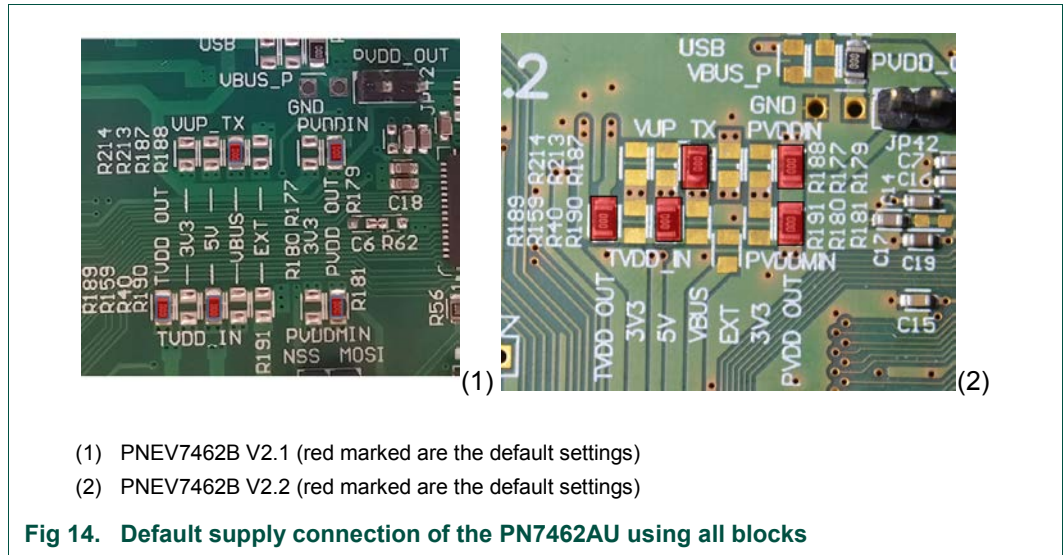
If all jumpers are set correctly Following LED`s should light green, 3V3, 5 V and VBUS. In Fig 13 the location of the three different LED`'s are shown.



### 3.1.3 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 14. Jumper JP42 needs to be open, to have PVDD\_OUT activated by PN7462AU.



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 14) needs to be set to the corresponding position. (default settings are marked in green):

**Table 2. 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.*

### 3.2 Host interface configuration

The PN7462AU supports interfacing one out of the four different host interfaces: USB 2.0 full speed with USB 3.0 hub connection capability, HSUART for serial communication, supporting standards speeds from 9600 baud to 115200 baud, and faster speed up to 1.288 Mbit/s, SPI with half duplex and full duplex capability with speeds up to 7 Mbit/s and I2C supporting standard mode, fast mode and high-speed mode with multiple address support

Interface selection is performed by appropriate jumper settings as follows.

#### 3.2.1.1 USB Host Interface configuration

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

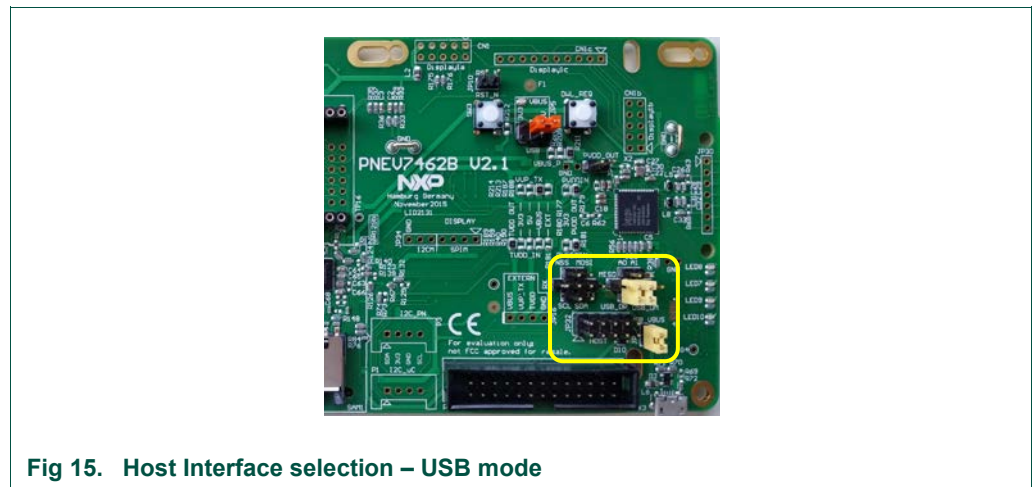


Fig 15. Host Interface selection – USB mode

#### 3.2.1.2 I2C Host Interface configuration

The yellow marked jumpers (Fig 16) needs to be set for using the I2C host interface of the chip. This will connect the I<sup>2</sup>C 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 board.



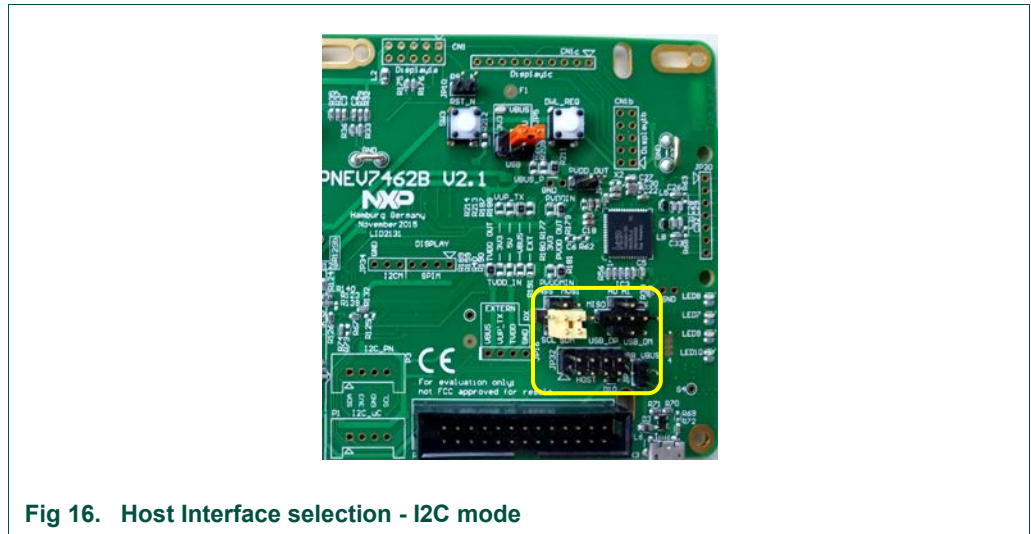


Fig 16. Host Interface selection - I2C mode

### 3.2.1.3 SPI Host Interface configuration

The yellow marked jumpers (Fig 17) 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 PN7462AU to the I/O P0(16) of the LPCXpresso board.

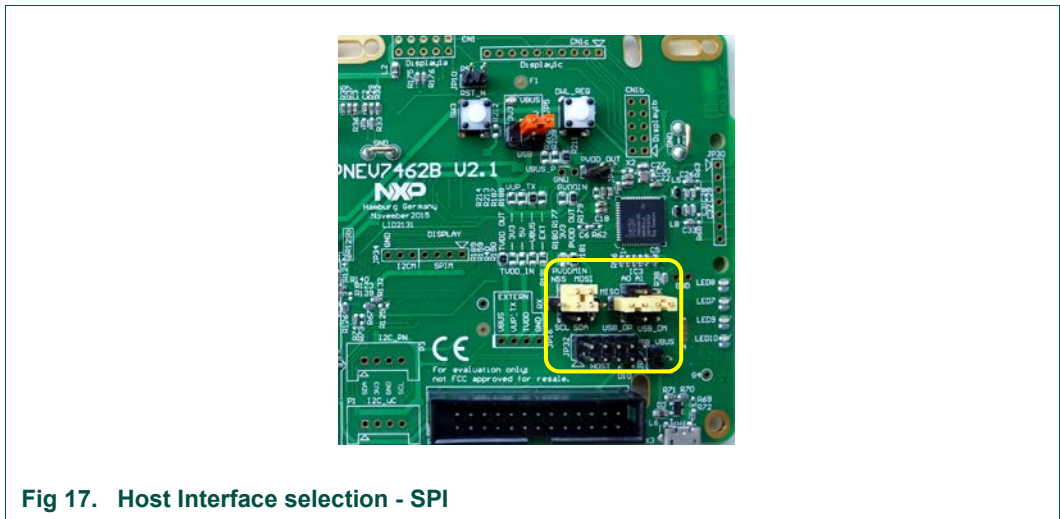


Fig 17. Host Interface selection - SPI

### 3.2.1.4 HSUART Interface configuration

The yellow marked jumpers (Fig 18) need to be set for using the HSUART host interface of the chip. This will connect the UART\_TX of the PN7462AU to the I/O P0(0), MISO to the I/O P0(17), SCK to the I/O P0(15), and also the UART\_RX of the PN7462AU to the I/O P0(1) of the LPCXpresso board.

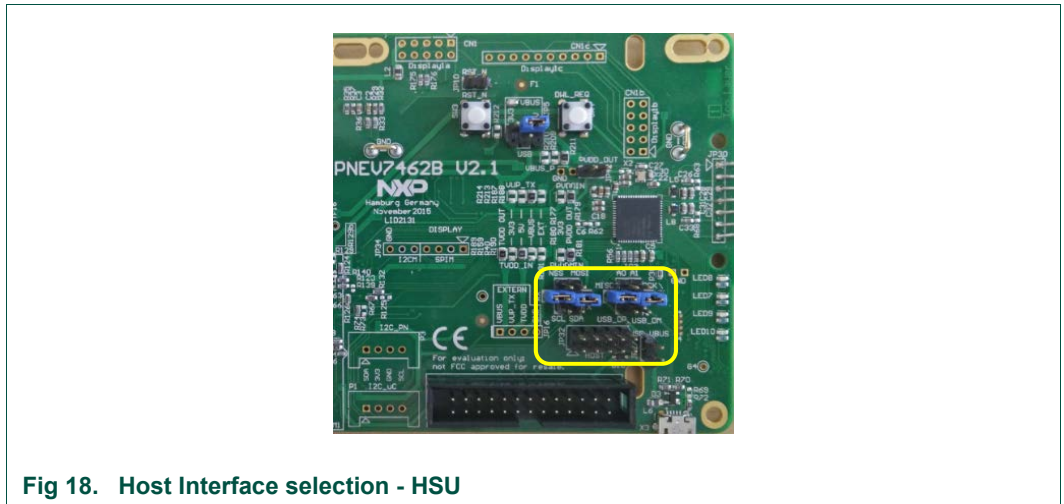


Fig 18. Host Interface selection - HSU

### 3.2.2 Debug interface

The PN7462 Customer Board is equipped with the JTAG/SWD interface. The JTAG/SWD connector is on the bottom left side of the board. LPC-Link2 standalone debug probe can be used to connect and to flash or debug application on the PN7462AU as illustrated on the Fig 10.

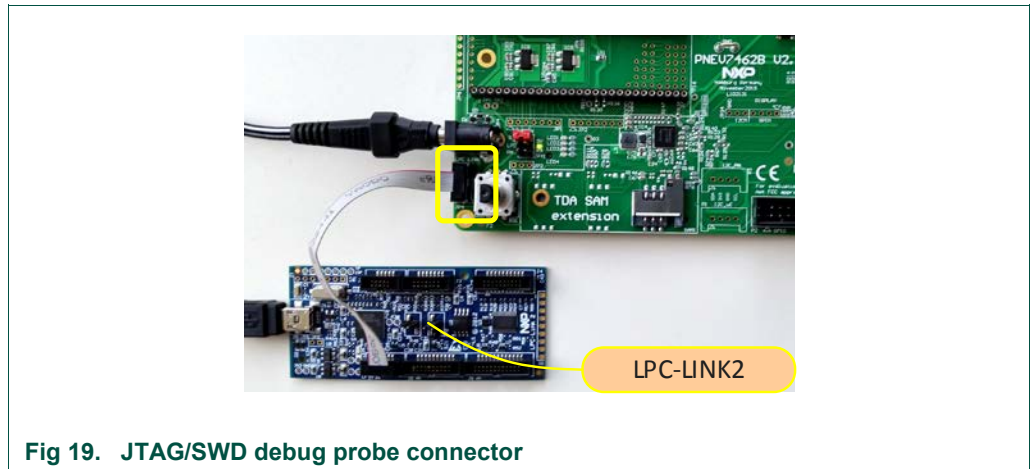


Fig 19. JTAG/SWD debug probe connector

## 4. NFC Cockpit getting started

This chapter provides the first steps to operate the board, using the NFC Cockpit (PN7462 GUI Version 3.6.0).

The PNEU7462 evaluation board is delivered with a graphical user interface application (GUI), the NFC Cockpit. The PN7462 NFC Cockpit can be used to explore the functionality of the PN7462 and perform RF and antenna design related tests. It allows a direct register access as well as EEPROM read and write access, and it allows to test and calibrate the DPC. The NFC Cockpit therefore can be used to configure & test the PN7462.

The default antenna is a 65mm x 65mm antenna with some metal layer inside the antenna area. This antenna is not an optimum antenna as such, but intends to demonstrate the performance and register settings of the PN7462 under typical design constraints like LCD or some metal (e.g. PCB) inside the antenna area.

## 4.1 Board preparation

To use NFC Cockpit the PNEV7462B board must be prepared for USB connection. The description on how to set up the board can be found in chapter 3 of this document. As a quick reference, following Jumpers should be set.

Recommended jumper settings:

Supply => PN (for full power use external supply)

VBUS => 5V

USB VBUS => set

USB D+ => set

USB D- => set

The jumper settings are shown in chapter 3. If full RF power is needed the supply jumper must be set to external supply.

## 4.2 Firmware and driver

NFC Cockpit requires a dedicated firmware running on the PN7462. This firmware is implementing CDC USB class translator driver (VCOM driver). The update of the firmware can be done by entering into the mass storage mode (see chapter 6.9). Firmware binary is available in the NFC Cockpit installation folder:

*"C:\nxp\NxpNfcCockpit\_v3.7.0.0\firmware\PN7462AU".*

In cases when the FW is overwritten, e.g. running any example from the PSP package, it must be updated again.

All drivers needed for NFC Cockpit are part of the installation package and are automatically installed.

## 4.3 NFC Cockpit installation

The NFC cockpit can be downloaded from the NFC Cockpit product web page [4]. After successful download, follow the installation wizard and finish the installation. The default installation directory is C:\nxp\NxpNfcCockpit\_v3.6.0.0.

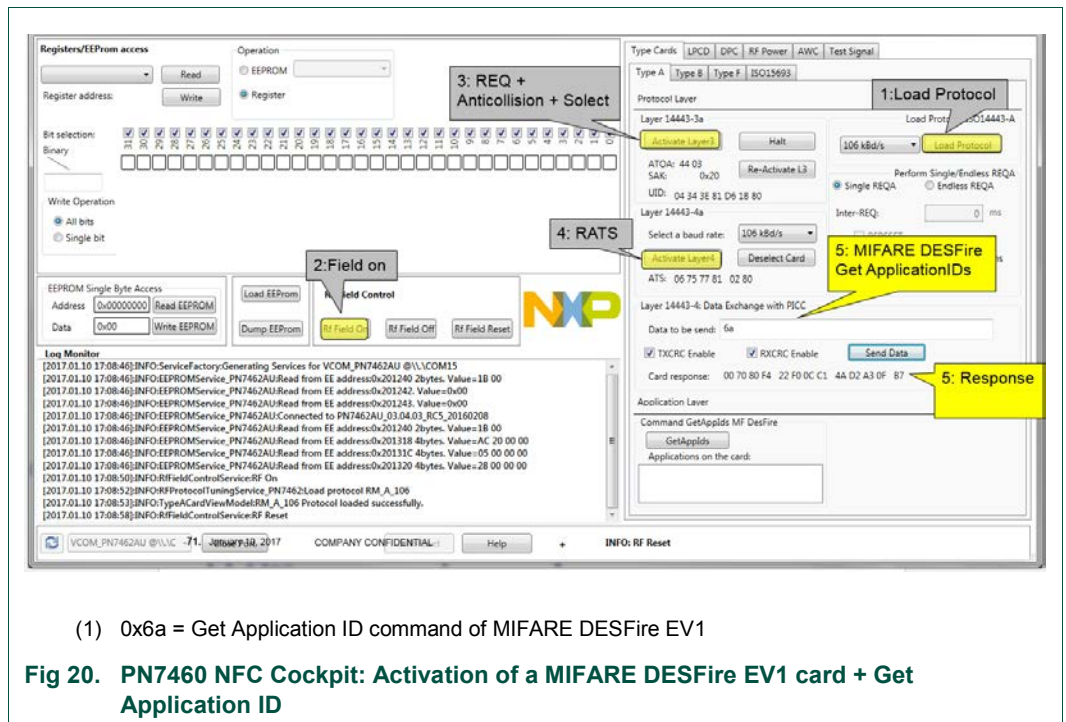
## 4.4 NFC Cockpit getting started

After starting the NFC Cockpit, the communication link between the PC and the PNEV7462B (via USB interface) is enabled automatically.

The Fig 20 shows the activation of a MIFARE DESFire card, using the <Load Protocol> + <Field On> + <Activate Layer3>, followed by <Activate Layer4>. The PN7462 NFC Cockpit shows the card responses like ATQA, SAK, and ATS.

Afterwards the ISO/IEC 14443-4 protocol can be used to exchange data. The Fig 20 shows the MIFARE DESFire command “Get Application ID” (0x6A), which returns the AIDs.

**Note:** Make sure that either the CRC is enabled or added manually in the data field.



Similar functionality does exist for ISO/IEC 14443 A and B, for NFC type F and for ISO/IEC 15693 communication.

Be aware that a LOAD\_RF\_CONFIG command must be executed manually before the corresponding protocol settings are loaded from the EEPROM into the registers. This can be used to perform

- (1) <Load Protocol> (e.g. type A 106)
- (2) <Field On>
- (3) <Single REQA> (using the EEPROM settings)
- (4) Select a TX register, e.g. RF\_CONTROL\_TX, enable TX\_SET\_BYPASS\_SC\_SHAPING
- (5) Change some register bits, and write back into RAM
- (6) <Single REQA> shows the register changes (probing the field and checking the envelop)

This allows an easy and quick optimization of Tx and Rx parameters before changing the EEPROM.

- (7) <Load Protocol> (e.g. type A 106)
- (8) <Single REQA> (using again the EEPROM settings)

#### 4.5 PN7462 Register access

The PN7462 NFC Cockpit allows the reading and writing of all the PN7462 registers (see Fig 21).

Selecting a register reads and shows the hexadecimal content as well as the corresponding bit values. The input allows to change each bit separately as well as writing hexadecimal values. Writing back the value changes the PN7462 register.

A help function automatically shows a short description of the register parts itself, if the mouse is moved over the names.

**Note:** Some register content cannot be changed manually (“read only”) and some content might be overwritten by the PN7462 firmware.

The screenshot shows the 'Registers/EEProm access' window. A yellow box highlights the 'Register access („RAM“) EEPROM Protocol access' section. Below this, there are buttons for 'EEPROM Single Byte Access' (Address, Read EEPROM, Write EEPROM) and 'RF Field Control' (Load EEPROM, Dump EEPROM, RF Field On, RF Field Off, RF Field Reset). A 'Log Monitor' window is open, displaying a series of log entries from 2017.01.10 17:08:46, including messages about VCOM\_PN7462AU @\VCOM15, EEPROMService\_PN7462AU, and RF field control actions. The main interface also shows protocol layer settings (Layer 14443-3a, 14443-4a) and application layer options (TXCRC Enable, RxCRC Enable).

- (1) Register area is a RAM area, i.e. might be overwritten or changed automatically

**Fig 21. PN7462 NFC cockpit register access**

All registers, which are used in the LOAD\_RF\_CONFIG command, can be read from the EEPROM. The user must select the register and the protocol.

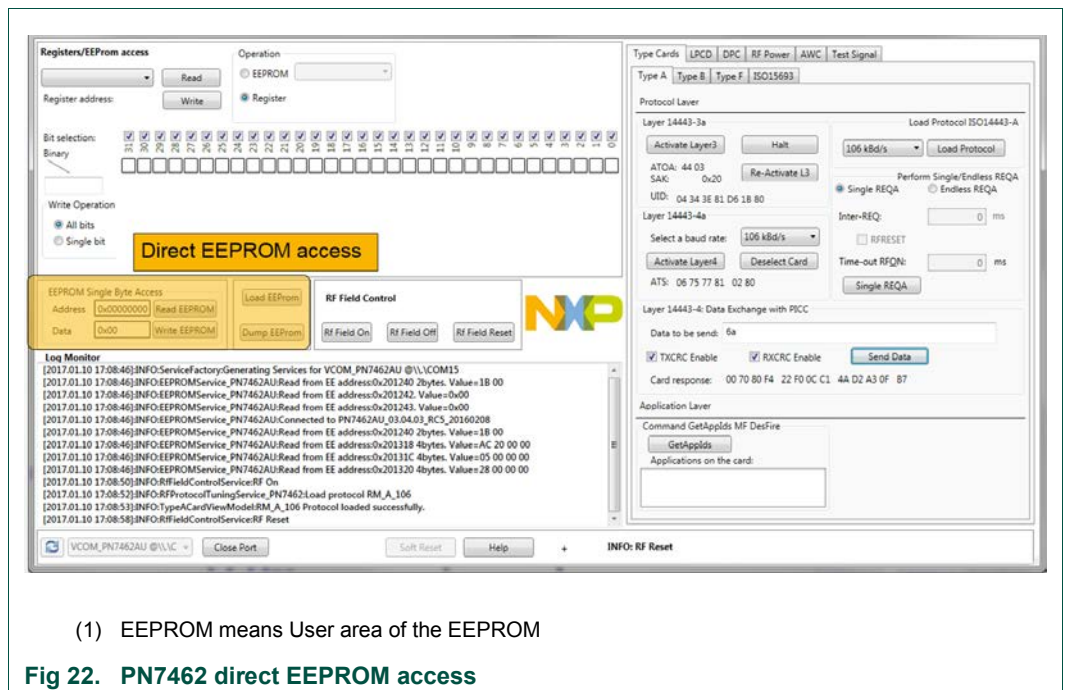
All registers, which are used in the LOAD\_RF\_CONFIG command, can be written into the EEPROM. The user must select the register and the protocol.

This allows an easy EEPROM update of the relevant TX and Rx registers after optimization in RAM.

#### 4.6 PN7462 EEPROM access

The NFC Cockpit allows 4 options of EEPROM access (see Fig 22):

- **Read EEPROM**  
Reads a single byte from EEPROM using byte address
- **Write EEPROM**  
Writes a single byte into EEPROM using byte address
- **Dump EEPROM**  
Stores the complete user area of the PN7462 EEPROM into a binary file. This can be used to generate a backup of all settings or to transfer optimized settings onto another board or into own software.
- **Load EEPROM**  
Loads a binary file and stores it into the user area of the PN7462 EEPROM.



### 4.7 PN7462 analog and digital test signals

The NFC cockpit allows to use the PN7462 internal test bus, to route some digital and analog test signals to the given test pins (GPIO1, 2 and GPIO 4, 5), as shown in. All details on the test signals can be found in [3].

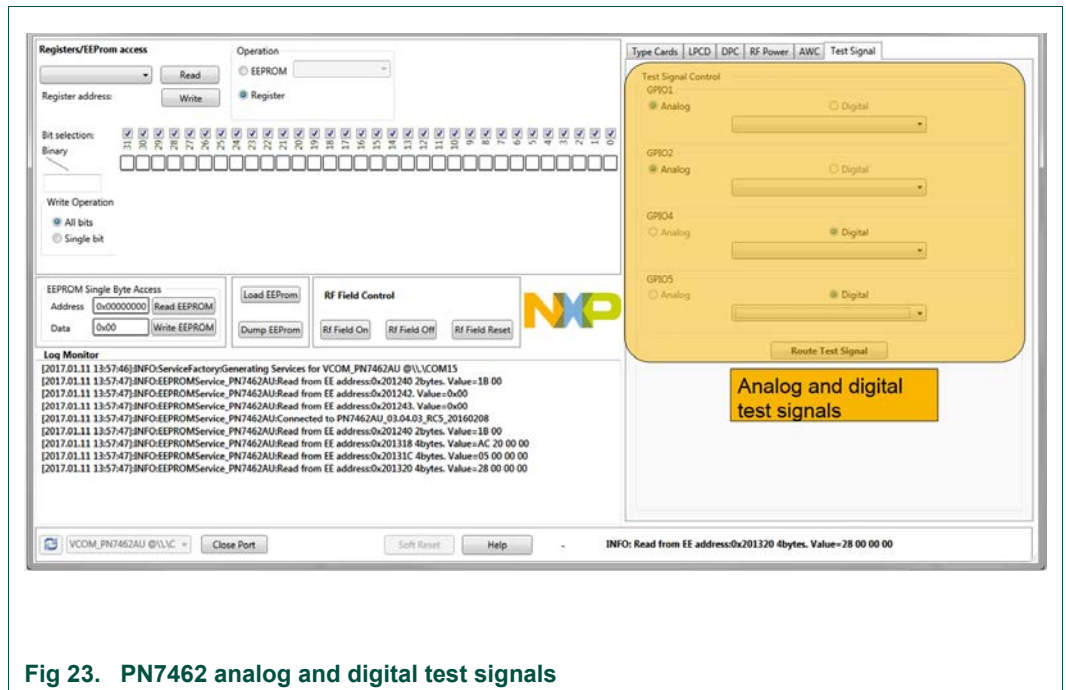


Fig 23. PN7462 analog and digital test signals

The analog test signals can directly be selected at GPIO1 and 2. For the digital test signals GPIO4 and 5 can be used.

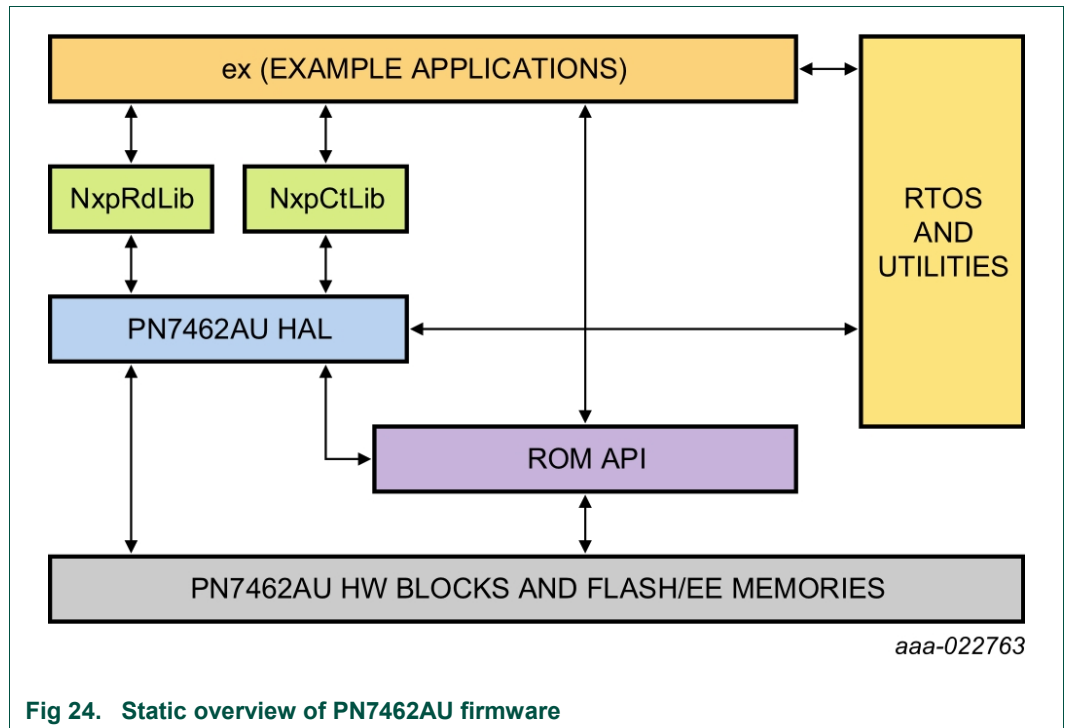
Afterwards the <Route Test Signal> activates the chosen signals.

## 5. 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





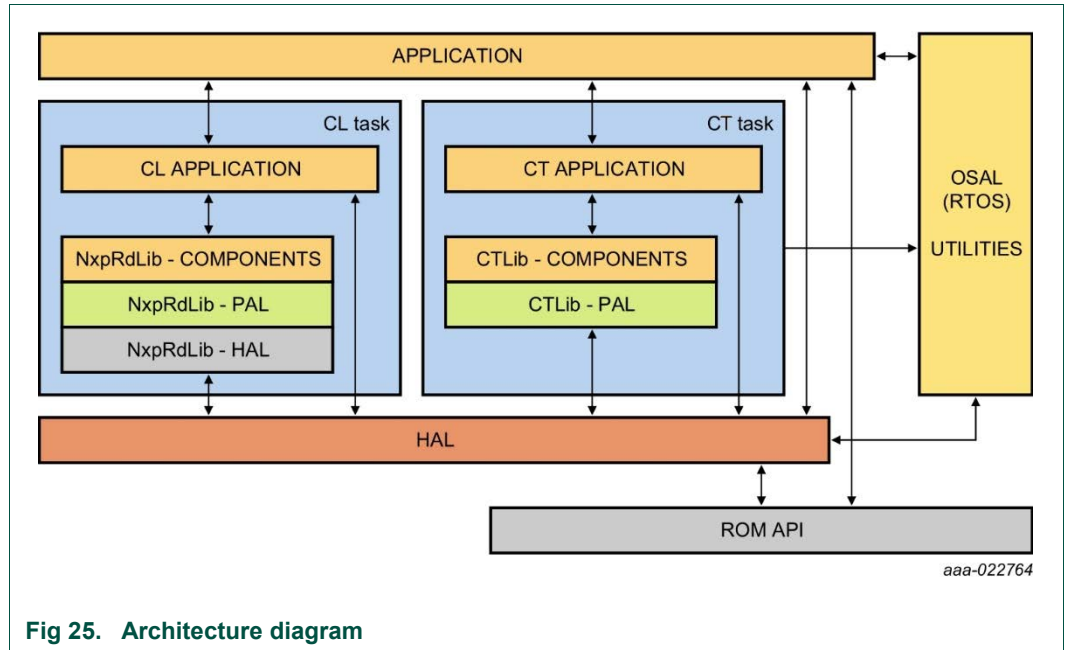


Fig 25. Architecture diagram

### 5.1 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.

### 5.2 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.

### 5.3 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.

### 5.4 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.*

5.5 Component view

5.5.1 Contactless component view

In contactless component view (Fig 26) for the “phExMain” example is shown.

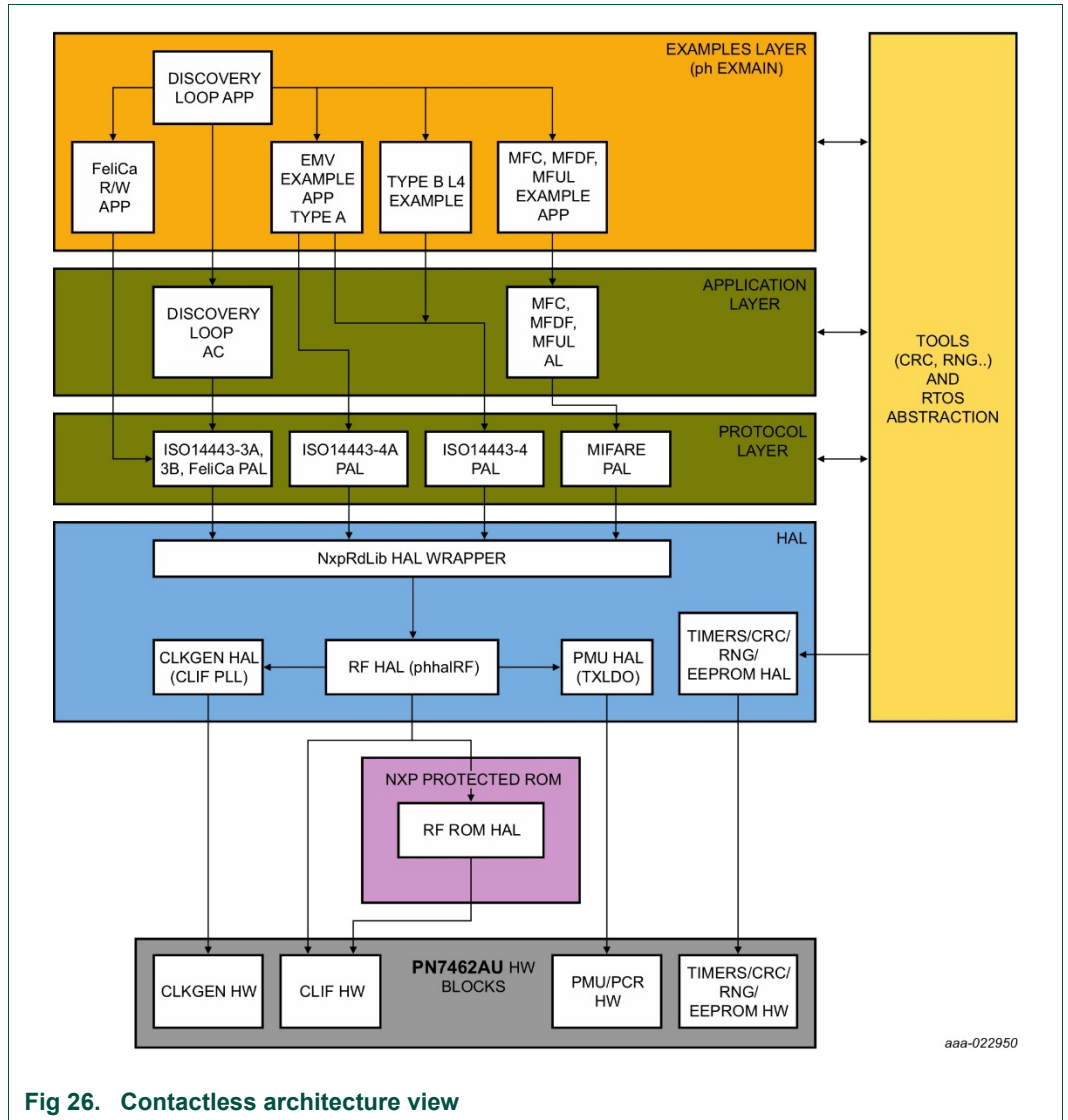


Fig 26. Contactless architecture view

5.5.2 Contact component view

In the Fig 27 contact component view for the “phExMain” example is shown.

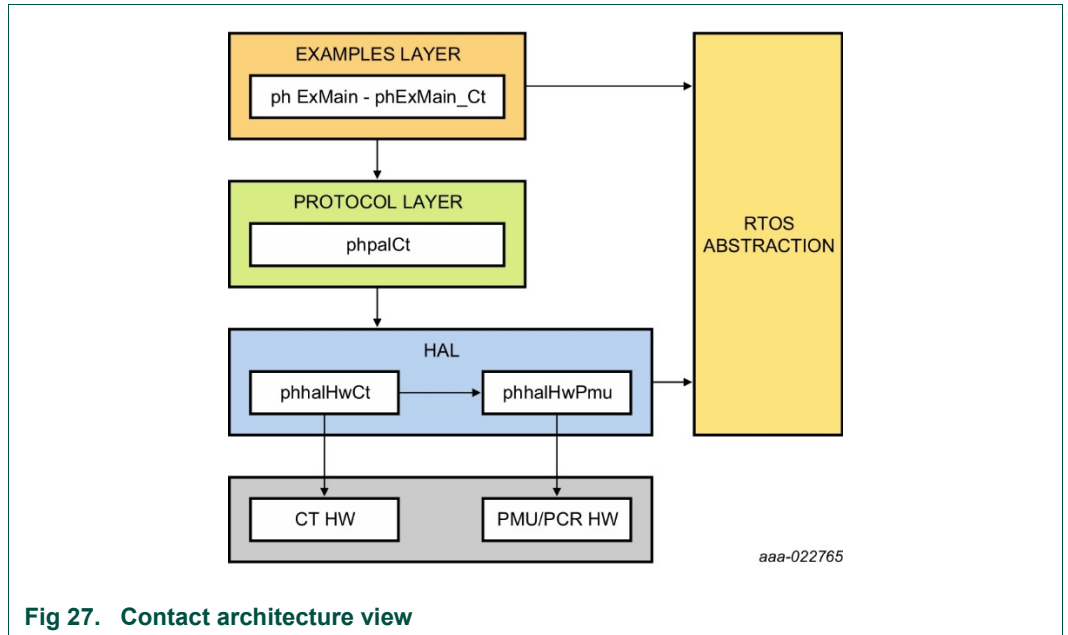


Fig 27. Contact architecture view

### 5.6 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 28 illustrates the mentioned implementation for the initialization procedure of a “phExtMain” application.

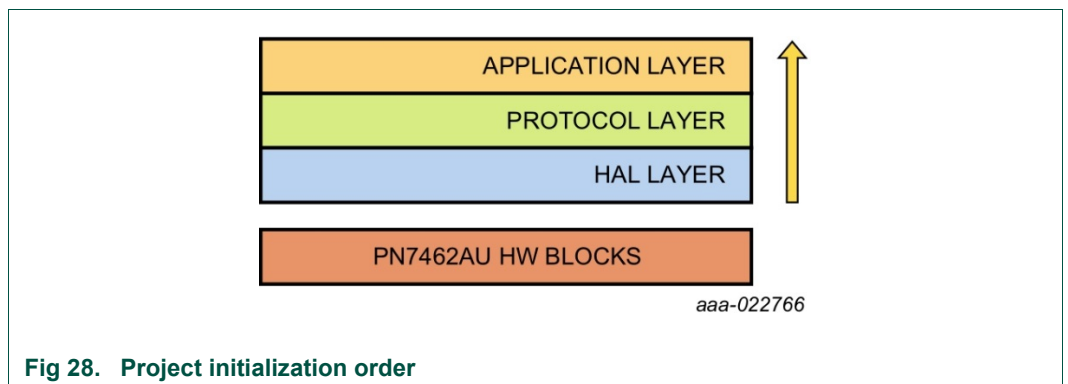


Fig 28. Project initialization order

### 5.7 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](http://www.nxp.com/redirect/freetos)

The Fig 29 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.

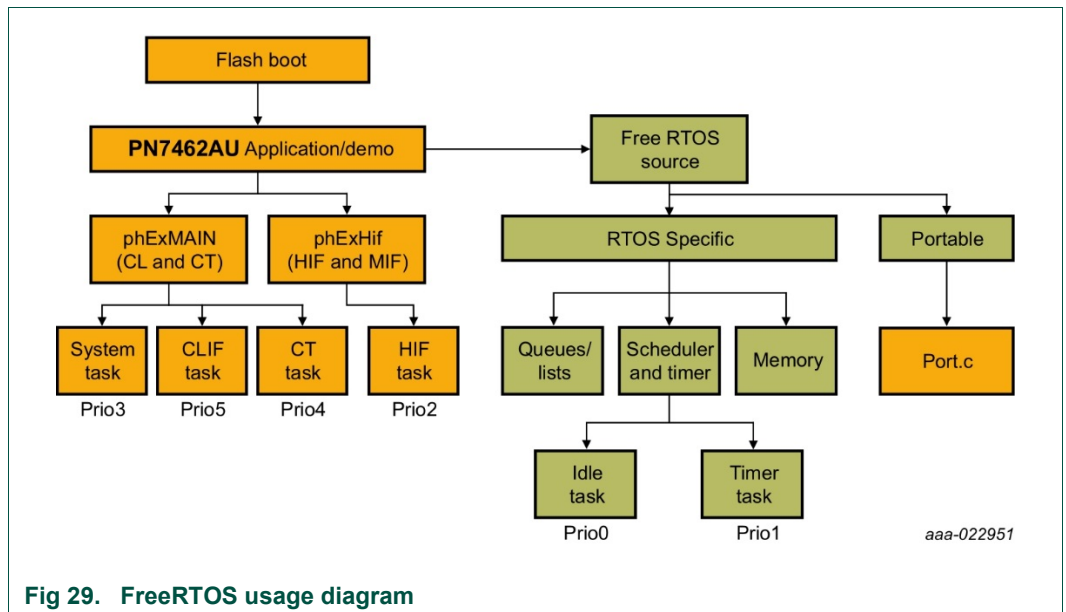


Fig 29. FreeRTOS usage diagram

## 6. Managing the PN7462AU SW projects with LPCXpresso IDE

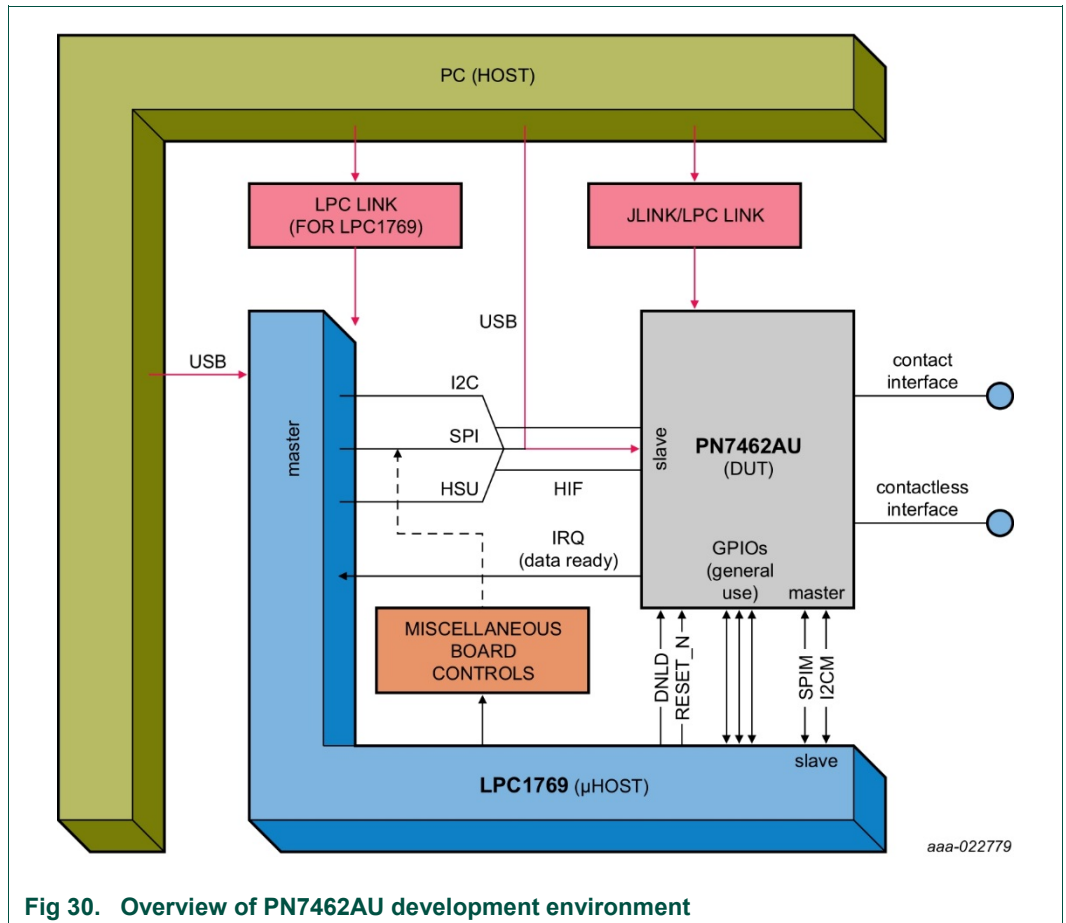
### 6.1 Development environment

For developing PN7462AU firmware and customer applications all components listed in the Table 3 are required.

**Table 3. Development environment**

Item	Version	Purpose
PN7462AU Customer board	2.1	Engineering development board
LPC Link 2	1.0	Standalone debug adaptor
LPCXpresso IDE	8.2.2	Development IDE
LPCXpresso PN7462AU plugin	com.nxp.pn7xxxx.update-8.0.0-SNAPSHOT-163	Adding support for PN7462AU to the LPCXpresso
PN7462AU FW and SW examples	4.050.03_001651	Installer package

Fig 30 gives general overview of the development environment elements and their interconnections:



**Fig 30. Overview of PN7462AU development environment**

## 6.2 Installation of the 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.

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.

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.

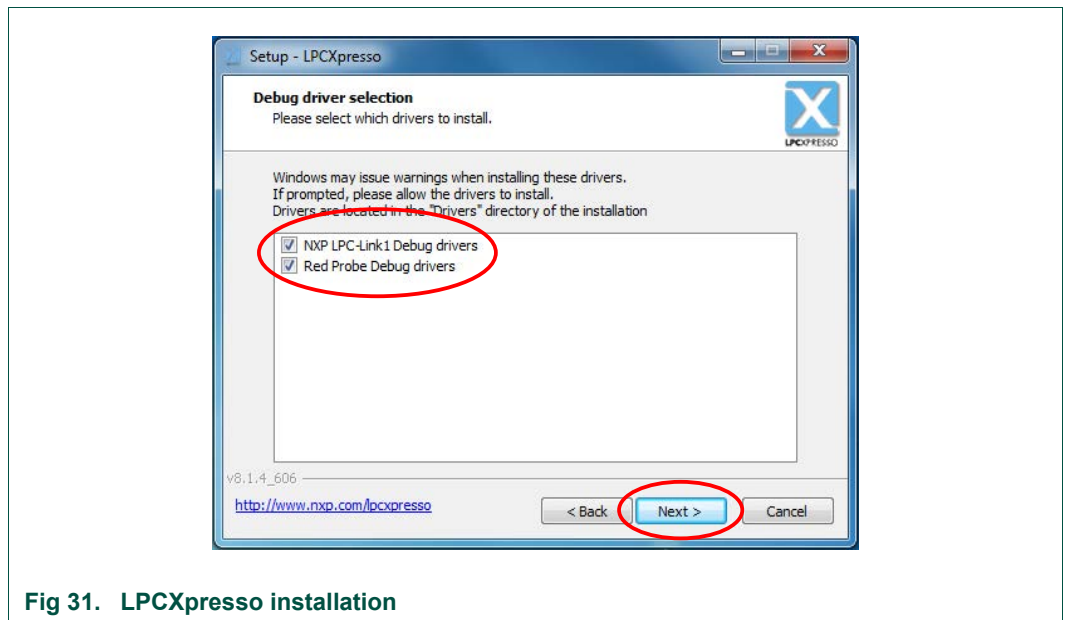


Fig 31. LPCXpresso installation

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.

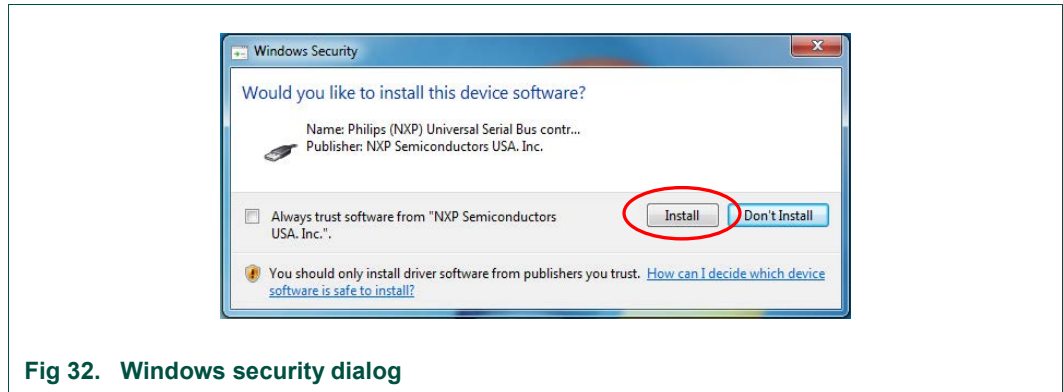


Fig 32. Windows security dialog

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

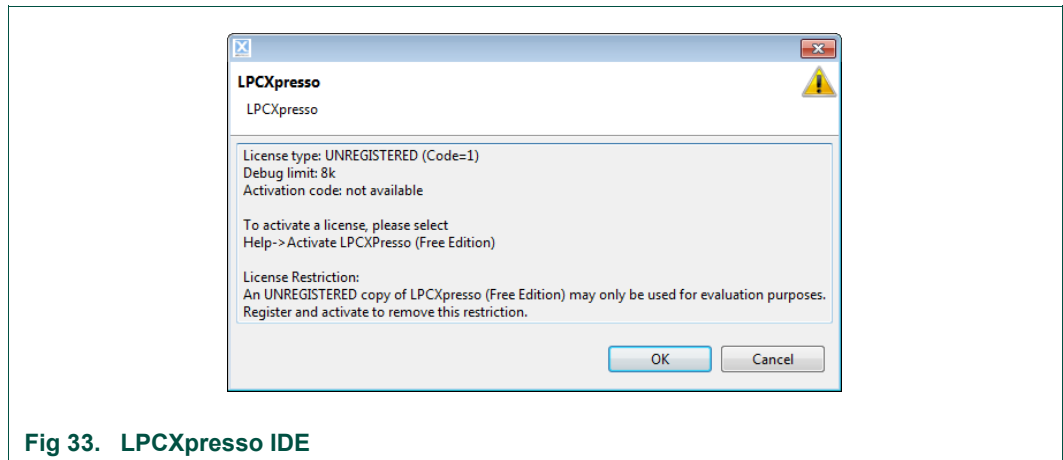


Fig 33. LPCXpresso IDE

Right 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.



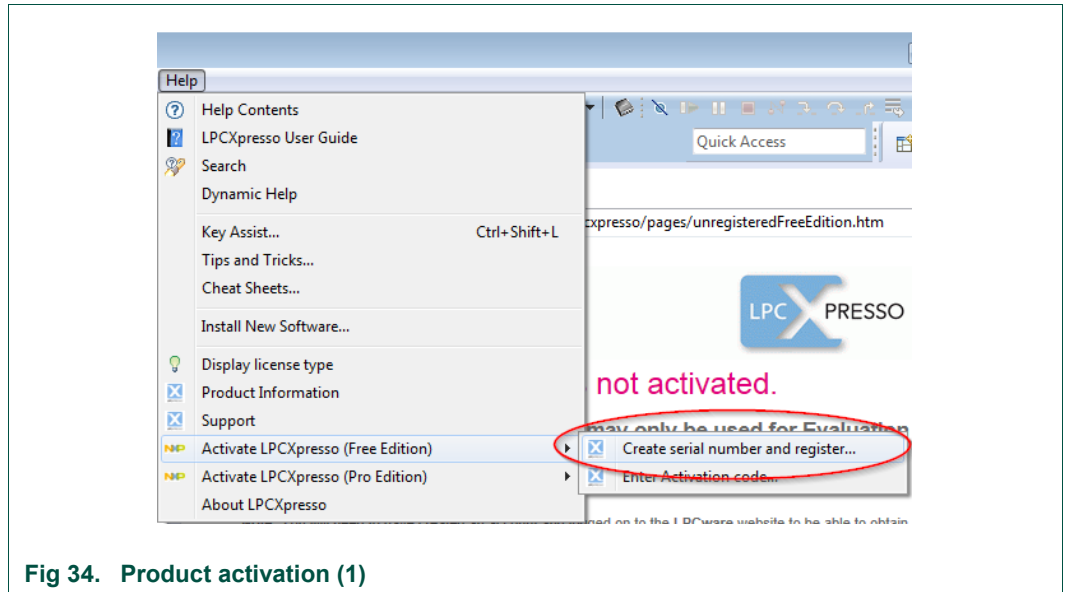


Fig 34. Product activation (1)

Product's serial number will be displayed.

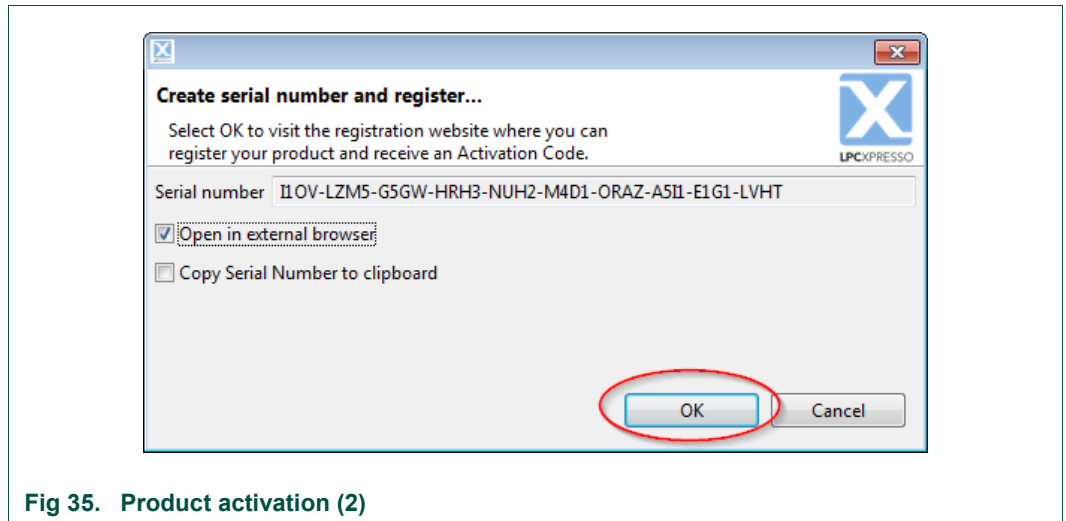
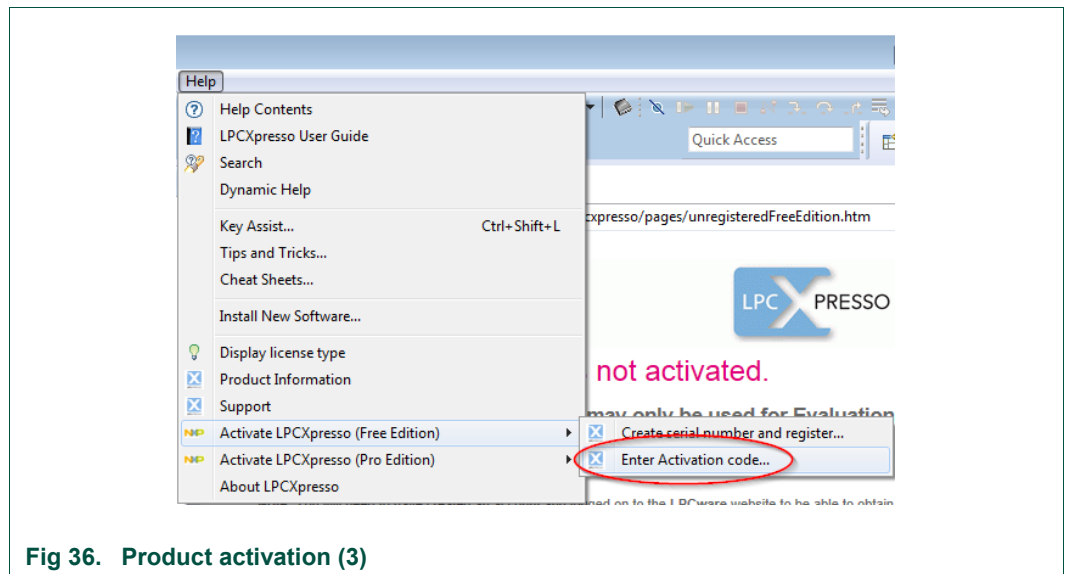


Fig 35. Product activation (2)

After pressing OK the browser will be opened on the Activation page. NXP user account is needed in order to activate LPCXpresso. If the user is logged in then the serial number will be entered already. Otherwise it is necessary to login and navigate to the <http://www.lpcware.com/lpcxpresso/activate> page and enter the product's serial number.



**Fig 36. 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.

### 6.3 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.

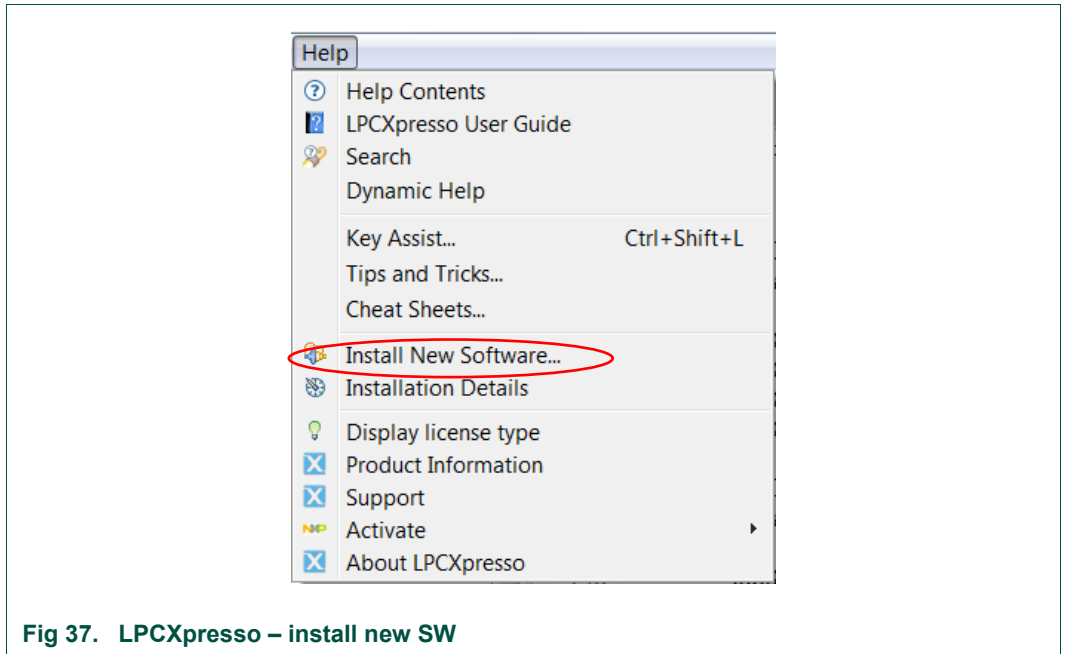


Fig 37. LPCXpresso – install new SW

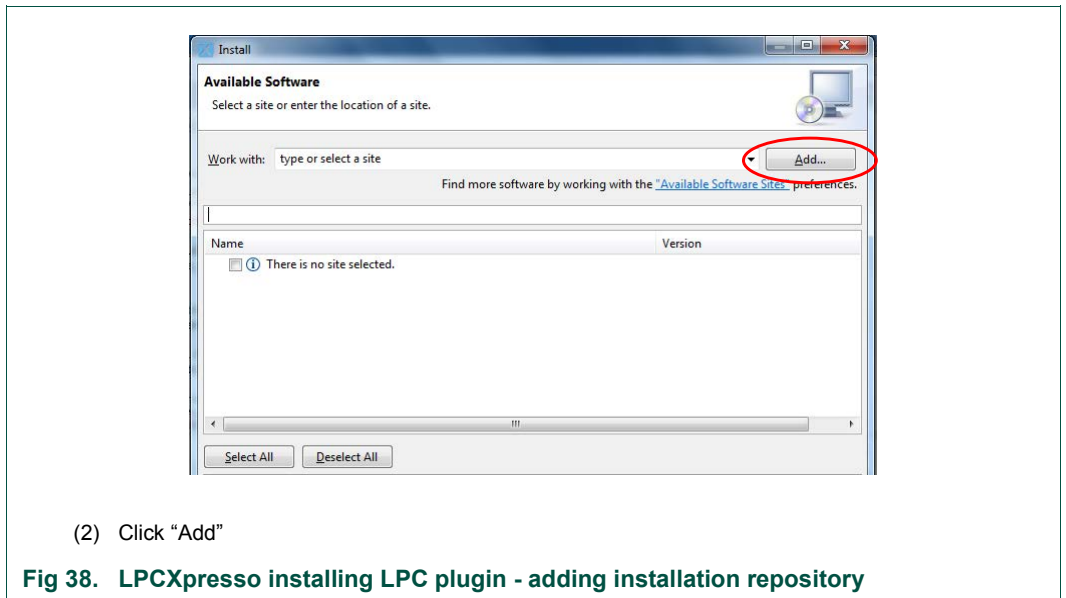
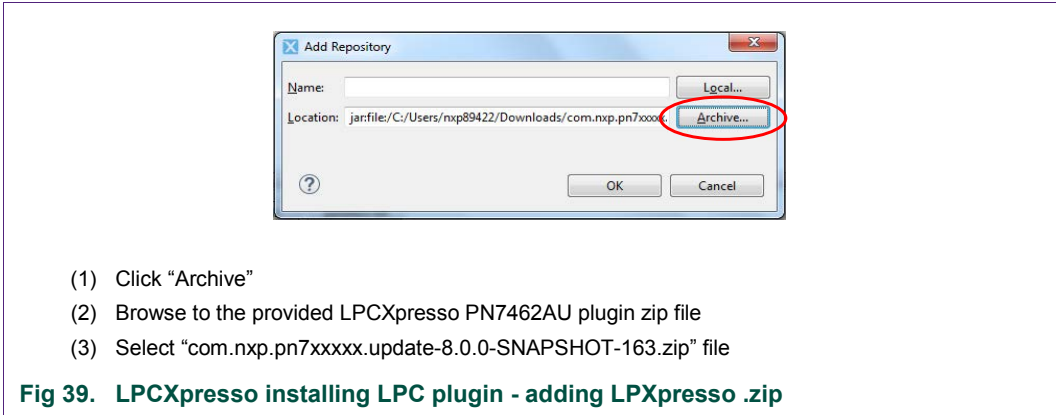


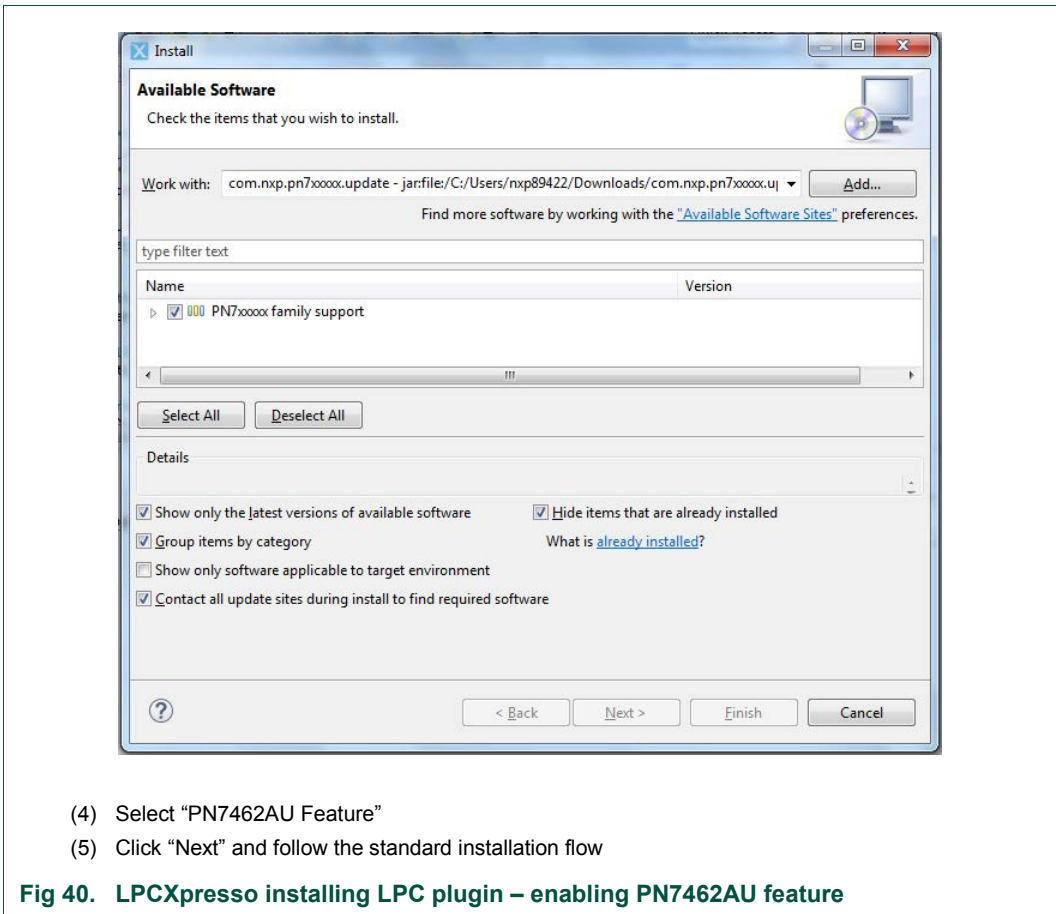
Fig 38. LPCXpresso installing LPC plugin - adding installation repository

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



- (1) Click "Archive"
- (2) Browse to the provided LPCXpresso PN7462AU plugin zip file
- (3) Select "com.nxp.pn7xxxx.update-8.0.0-SNAPSHOT-163.zip" file

**Fig 39. LPCXpresso installing LPC plugin - adding LPXpresso .zip**



- (4) Select "PN7462AU Feature"
- (5) Click "Next" and follow the standard installation flow

**Fig 40. LPCXpresso installing LPC plugin – enabling PN7462AU feature**

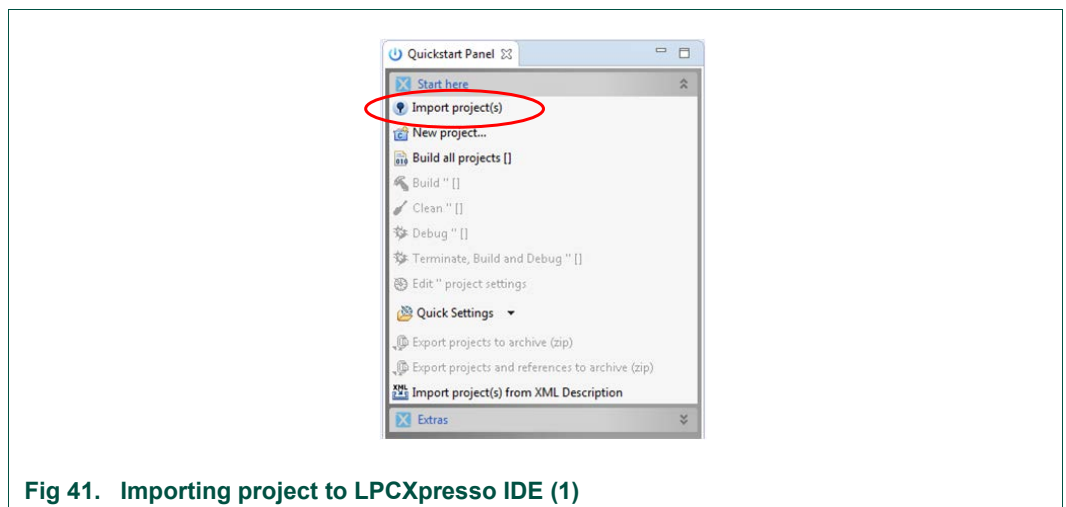
## 6.4 Importing provided SW example projects

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 provided in archive file and are part of the “PN7462AU Product Support Package”, this package needs to be installed.

To import the SW example projects please follow next steps.

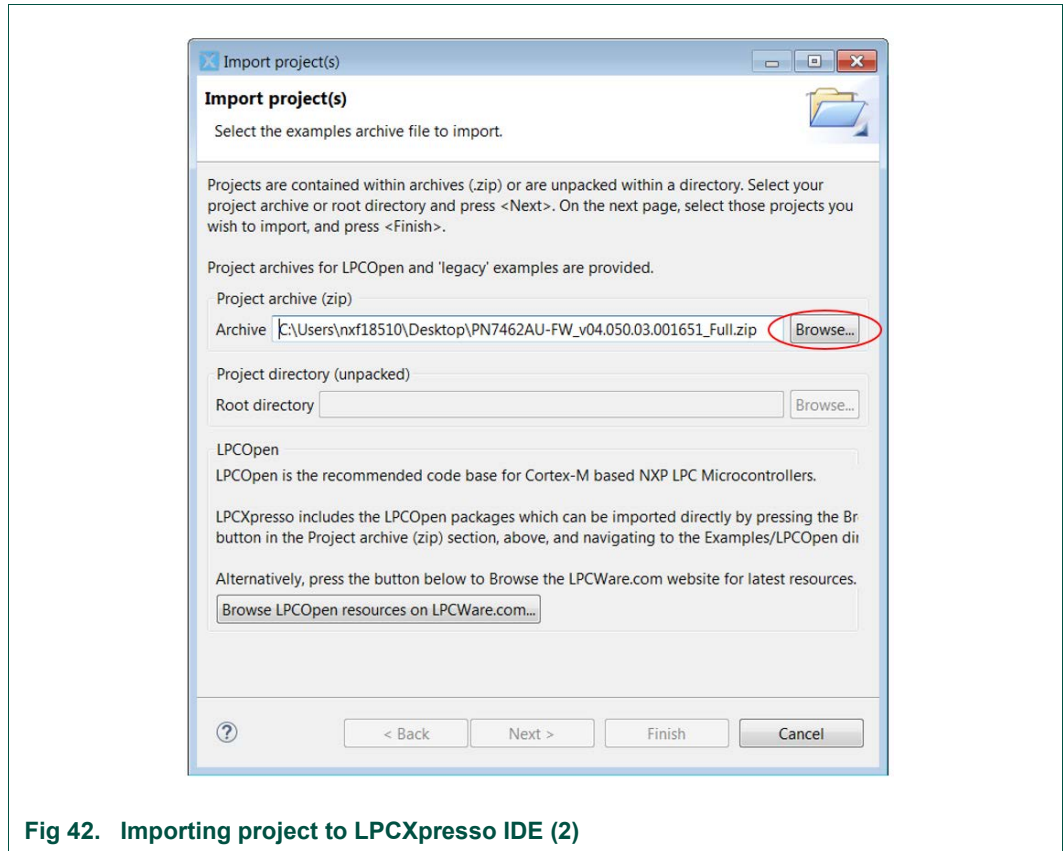
- Start the LPCXpresso IDE and select new workspace
- Select the option “Import project(s)” (see picture below)
- Browse to the zip archive
- LPCXpresso IDE unzips the software package
- The software package is ready for use



**Fig 41. Importing project to LPCXpresso IDE (1)**

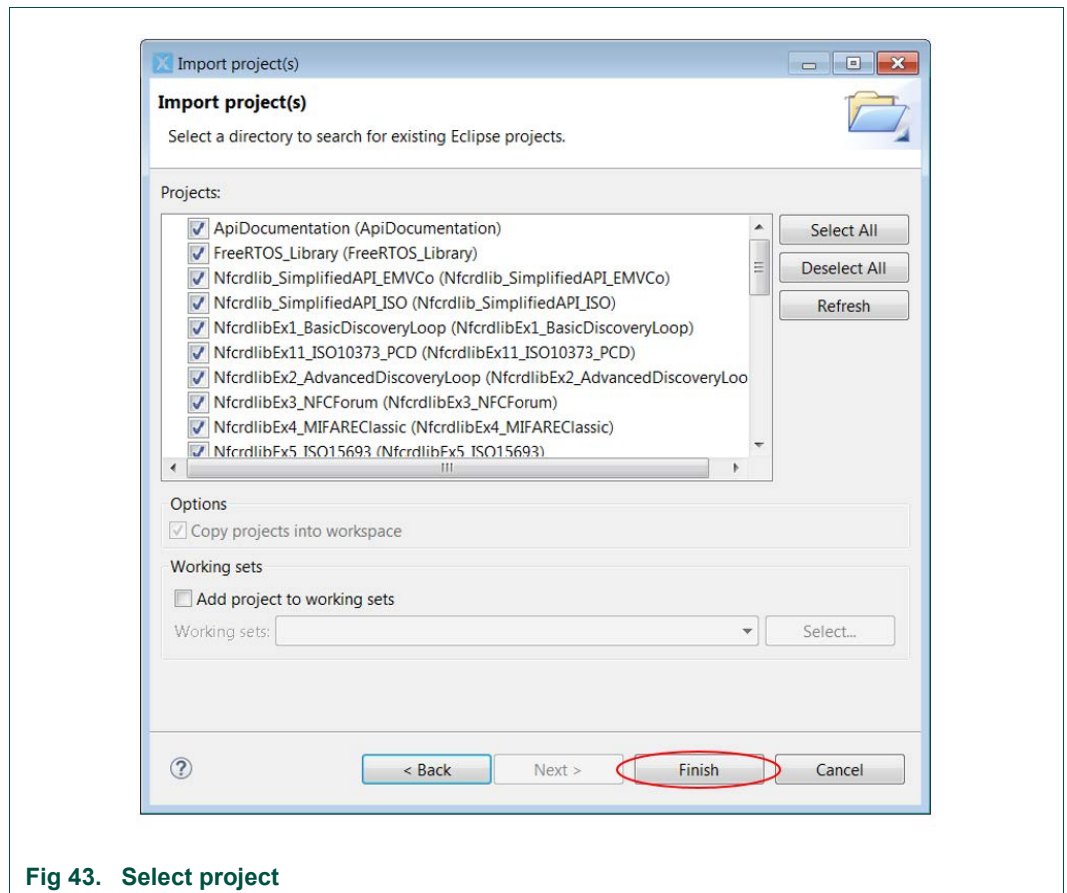
In the “Quickstart Panel” window, which can be found on the left hand side, click on Import project(s).

The dialog for importing projects opens.



**Fig 42. Importing project to LPCXpresso IDE (2)**

Browse to the project zip file *PN7462AU-FW\_v04.050.30.001651\_Full.zip* and click “Next”.



**Fig 43. 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.

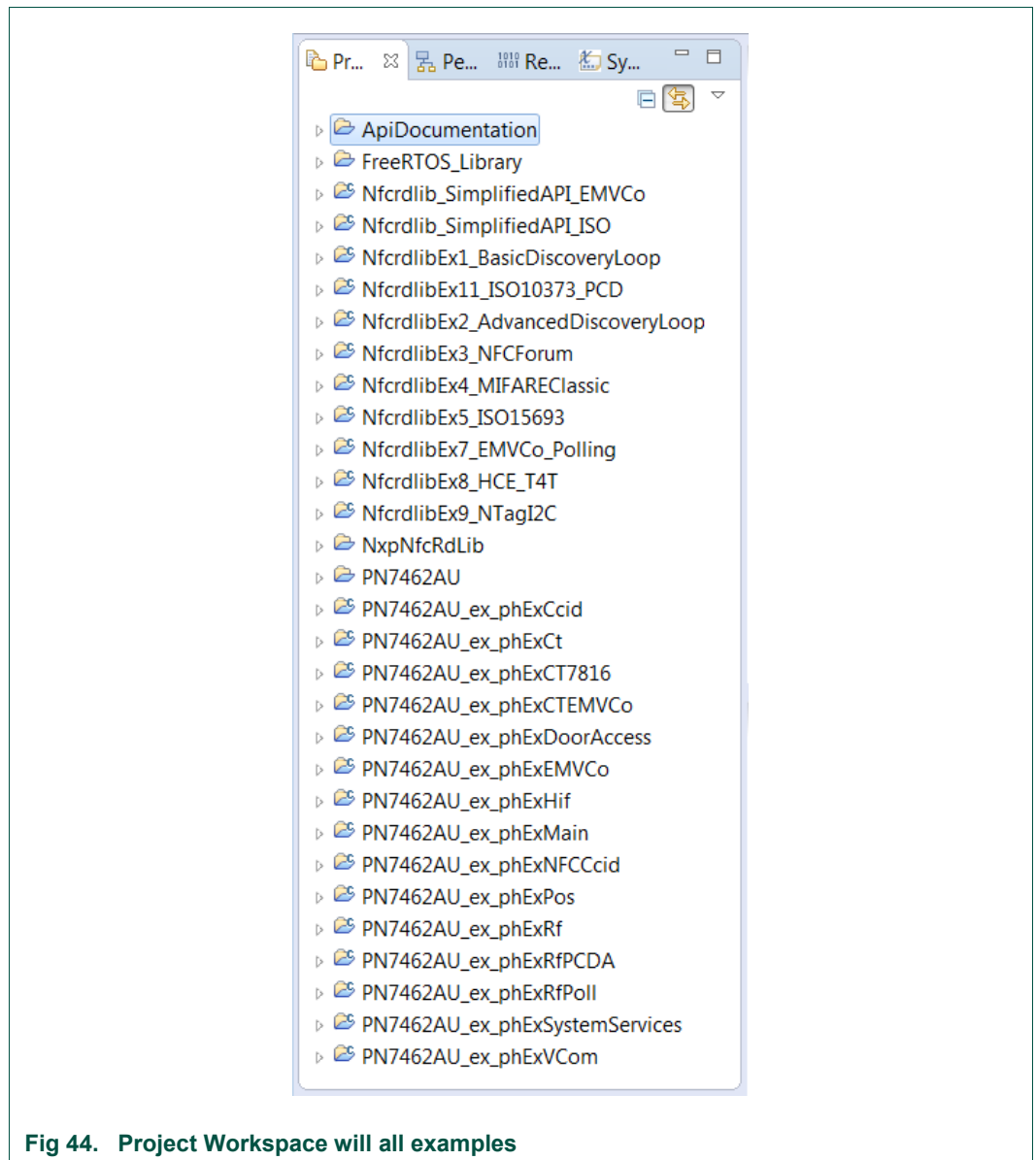


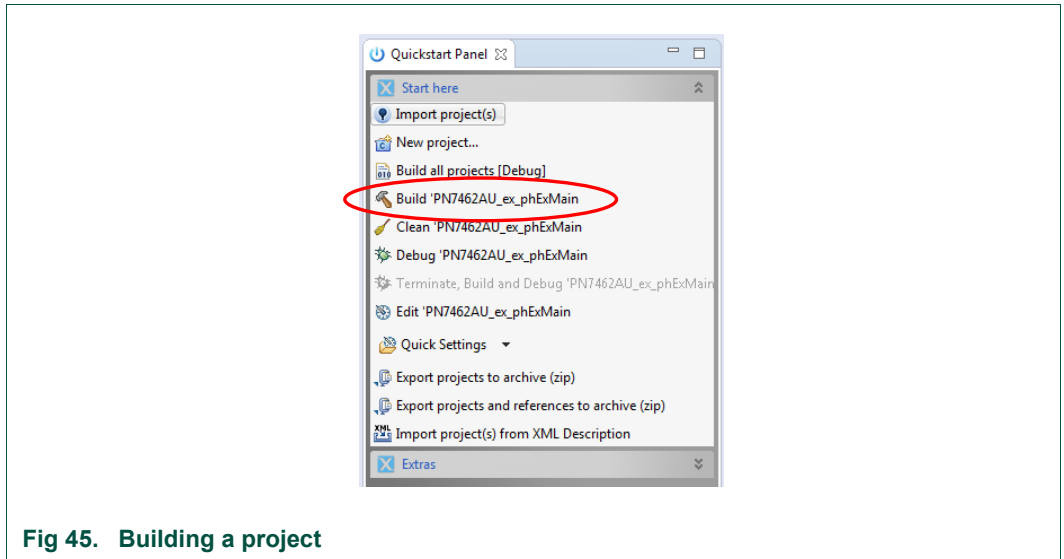
Fig 44. Project Workspace will all examples

## 6.5 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.

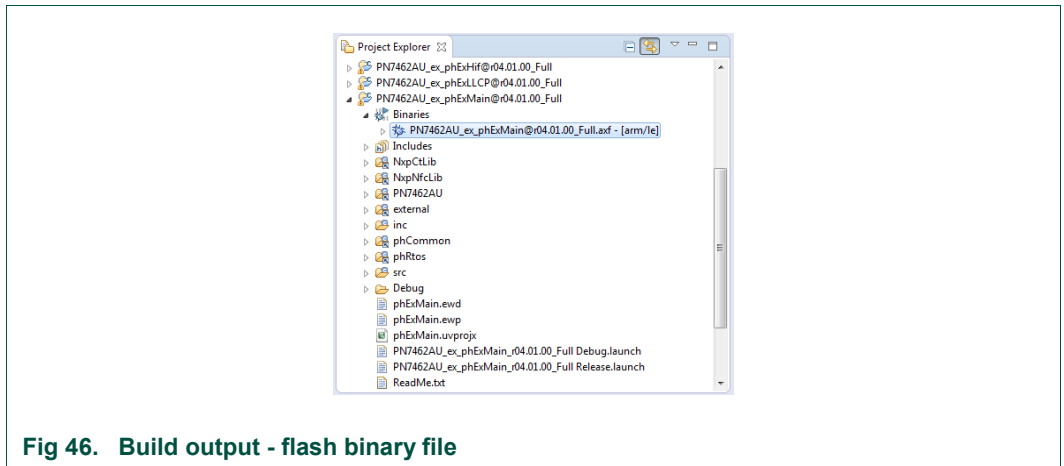
The project can be built as shown in the Fig 45.





**Fig 45. Building a project**

As a part of the build process, 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”).



**Fig 46. 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”.

Build result can be monitored on the build console Successful build Fig 47.

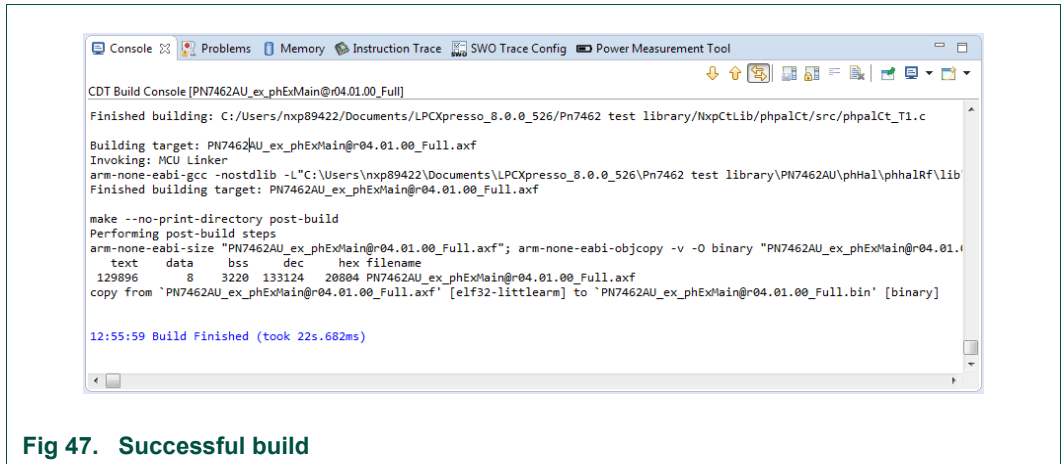


Fig 47. Successful build

### 6.6 Running and debugging the example projects

This description shows how to run the PN7462AU “PN7462AU\_ex\_phExMain” example application for the PN7462AU customer development board in debug mode. 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.

The PNEV7462B customer board needs to be connected to the host PC running the LPCXpresso software via LPC-LINK2, as shown in Fig 48.

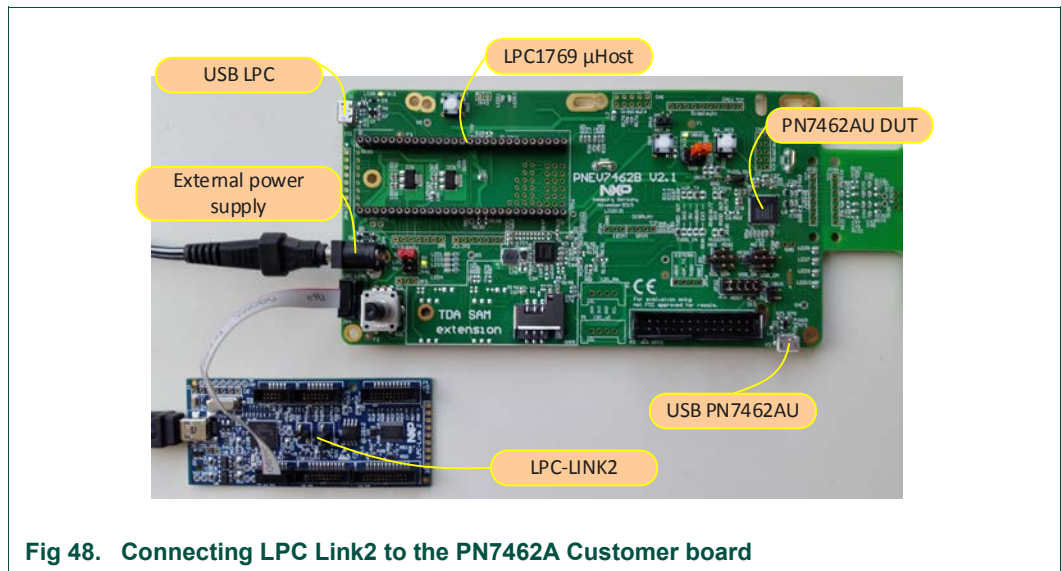


Fig 48. Connecting LPC Link2 to the PN7462A Customer board

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 49. The LPCXpresso IDE will first build application and then start debugging.

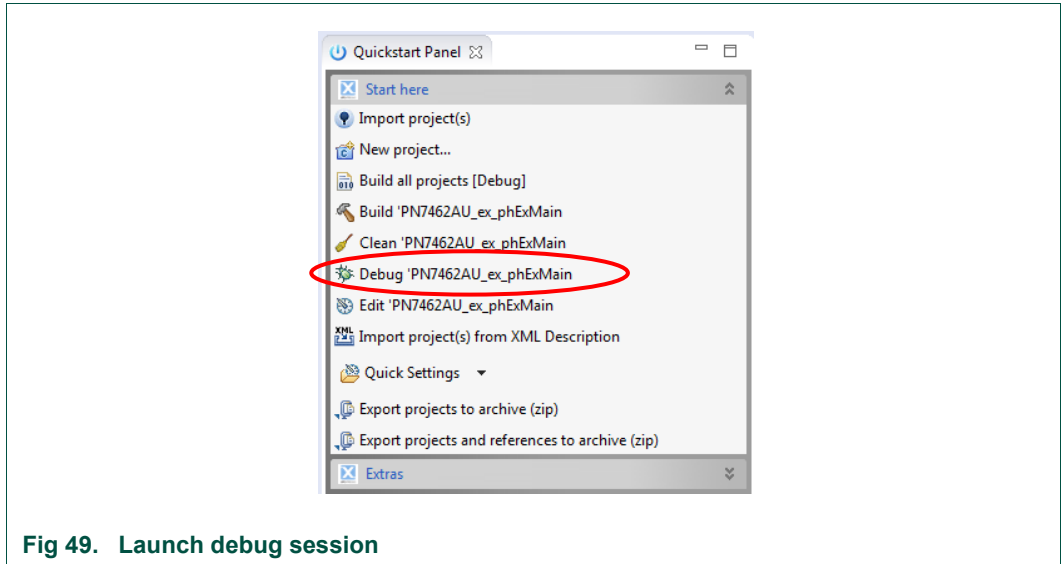


Fig 49. Launch debug session

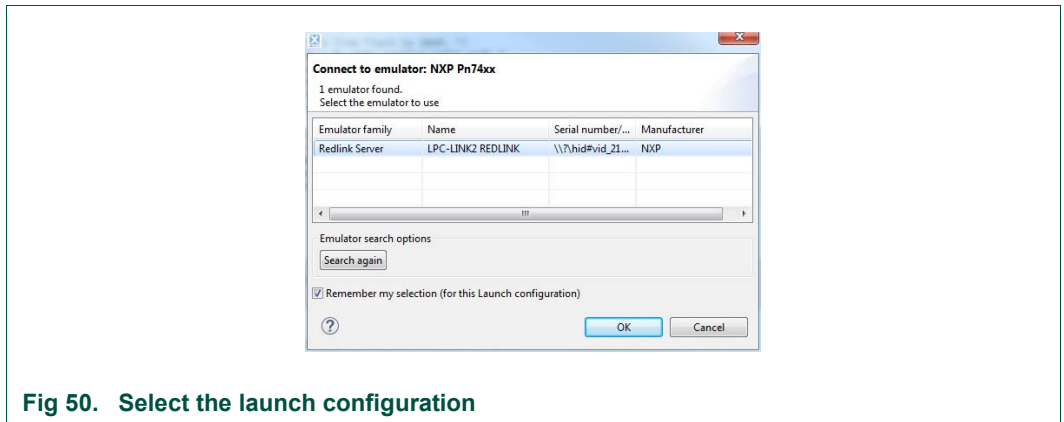


Fig 50. Select the launch configuration

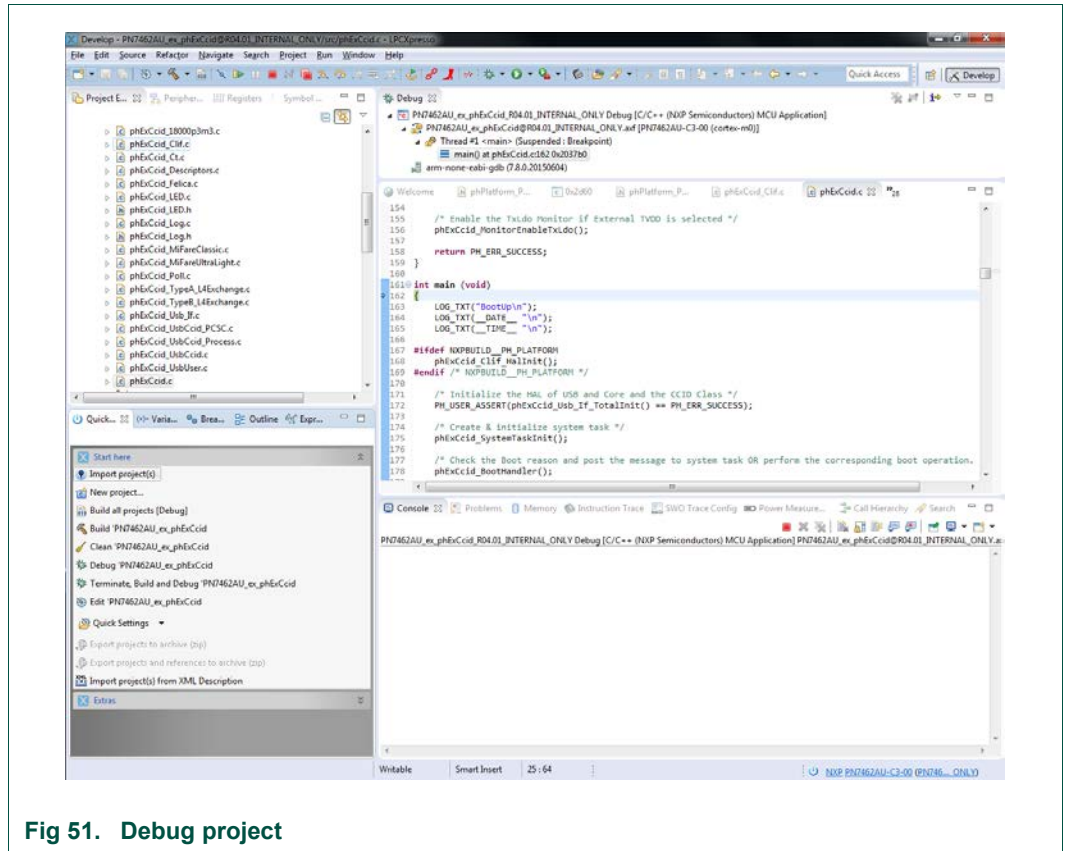


Fig 51. Debug project

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

### 6.6.1 Break points

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

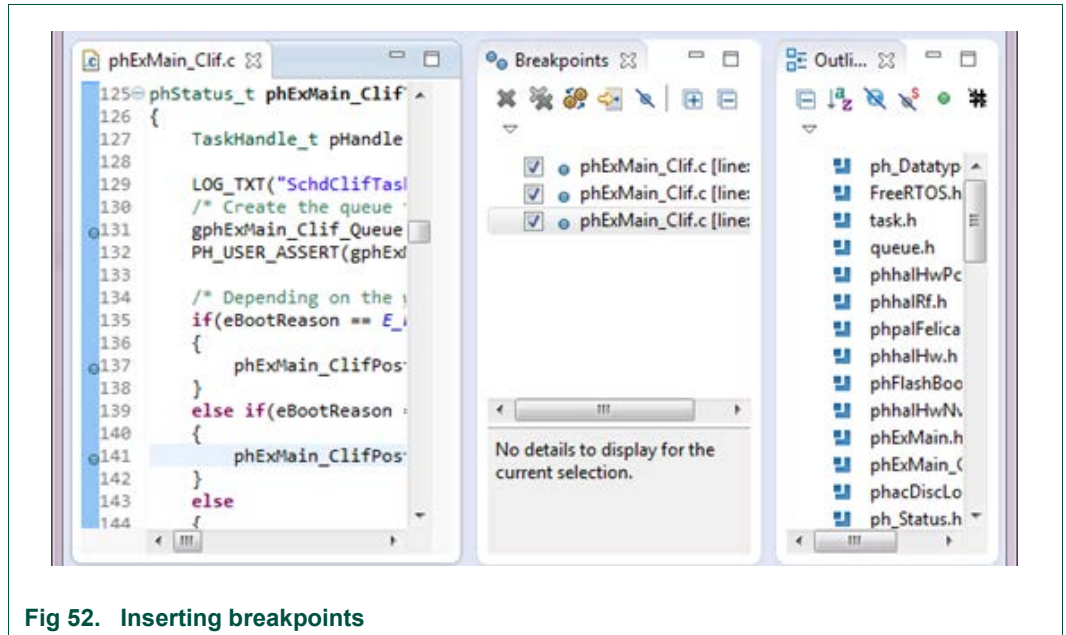


Fig 52. Inserting breakpoints

### 6.6.2 Debug traces

The debug traces can be seen on console as shown in Fig 53.

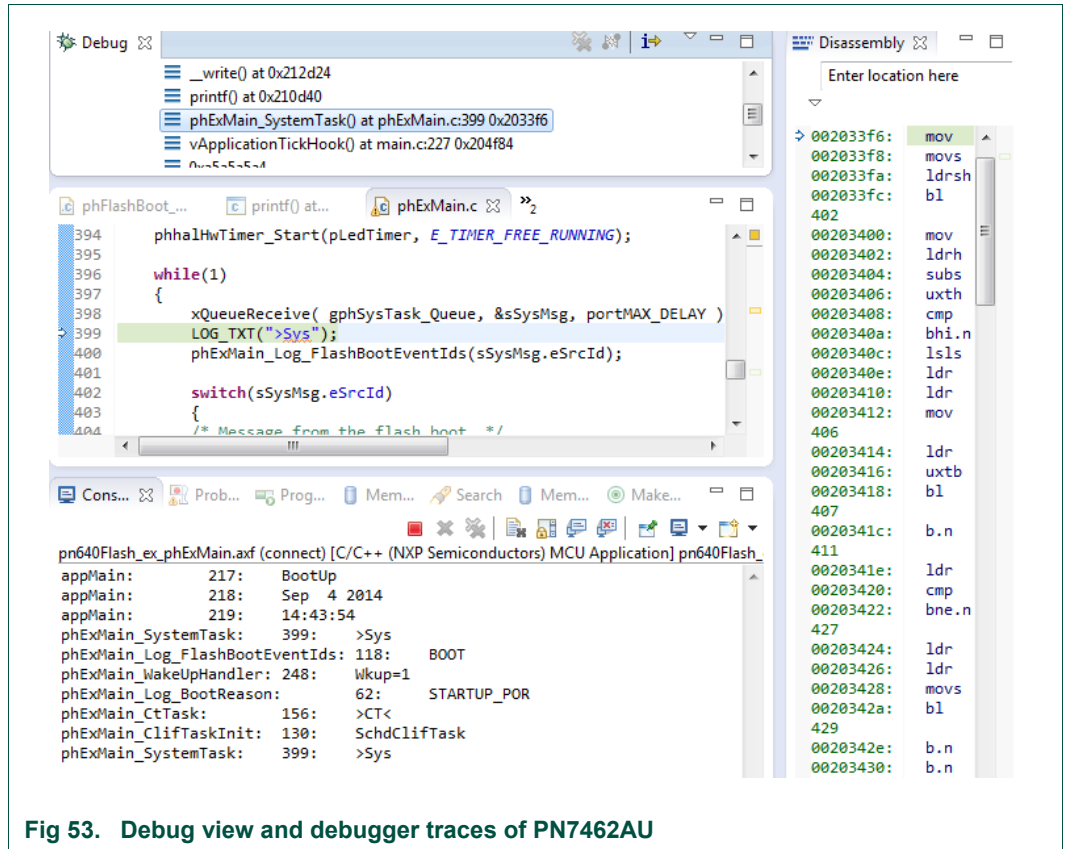


Fig 53. Debug view and debugger traces of PN7462AU

### 6.6.3 Peripheral view

After installing PN7462AU Plugin to the LPCXpresso, (See chapter 6.3), 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.

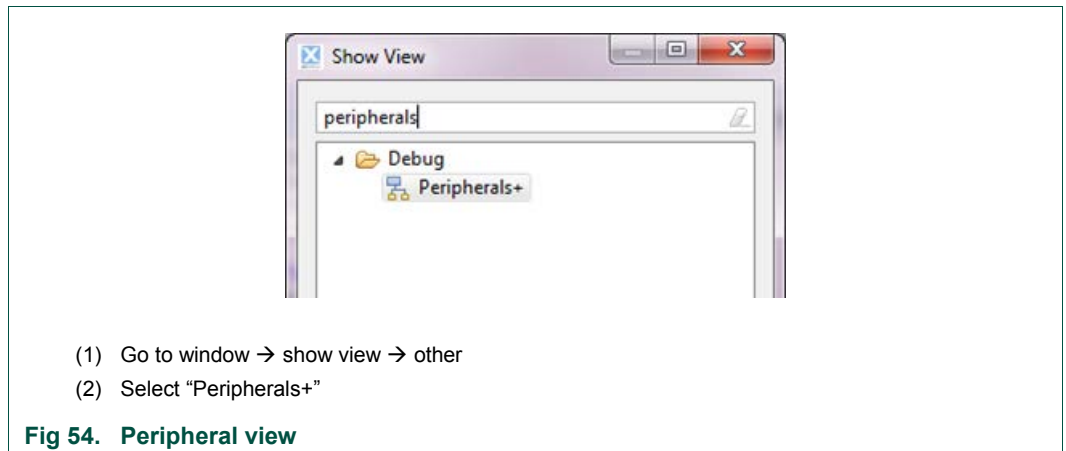


Fig 54. Peripheral view

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).

The screenshot displays a debugger interface with the following components:

- Project Explorer:** Shows a tree view of peripherals including CLIF, CRC, CT, ClockGen, DCR, EECTRL, HostIf, I2CM, NVIC, PCR, and PMU.
- Registers View:** A table listing peripherals and their addresses and descriptions. The EECTRL peripheral at address 0x200000 is selected.
- Monitors View:** Shows the EECTRL [Pn640] register view with a sub-table of registers and their values. The EECTRL Dynamic Control Register (WO), Write only is highlighted.
- Assembly View:** Shows assembly code traces with addresses and instructions such as `mov`, `ldr`, `subs`, `uxth`, `cmp`, `bhi.n`, `lsls`, `ldr`, `uxtb`, `bl`, and `b.n`.

- (1) Select the relevant peripheral
- (2) Register, bit fields and description in the relevant view

**Fig 55. Debug view and debugger traces of PN7462AU**

### 6.7 Updating Customer Board Firmware (Flash memory)

The customer board firmware is located in the PN7462AU flash memory. The Flash memory can be updated either Using USB Mass Storage mode (primary downloader) or using SWD interface via LPC-LINK2 debug probe.

### 6.8 Updating Flash via SWD interface

Ensure that LPC-LINK2 is connected to PN7462AU Fig 56 and follow the steps shown below.

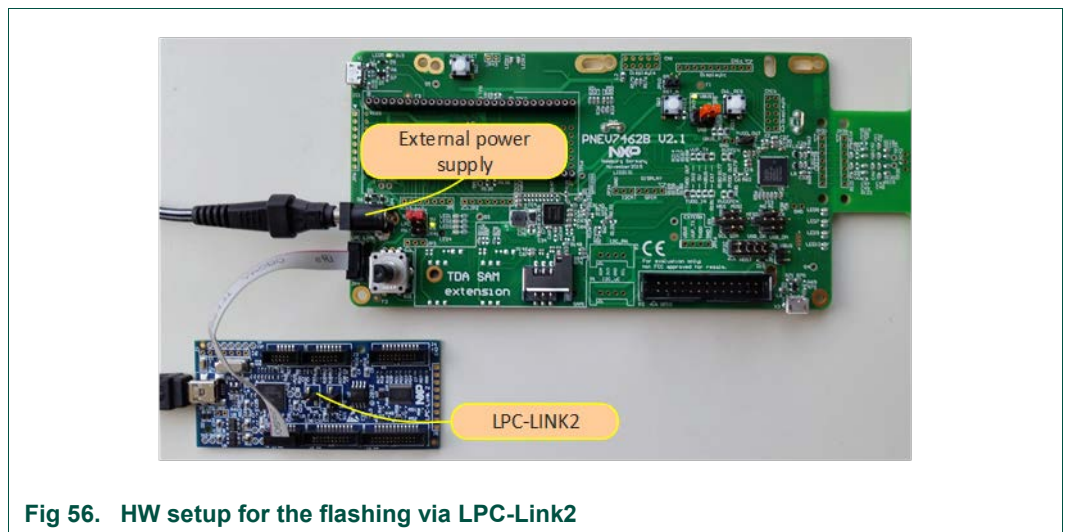
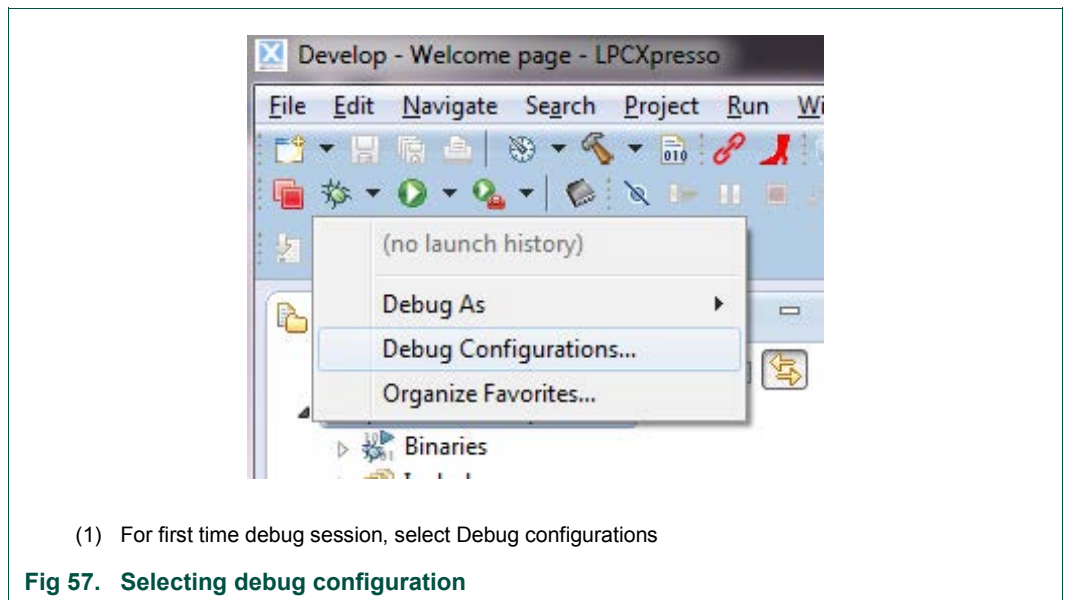


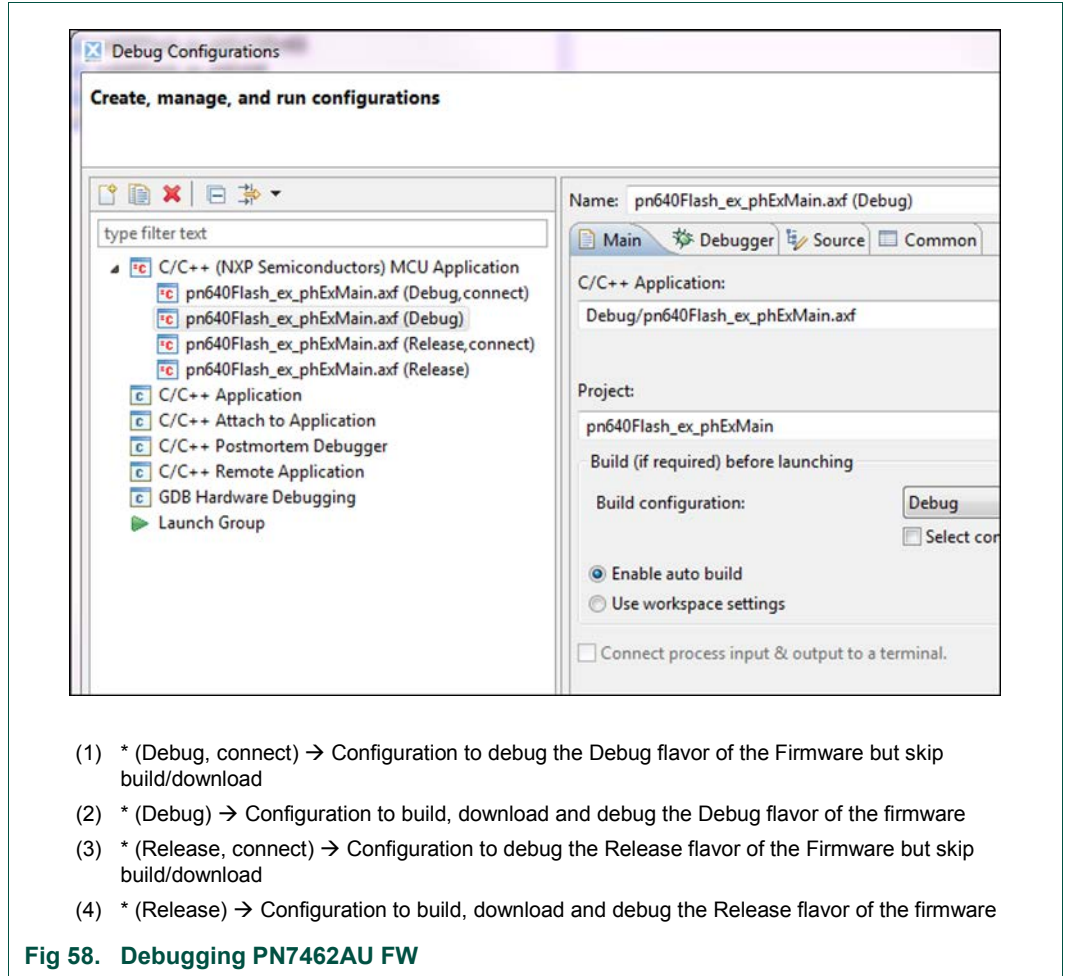
Fig 56. HW setup for the flashing via LPC-Link2



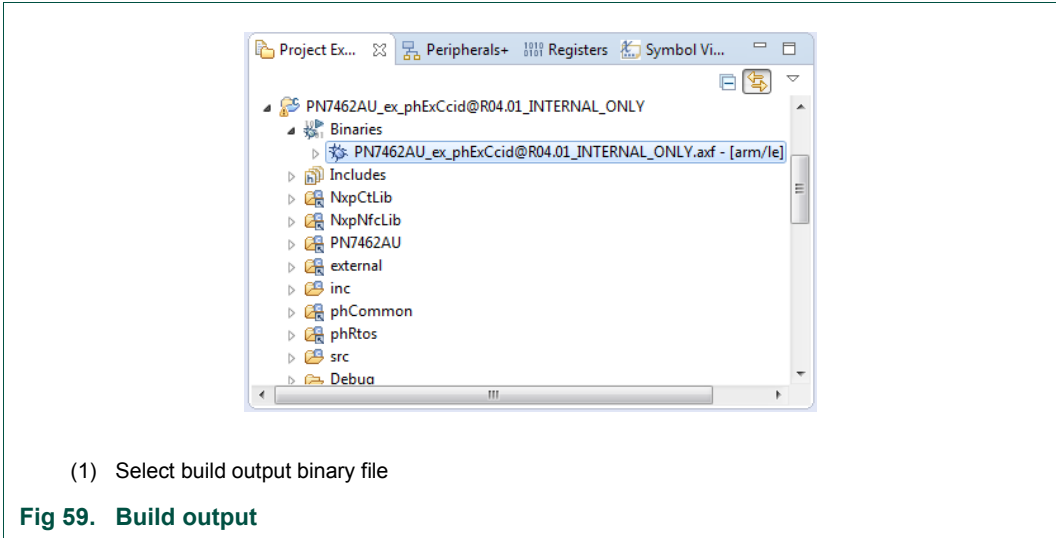
(1) For first time debug session, select Debug configurations

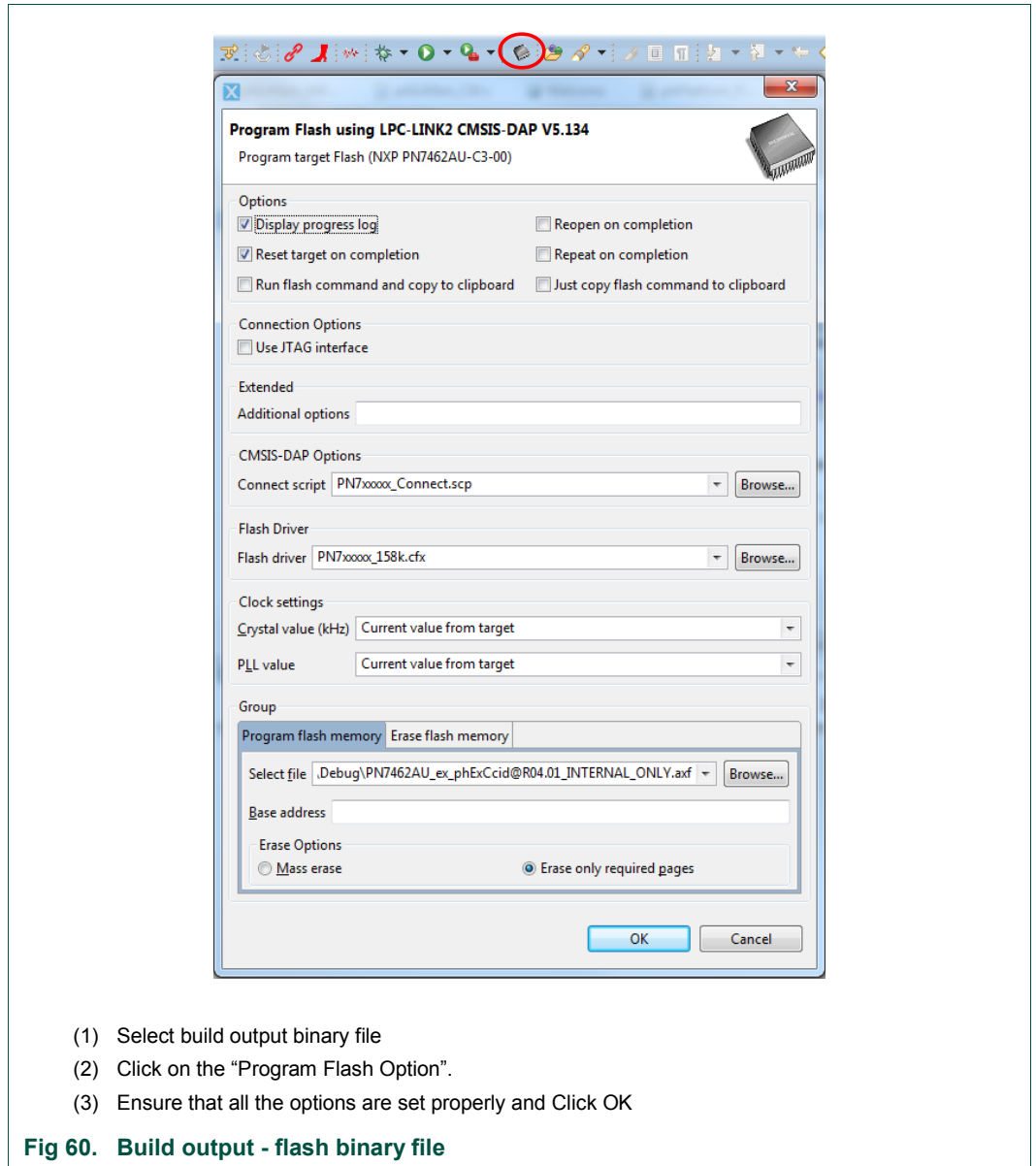
Fig 57. Selecting debug configuration





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.





- (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 60. Build output - flash binary file**

## 6.9 Updating Flash via USB MSD interface

PN7462AU can update FLASH contents via USB Mass Storage interface (primary download mode).

To mount PN7462AU IC as USB Mass storage drive (The steps mentioned here refer to the 3.2.1.1)

- Ensure that “HIF selection” is USB, see Fig 61
- USB Port of PC is connected to the USB socket on the customer demo board
- Press “RST\_N” switch.

- Press “DWL\_REQ” switch.
- Release “RST\_N” and keep holding “DWL\_REQ”.
- Release “DWL\_REQ” after a few seconds.

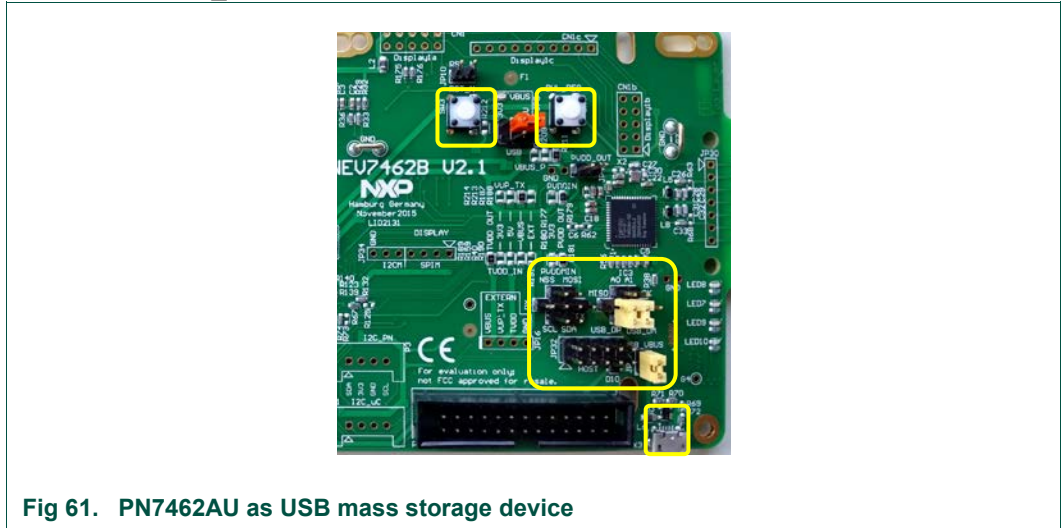


Fig 61. PN7462AU as USB mass storage device

Now the customer demo board is detected by PC as USB MSD (Mass Storage Device).

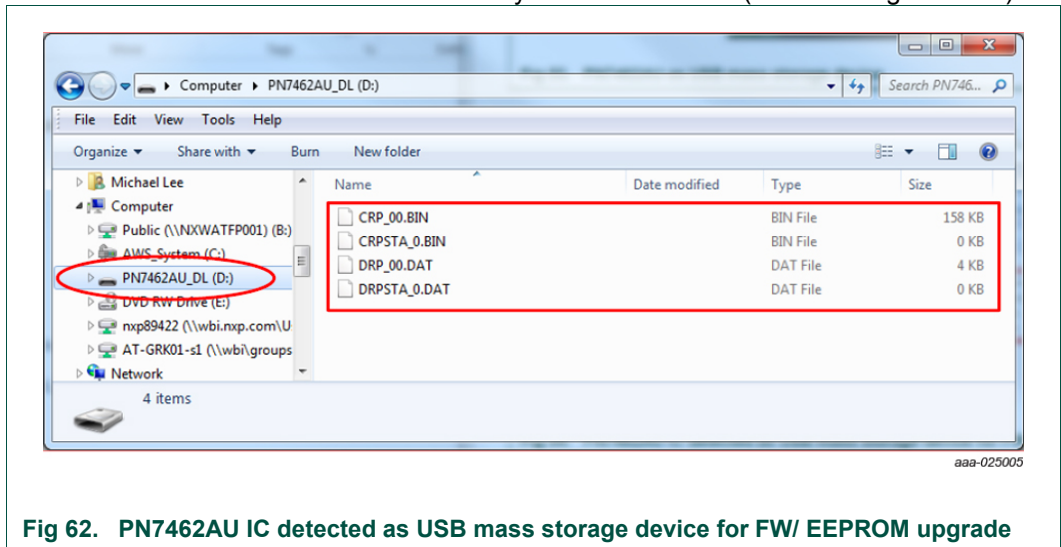


Fig 62. PN7462AU IC detected as USB mass storage device for FW/ EEPROM upgrade

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

Table 4. Files found in USB mass storage

File	Description
CRP_<nn>.BIN	PN7462AU’s user flash code (see Table 5 for description of <nn>)
CRPSTA_<s>.BIN	Status of previous write operation to user flash (see Table 6 for description of <s>)

File	Description
DRP_<nn>.DAT	PN7462AU's user EEPROM date (see Table 5 for description of <nn>)
DRPSTA_<s>.DAT	Status of previous write operation to user EEPROM (see Table 6 for description of <s>)

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

**Table 5. 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 6. 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)

To effectively update the user flash via USB once the device is in USB Mass storage mode the following the instructions are to be followed:

- Navigate to the newly mounted PN7462AU drive
- Delete CRP\_<nn>.bin file
- Copy the Flash Binary file to the new Drive. As part of project build via LPCXpresso, the user Flash Binary is created. (See Fig 59).
- 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”.

#### 7.1.1 Application messages – debug printouts

The most of the examples supporting debug messages printouts via the LPC-Link2 debug probe directly to the LPCXpresso IDE. The LPC-Link2 probe needs to be connected to the customer board by SWD (Fig 56) and example should be in debug 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”.

#### 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 63 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 63. LEDs status**

## 7.2 PN7462AU\_ex\_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 Demo setup

This section describes in detail the setup and execution environment required for *phExMain* application.

The following devices are required to run the example:

- PN7462AU customer board v2.1
- LPC-Link2 board
- Power adapter
- Contactless cards Type A, Type B, Type F
- Contact card ISO7816 compatible

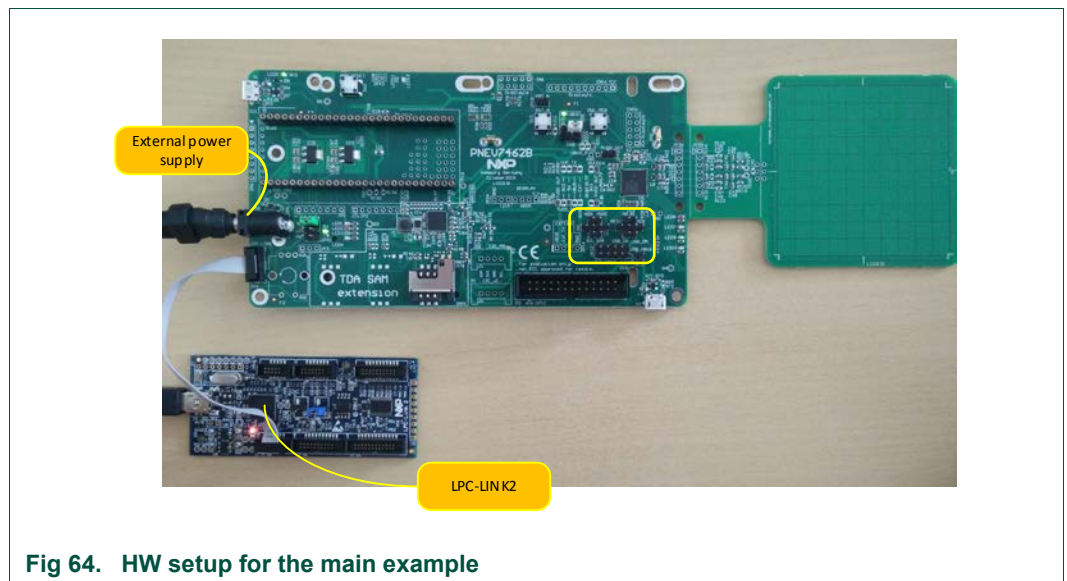


Fig 64. HW setup for the main example

### 7.2.2 Features

“*phExMain*” 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	Yes
HIF/MIF Interface	No

### 7.2.3 FreeRTOS usage by the example

Example can be built in two configuration modes, with FreeRTOS and without FreeRTOS support.

By setting precompile directive “#PHFL\_HALAPI\_WITH\_RTOS” or “#PHFL\_HALAPI\_NO\_RTOS” the mode of the configuration is specified.

To build example in one or another mode, comment/uncomment proper directive in the “APP\_NxpBuild.h” file.

In case of FreeRTOS mode, 3 tasks are used to control application flow:

- System Task which switches between CLIF functionality and CT functionality and handles system functions such as going to standby
- CLIF Task which executes NFC A, B , F reader application or EMV application and discovery loop
- CT Task which does an activation of a JCOP contact card, selects the T=1 protocol and executes EMV Card application.

In case of non RTOS support one main task is taking control on the application flow.

### 7.2.4 Operation with standby and without standby

Based on the compile time options, system task is responsible for managing different operation modes. The application can be in:

- Standby mode with wakeup timer and CT presence as wakeup configuration.
- Full power mode with GP Timer1 and CT presence interrupt enabled.

The standby feature can be enabled/disabled at compile time with the precompile directive. To enable standby mode “#PHFL\_ENABLED\_STANDBY” directive needs to be uncomment and can be found in the “APP\_NxpBuild .h” file.

```
#define PHFL_ENABLE_STANDBY
```



In this configuration, the FW by default puts the IC to standby and enables the wakeup sources such as contact card presence, wakeup timer and RF level detector (only for listen mode). The wakeup timer duration is taken from EE configuration (by default: 300ms). The IC wakes up at every wakeup timer duration and polls for contact card presence and if not present, polls for contactless technologies (or LPCD).

When a contactless or contact card is detected, the FW executes the corresponding application and returns back to standby.

In case “Standby” feature is not enabled, the FW uses General Purpose Timer 1 as wakeup timer and CT presence interrupt to either branch to Contactless application or Contact application respectively.

The below 2 diagrams explain the Standby and Non Standby scenarios. See Section 5.7 for further reference regarding RTOS.

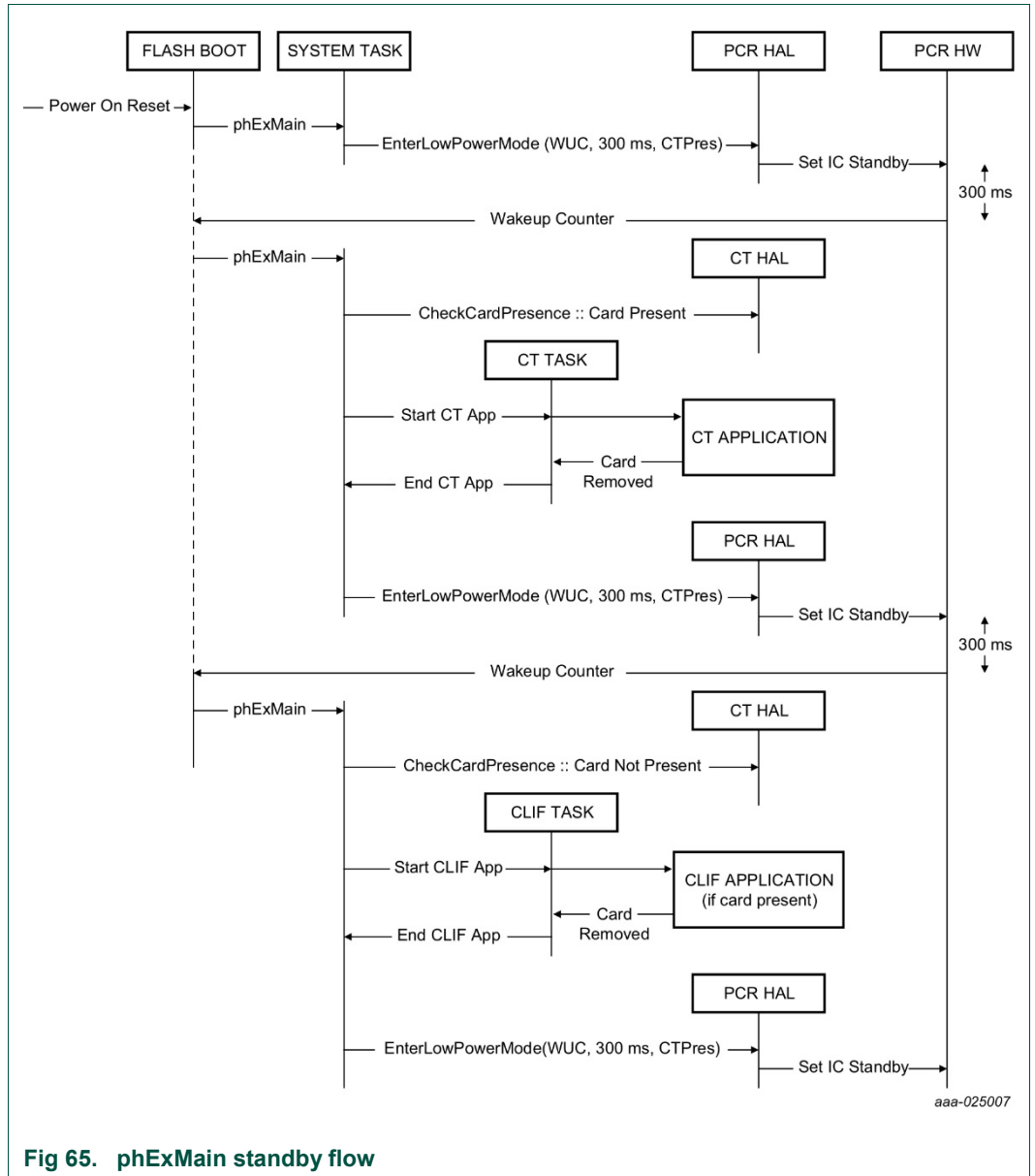


Fig 65. phExMain standby flow

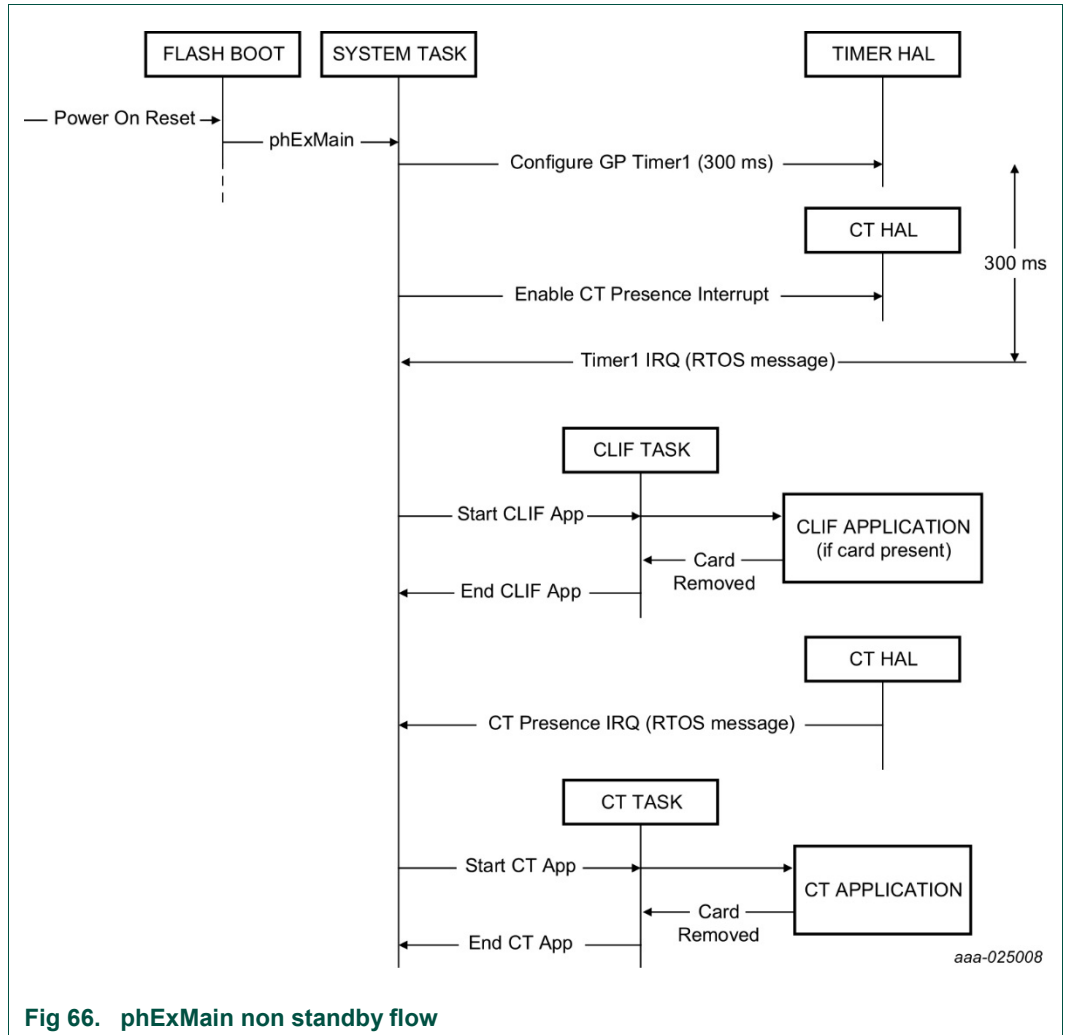


Fig 66. phExMain non standby flow

### 7.2.5 MF Classic

The MF Classic example is implementing basic read/write functionality using a MF Classic card with a predefined default key. If a Type A card is detected with SAK of 0x8 (1K MFC card) or 0x18 (4K MFC card), then the MF Classic example is executed. The example performs initial authentication of block X and then reads/writes to this sector. Further the example performs authentication of another block (say X +1) using the session key established during initial authentication (this is called re-authentication). It then performs read/write to this block X+1. See the function `phExMain_MiFareClassic()` and `phExMain_MifareOperations()`

Supported Functionality:

- Authentication & Re-Authentication
- Read/Write

Implementation in the `phExMain_MiFareClassic.c` file.

### 7.2.6 MF Ultralight

The MF Ultralight example is implementing basic read/write functionality using a non-secure MF Ultralight card. If a Type A card is detected with SAK 0x00, then MF Ultralight example is executed.

The example performs read and write to predefined pages of the card.

Supported functionality:

- READ
- WRITE
- Since the stack also supports Type 2 tag, a check is performed to see if the card is NDEF tag or ultralight tag

Implementation in the *"phExMain\_MiFareUltralight.c"* file.

### 7.2.7 MF DESFire

If the detected SAK is 0x20 (expected card is MFDF EV1), then ISO14443-4 Type A reader example is executed. The MF DESFire example implements L4 exchange of "GetVersion" command at 106, 212, 424 and 848 kbps. No other commands are currently exchanged as they require authentication and crypto operations (which are not currently supported in the release).

Supported functionality:

- Get Version
- 106/212/424/848 kbps
- No Encryption

Implementation in the *"phExMain\_TypeA\_L4Exchange.c"* file.

### 7.2.8 Jewel reader

If the ATQA of a Type A card denotes jewel card, the jewel example is executed.

The jewel example assumes a non-secure jewel card. The example performs read and write to predefined blocks of the card.

Supported functionalities:

- Read
- Write
- Since the stack also supports Type 1 tag, a check is performed to see if the card is an NDEF tag or jewel card.

Implementation in the *"phExMain\_Jewel.c"* file.

### 7.2.9 ISO15693 – ICODE SLIX

The example performs read and write to predefined blocks of the card.

Supported functionality:

- READ SINGLE BLOCK
- WRITE SINGLE BLOCK
- 26kbaud Tx (1out of 4 coding) and 26kbaud Rx

Implementation in the *"phExMain\_ISO15693.c"* file.

### 7.2.10 ISO18000-3.3 – ICODE ILT

The example performs read and write to predefined blocks of the card.

Supported functionality:

- READ BLOCK
- WRITE WORD
- TX TARI = 9.44 and RX 424\_2 Manchester Period

Implementation in the *"phExMain\_ISO18000p3m3.c"* file.

### 7.2.11 Type B eZLINK/ SLE card

If the detected technology is Type B, then ISO14443-4 Type B reader is executed.

This example is used demonstrate the ISO144434 exchange of APDUs to a Type B card at all baud rates.

Supported functionality:

- Get Challenge
- 106/212/424/848 kbps
- No encryption

Implementation in the *"phExMain\_TypeB\_L4Exchange.c"* file.

### 7.2.12 Type F (FeliCa tag)

This example is used to read and write FeliCa frames to FeliCa tags at 212/424 kbps.

Since the stack supports Type 3 tags, check is performed to see if the card is a NDEF tag or FeliCa card.

Supported functionality:

- CHECK
- UPDATE
- 212/424 kbps

Implementation in the *"phExMain\_FeliCa.c"* file.

### 7.2.13 ISO14443-4 card mode (till activation)

During listen, the example can be configured to either act as ISO14443-4 card emulator (SAK = 0x20) or NFC-DEP Target (SAK = 0x40). This configuration is done via #define macro in *phExMain\_Clif.h*

If the SAK is 0x20, discovery loop detects peer ISO14443A reader, the `phExMain_CardMode` is executed, that responds to RATS from the reader. This example does not demonstrate L4 APDUs exchange (it is done in `phExHCE` and `phExNFCForum`).

#### 7.2.14 Passive and active ISO18092 initiator (till activation)

During active poll mode, if `ATR_REQ` is received or during passive poll mode, if SAK denotes 0x40, then the example implemented in `phExMain_PasIni.c/ phExMain_ActIni.c` is executed. These examples simply transmit a `DEP_REQ` command with arbitrary payload to peer target. The purpose of this example is only to demonstrate integration of ISO18092.

#### 7.2.15 Passive ISO18092 target (till activation)

During listen, the example can be configured to either act as ISO14443-4 card emulator (SAK = 0x20) or NFC-DEP Target (SAK = 0x40). This configuration is done via `#define` macro in `phExMain_Clif.h`.

If the SAK is 0x40, discovery loop detects peer ISO18092 initiator, the `phExMain_PasTgt` is executed, that responds to `ATR_REQ` from the initiator. This example further waits for a NFC-DEP frame from the initiator.

#### 7.2.16 Contact Example

If the IC boots because of CT presence wakeup reason or if the CT presence interrupt is generated, the System task starts the CT task. The CT task performs the Contact application. The contact application activates the card, and determines the card is of EMVCo payment card or nonpayment card. If payment card is detected, the application further communicates with the card to know which type of card (Master card, VISA or AMEX card), and prints the information.

Supported ISO7816 Functionality:

- ATR Parsing
- Create MF
- Create EF
- Select EF
- Write Binary
- Read Binary
- Delete EF
- SCOSTA Card
- TA1 = 97
- Class A ( DCDC always in double mode)

#### 7.2.17 RTOS task management

The `phExMain` example can be executed in both RTOS. In RTOS environment, three tasks are created.

1. System task
  - a. Creates CLIF task
  - b. Creates CT task
  - c. Waits for PMU/PCR exception events
  - d. Waits for CLIF Task completion if standby is enabled
  - e. Waits for CT Task completion if standby is enabled
  - f. Enter low power mode (standby)
2. CLIF task
  - a. Starts GP timer for listen duration if standby is **not** enabled and wait for GP timer expiry
  - b. Configure external RF on detection
  - c. If boot reason is WUC counter or GP timer expiry, perform polling mode of discovery loop
  - d. If boot reason is RFLD or external RF is detected, perform listen mode of discovery loop
  - e. Notify system task if polling/listening is completed and standby is enabled
3. CT task
  - a. Enable CT presence interrupt and wait for CT presence interrupt
  - b. If boot reason is CT presence or CT presence interrupt is detected, perform CT example
  - c. Notify system task if polling/listening is completed and standby is enabled

Fig 67 and Fig 68 illustrate one instance of phExMain execution for both standby and non-standby scenarios.

- Please note that the CLIF and CT tasks are independent and can concurrently operate the CL and CT interfaces. During such concurrent operation, there is a possibility that CT interface may be unstable. It is up to the application design to configure interrupt and task priorities for a stable operation.

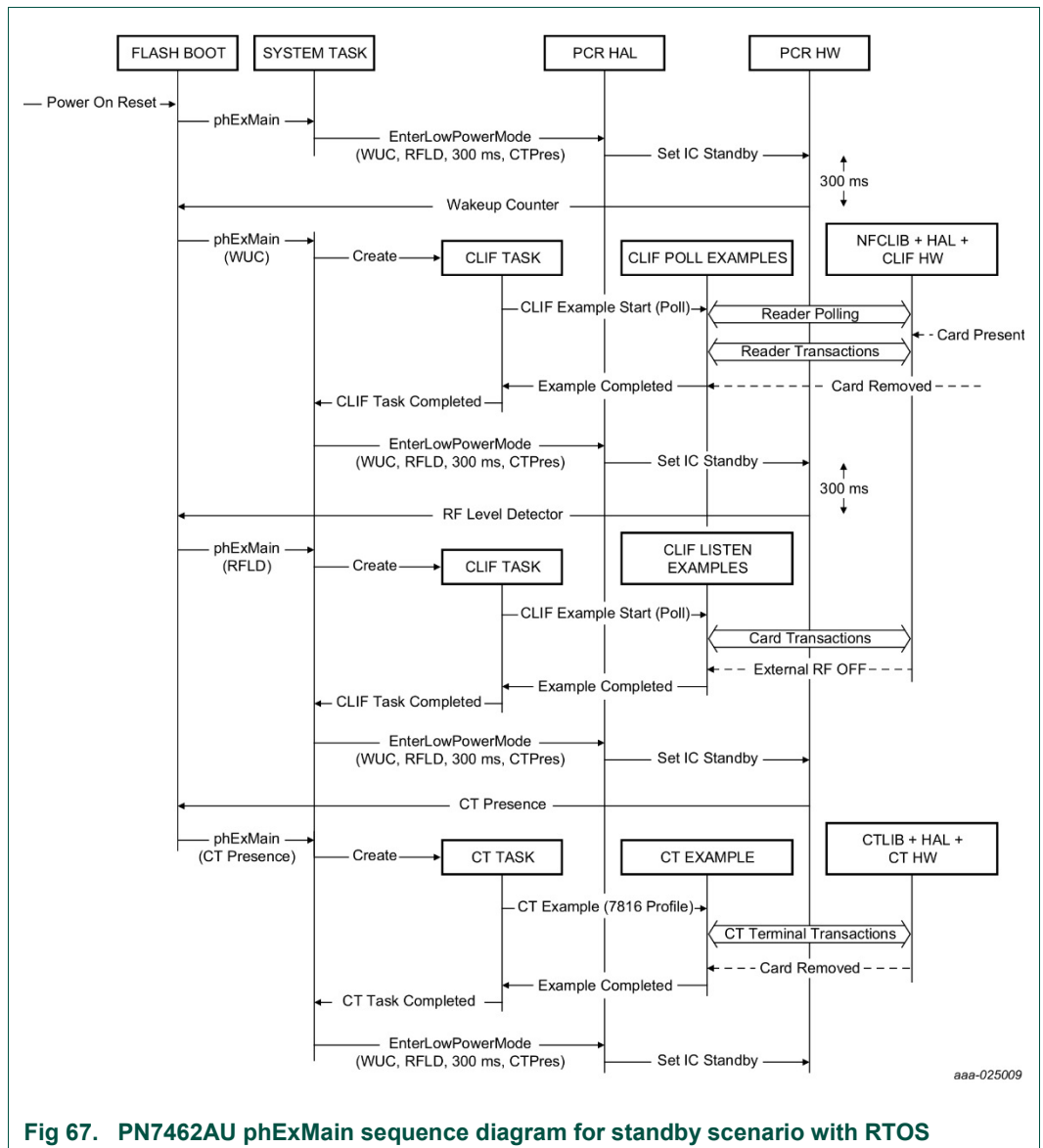


Fig 67. PN7462AU phExMain sequence diagram for standby scenario with RTOS



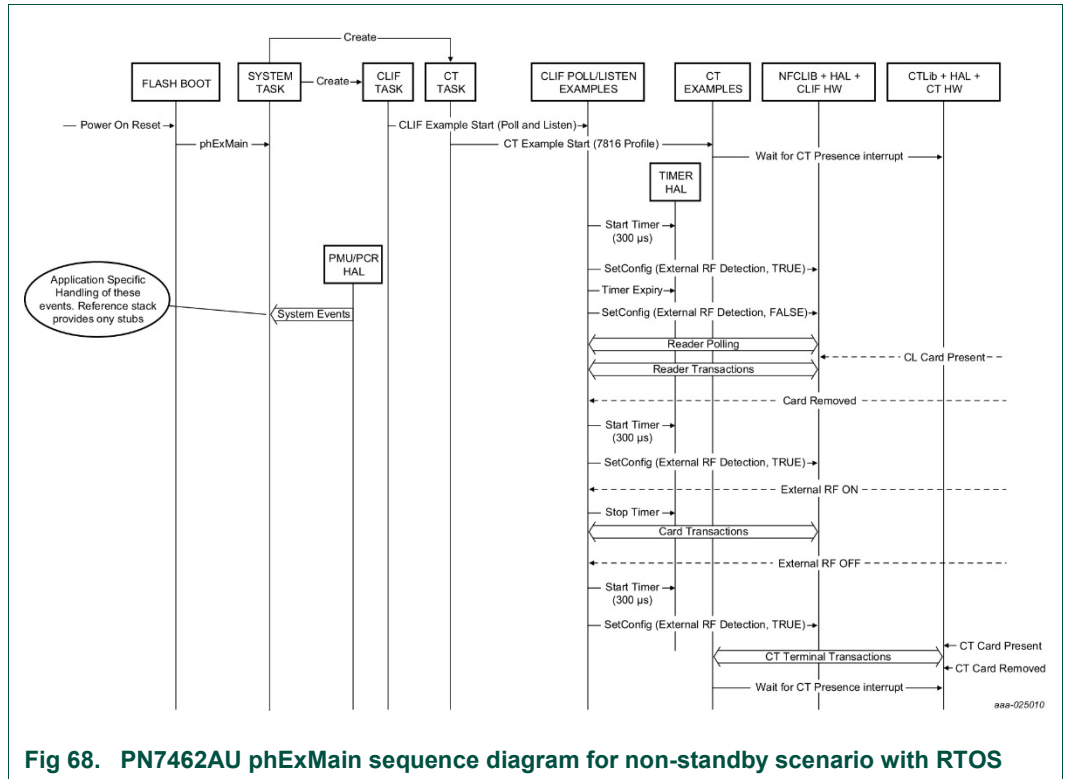


Fig 68. PN7462AU phExMain sequence diagram for non-standby scenario with RTOS

### 7.2.18 No-RTOS management

In case of No-RTOS the entry point from flash boot is *phExMain\_NoRTOS*. The functionality remains the same except that the CLIF example and the CT examples are called from a single executive while loop based on timer interrupt or external RF detection or CT presence interrupt.

### 7.3 PN7462AU\_ex\_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 Demo setup

This section describes in detail the setup and execution environment required for the “*phExEMVCo*” application.

The following things are required for setup:

- PN7462AU customer board v1.1
- LPC-Link2 board
- Power adapter
- Contact and contactless EMVCo card

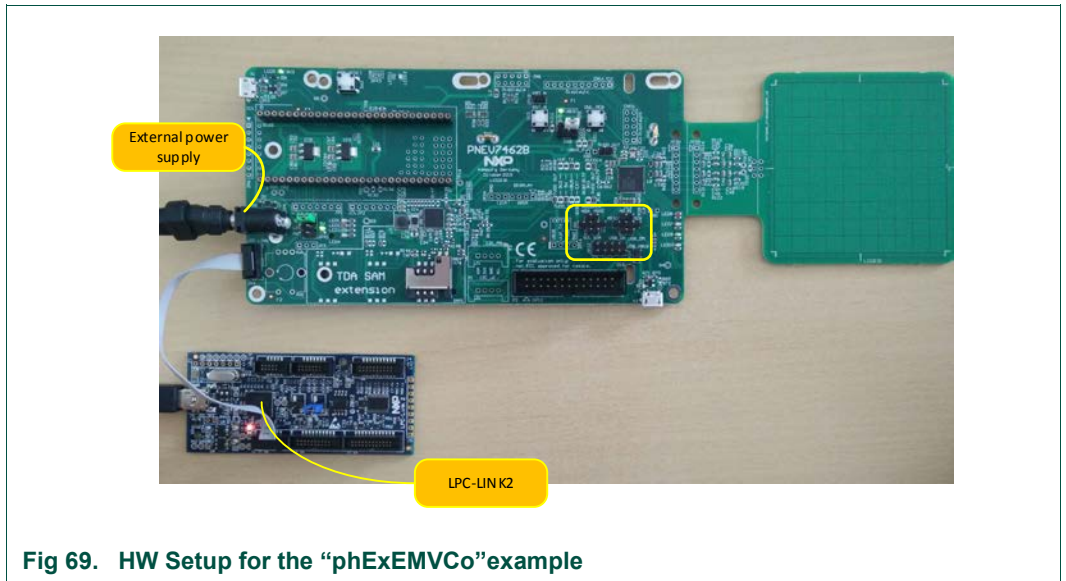


Fig 69. HW Setup for the “*phExEMVCo*” example

#### 7.3.2 Features

“*phExEMVCo*” example is covering next features:

Table 8. “*phExEMVCo*” Example features

Feature	supported
CLIF Interface	Yes
CT Interface	Yes
NXP NFC Reader Library	Yes
CT Reader Library	Yes

Feature	supported
FreeRTOS	Yes
Non RTOS	Yes
Standby mode	No
HIF/MIF Interface	No

### 7.3.3 EMVCo polling loop

In this example EMVCo polling loop is enabled. In this profile, only Type A and B technology polling is enabled and bail out is set such that both A and B techs are polled even if one of them is detected. This is to ensure no two cards of same/different tech are present in the POS for EMV transaction. Low Power Card Detection (LPCD) is disabled in this profile.

### 7.3.4 EMV transaction

This example implement next EMV Transactions:

- SELECT (PPSE)
- SELECT command
- GET PROCESSING OPTIONS
- READ RECORD
- GENERATE AC

Supported EMV functionality:

- ATR Parsing accordingly to the EMV specifications
- 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 double mode)

### 7.3.5 RTOS task management

- For information regarding RTOS task management, refer to Section 7.2.17

### 7.3.6 No RTOS management

- For information regarding No RTOS management, refer to Section 7.2.18

### 7.4 PN7462AU\_ex\_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 Demo setup

This section describes in detail the setup and execution environment required for “*phExRf*” application.

The following devices are required to run the example:

- PN7462AU customer board v2.1
- LPC-Link2 board
- Power adapter
- Contactless cards Type A, Type B, Type F, ISO15693
- NFC Enabled Phone

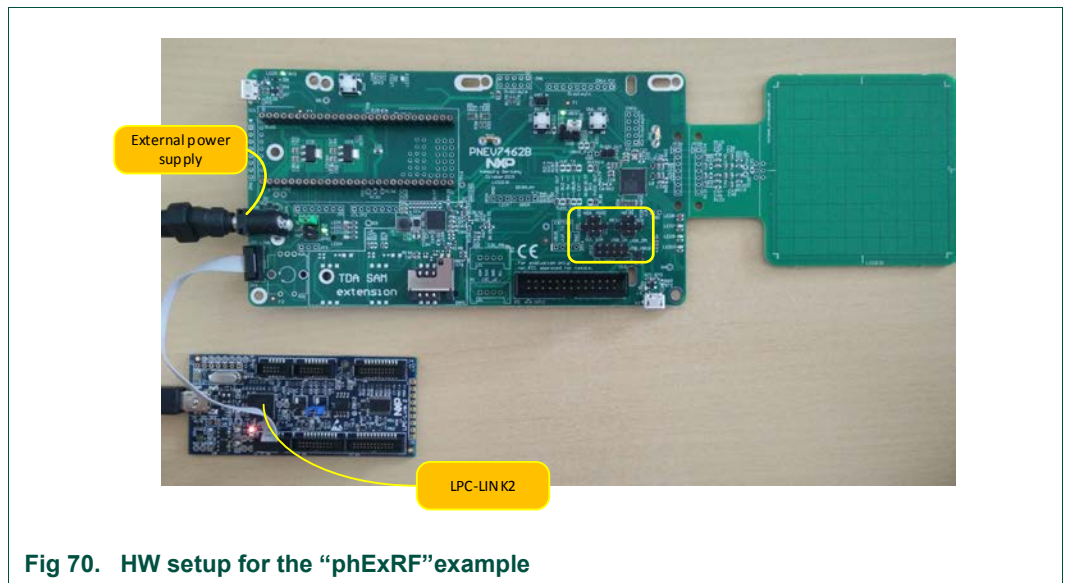


Fig 70. HW setup for the “phExRF” example

#### 7.4.2 Features

“*phExRF*” example is covering next features:

Table 9. “phExRF” Example features

Feature	supported
CLIF Interface	Yes

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

### 7.4.3 Application Flow

The figure below demonstrate the application flow.

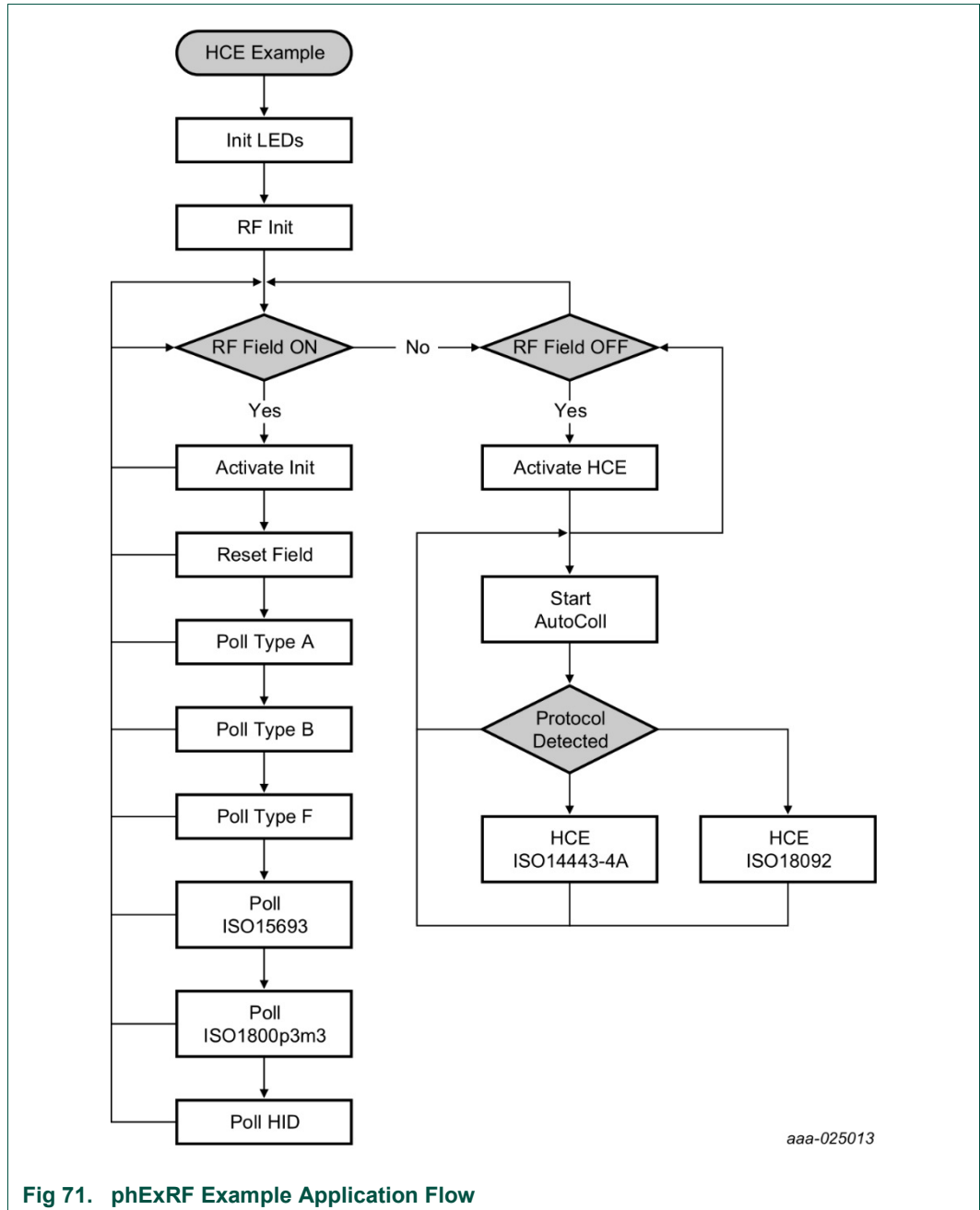


Fig 71. phExRF Example Application Flow

### 7.5 PN7462AU\_ex\_phExRFPoll example (CL functionality)

The “*phExRfPoll*” is an example which implements the polling for contactless cards without NFC Reader Library support for the optimization of the RF-signal. Through defines in *phExRfPoll.c* the specific technology to poll for can be selected.

By setting breakpoints RF registers can be modified to change the signal.

## 7.6 PN7462AU\_ex\_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. *phExCt* performs activation of an EMVCo card. SELECT Master card Adu is sent depending on the protocol supported by the card and expects RAPDU 0x90 0x00.

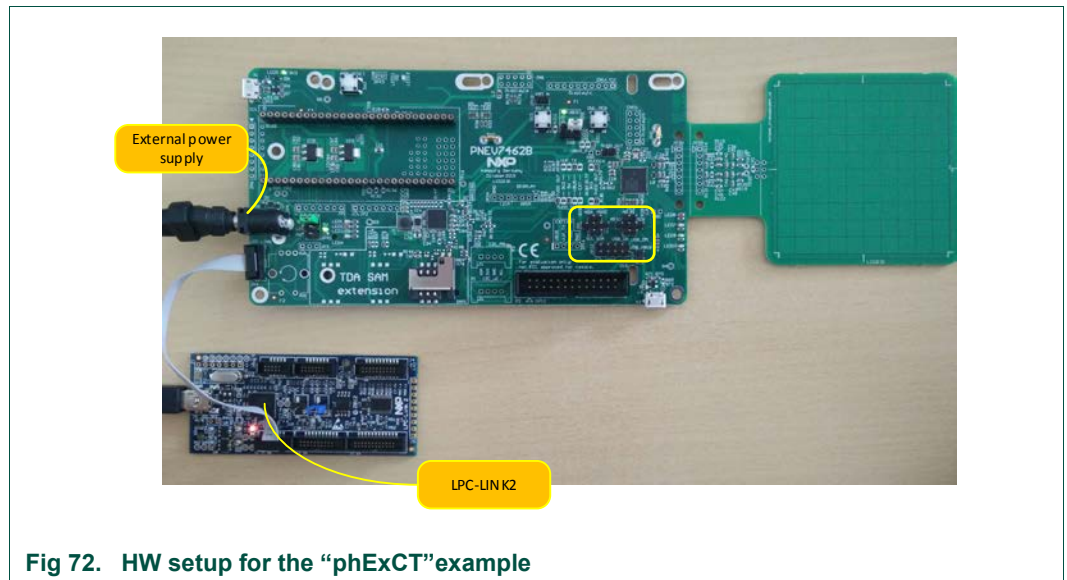
The example is also capable of determining the non-EMVCo card or non-Master card. After the transactions, deactivation is performed. Also, this example demonstrates the non-RTOS integration of application with HALs.

### 7.6.1 Demo setup

This section describes in detail the setup and execution environment required for “*phExCT*” application.

The following devices are required to run the example:

- PN7462AU customer board v2.1
- LPC-Link2 board
- Power adapter
- Contact card ISO7816 compatible



## 7.6.2 Features

“*phExCT*” example is covering next features:

**Table 10. “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

## 7.6.3 Application Flow

The figure below demonstrate the application flow.



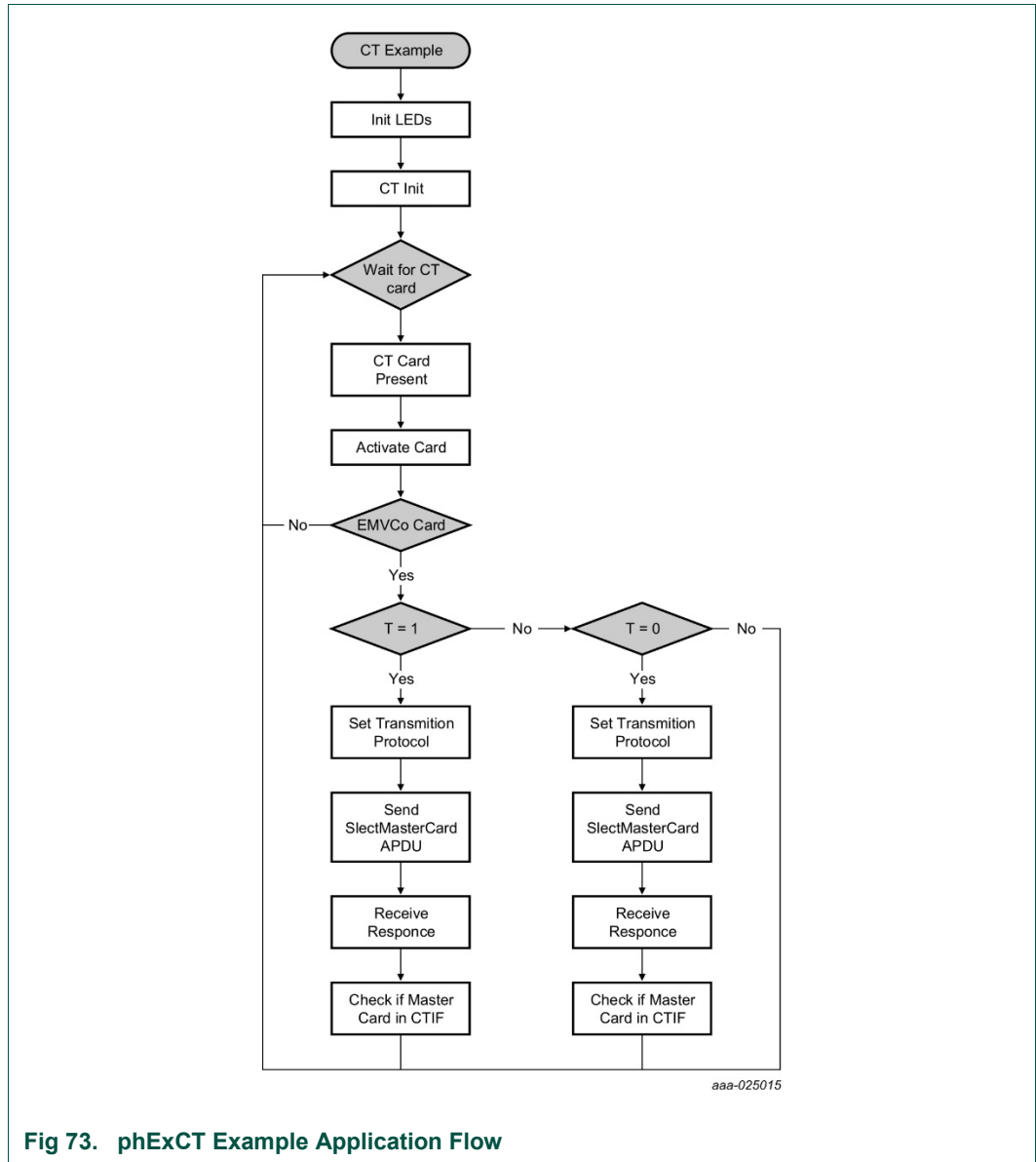


Fig 73. phExCT Example Application Flow

**7.6.4 EMVCo activation**

The example performs EMVCo activation and EMVCo ATR parsing. It also determines the protocol supported by the card.

**7.6.5 SELECT master card**

The example sends a SELECT master card APDU and expects a RAPDU 0x90 0x00.

## 7.7 PN7462AU\_ex\_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. After the transactions, deactivation is performed. The example is also capable of determining the Non-EMVCo card.

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

### 7.7.1 Demo setup

In this example the same setup is used as in “phExEMVCo” example.

### 7.7.2 Features

“*phExCT*” example is covering next features:

**Table 11. “phExRf” Example features**

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

### 7.7.3 EMVCo activation

The example performs an EMVCo activation and EMVCo ATR parsing. It also determines the protocol supported by the card and applies the protocol supported.

### 7.7.4 Apdu transactions

The example is capable of sending select commands for nine pre-selected types of EMVCO cards after successful activation.

4. Master Card : Credit or Debit (tested)
5. Visa Card : Credit or Debit (tested)
6. Master Card : Maestro (debit card)
7. Master Card : Cirrus (inter-bank network)
8. Master Card : Maestro UK
9. Visa Card : Electron card
10. Visa Card : V PAY card
11. Visa Card : VISA Plus card
12. Amex Card (tested)

## 7.8 PN7462AU\_ex\_phExCT7816 example (CT functionality)

The “PN7462AU\_ex\_phExCt7816” example implements the CT functionality with CT library support. Application use CT library APIs and perform same CT functionality as in “PN7462AU\_ex\_phExMain” example with the only difference that for the transaction static predefined packets are used. The example is built to work on a SCOSTA card. PN7462AU\_ex\_phExCT7816 demonstrates the CT Protocol Lib + HAL API's. After the transactions, deactivation is performed.

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

### 7.8.1 Demo setup

In this example the same setup is used as in “phExMain” example.

### 7.8.2 Features

“phExCT7816” example is covering next features:

**Table 12. “phExCT7816” Example features**

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

### 7.8.3 ISO7816 activation

The example performs an ISO7816 activation and ISO7816 ATR parsing. Also, it determines the protocol supported by the card and applies the protocol supported.

### 7.8.4 APDU transactions

The following APDU's are sent after the activation of the card. If the card supports the following APDU's (e.g. SCOSTA), proper responses will come from the card.

- Create MF
- Create EF
- Select EF
- Write binary
- Read binary
- Delete EF

### 7.9 PN7462AU\_ex\_phExHif example

This example demonstrates host interface loopback 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.

#### 7.9.1 Demo setup

This section describes in detail the setup and execution environment required for the “phExHif” application.

The following things are required for setup:

- PN7462AU customer board v1.1
- LPC1769 board
- LPC-Link2 board
- Power adapter

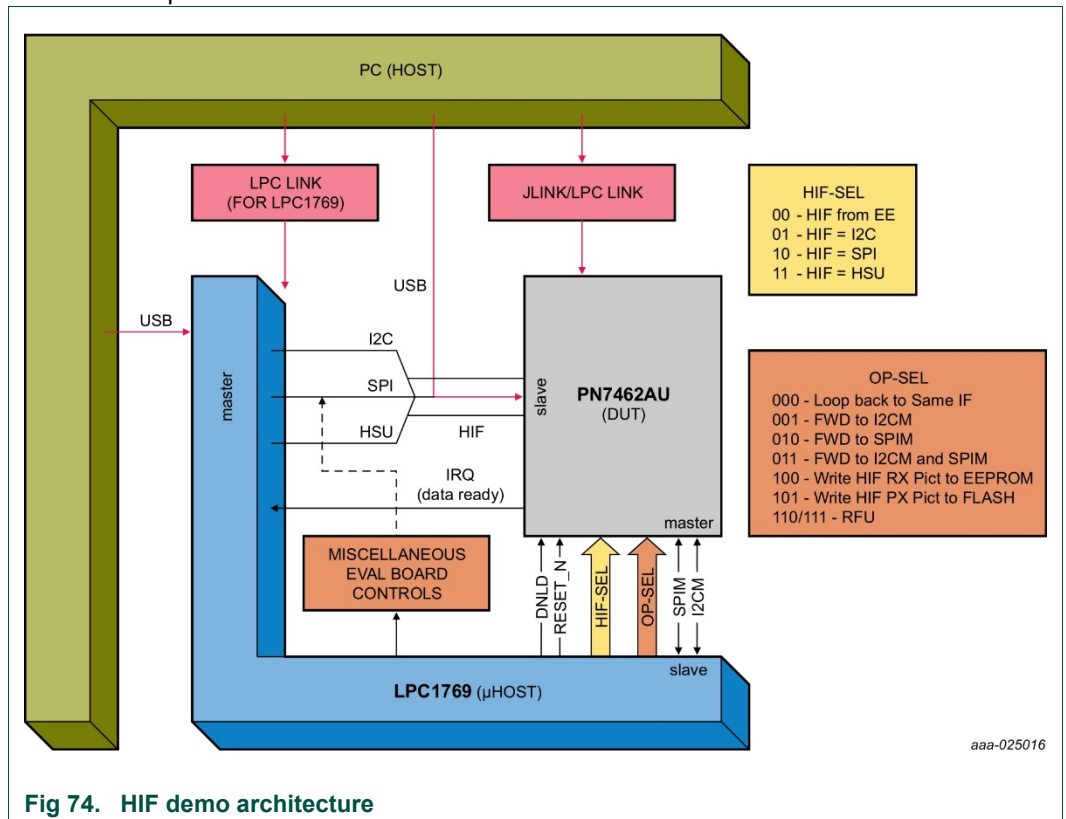


Fig 74. HIF demo architecture

The GPIOs of LPC and PN7462AU are used to determine which functionality of this example has to be executed. It is also used to select the host interface or master interface to be used. For each different functionality, a different LPCExpresso project is

required for LPC1769 side while PN7462AU side has only the phExHif LPCExpresso project.

The phExHIF indicates its readiness to LPC1769 through GPIO1 of PN7462AU connected to GPIO0.0 of LPC1769 (APP ready pin).

### 7.9.2 Features

“phExHif” example is covering next features:

**Table 13. “phExHif” Example features**

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

### 7.9.3 HIF selection

**Table 14. Host interface selection**

GPIO5_PN7462AU ← GPIO2.0_LPC	GPIO4_PN7462 ← GPIO2.1_LPC	Chosen HIF
0	0	Invalid
0	1	I2C
1	0	SPI
1	1	HSU

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

### 7.9.4 Operation selection

The tabulated GPIO configuration selects the operation to be performed by examples shown in Table 15.

**Table 15. Operation on the packet received on HIF**

GPIO8_PN7462AU ← GPIO2.2_LPC	GPIO7_PN7462AU ← GPIO2.3_LPC	GPIO6_PN7462AU ← GPIO2.4_LPC	Chosen HIF
0	0	0	Loopback on HIF
0	0	1	Forward HIF Rx Packet to I2CM Tx

GPIO8_PN7462AU ← GPIO2.2_LPC	GPIO7_PN7462AU ← GPIO2.3_LPC	GPIO6_PN7462AU ← GPIO2.4_LPC	Chosen HIF
0	1	0	Forward HIF Rx Packet to SPIM Tx
0	1	1	Forward HIF Rx Packet to SPIM & I2CM Tx Both
1	0	0	Program EEP with HIF Rx Packet
1	0	1	Program FLASH with HIF Rx Packet
1	1	0	Forward HIF Rx Packet to CT
1	1	1	RFU

Application ready PIN is connected with GPIO0.0\_LPC on the LPC board.

Important guidelines:

- 1 → Logical HIGH as seen/set by the GPIO.
- 0 → Logical HIGH as seen/set by the GPIO.
- Start PN7462AU Application before launching LPC Application
- Since both the products are LPCXpresso IDE based, while updated FW Image via the LPC Link, Ensure that the image being downloaded is to correct product.

### 7.9.5 EEPROM configuration dependencies

Values from the following EEPROM structures are used in this example:

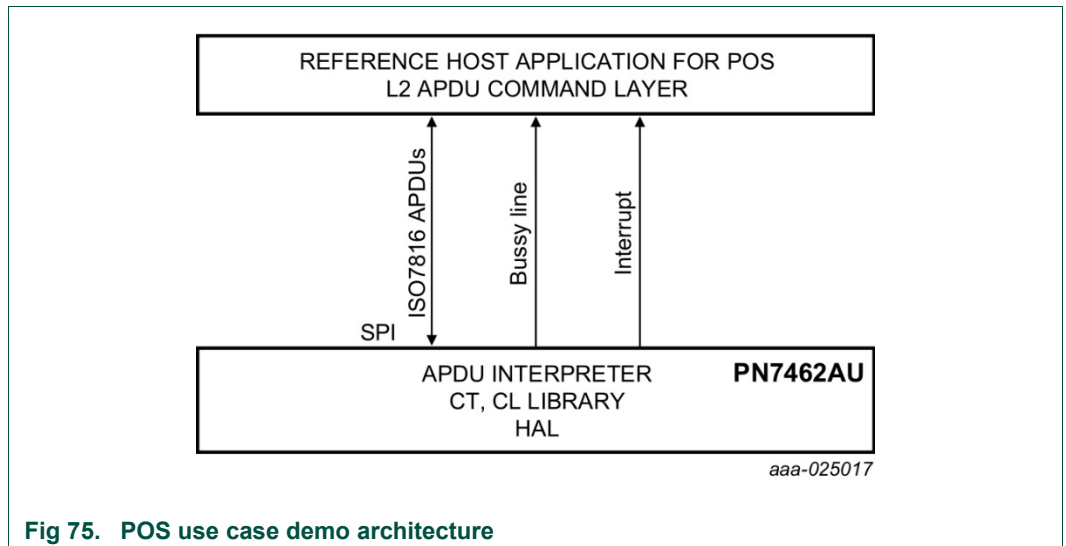
- Boot::EEPROM
- Boot::FLASH
- Boot::CT
- Boot::GPIO
- HW::I2CM
- HW::SPIM
- HW::HIF

**7.9.6 LPCEXpresso projects provided for LPC1769**

- LPCEXHif\_HSU\_LoopBack\_App
- LPCEXHif\_HSU\_to\_I2CM\_SPIM\_App
- LPCEXHif\_I2C\_Loopback\_App
- LPCEXHif\_I2C\_to\_SPIM\_App
- LPCEXHif\_SPI\_CT\_App
- LPCEXHif\_SPI\_LoopBack\_App
- LPCEXHif\_SPI\_to\_EEPROM\_App
- LPCEXHif\_SPI\_to\_FLASH\_App
- LPCEXHif\_SPI\_to\_I2CM\_App
- Supporting libraries
  - PN640\_lpc17xx\_lib, CMSISv2p00\_LPC17xx

**7.10 PN7462AU\_ex\_phExPos example**

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



**Fig 75. POS use case demo architecture**

The POS demo architecture is split into application layer (L2) and low level EMVCo compliant layer L1 which is hosted on the PN7462AU. The application layer L2 commands are simulated in reference microcontroller board (LPC1769) and L1 layer components are placed in PN7462AU.

The application APDU commands (L2) are communicated to PN7462AU through SPI host interface. PN7462AU GPIO pin is used to synchronize command / response between LPC1769 and PN7462AU.

Interrupt pin is used to notify valid ISO 14443-4 card to LPC1769.

*Note*  
Detailed description and how to use example is described in "POS Use Case Demo Setup Manual".

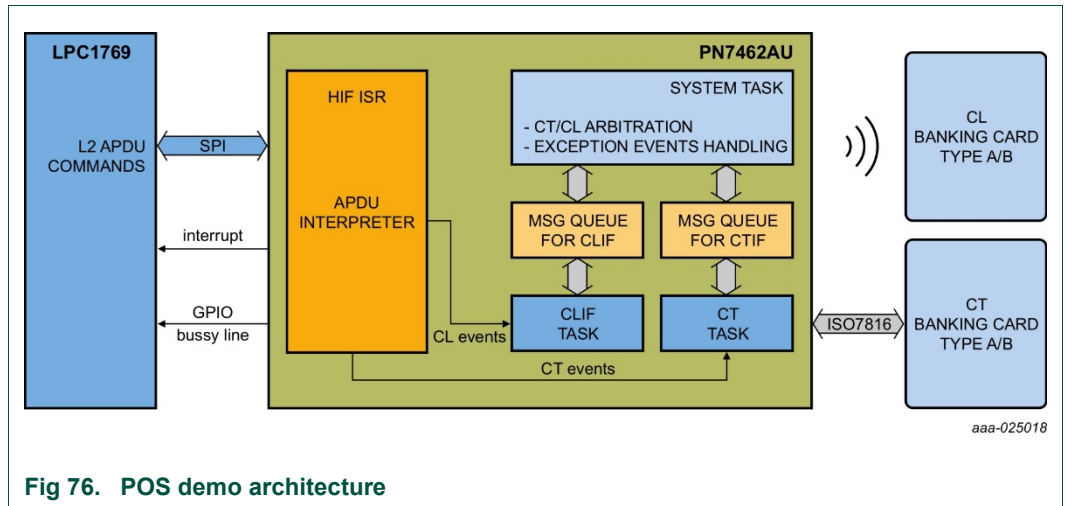


Fig 76. POS demo architecture

### 7.11 PN7462AU\_ex\_phExCcid example

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.



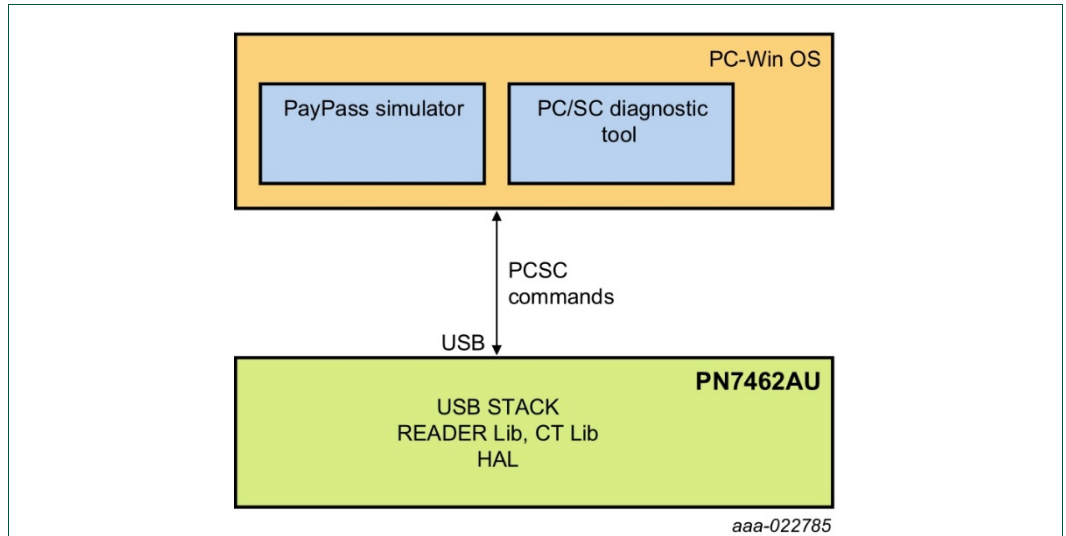


Fig 77. PC USB reader example block diagram

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.

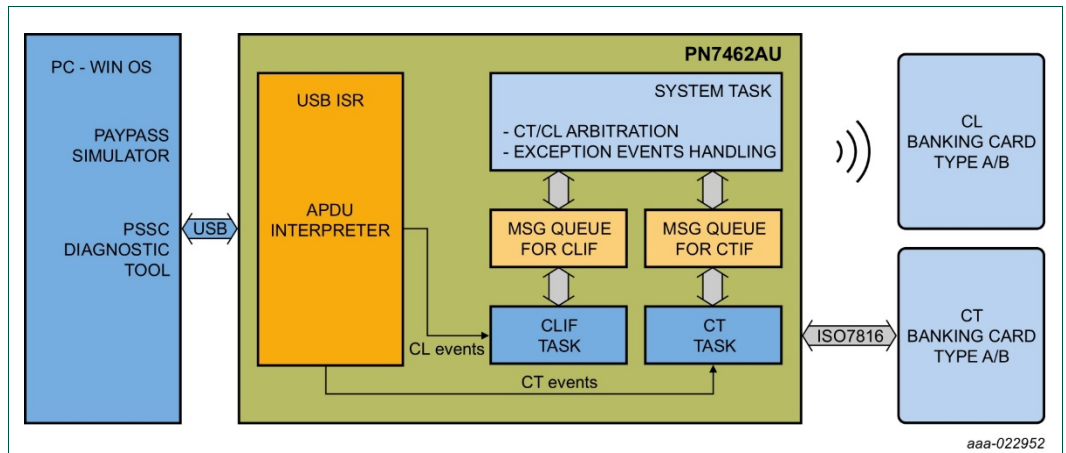


Fig 78. Example architecture

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

### 7.12 PN7462AU\_ex\_phSystemServices example

This example application demonstrates system services invocation. The PN7462AU provides ROM services that are accessible via flash APIs, also described in /PN7462AU/phROMIntf/phha1SysSer/inc/phha1SysSer.h and with more detailed description in API documentation.

This application requires user interface for performing the operations so it is needed to use debug mode. Some of the featured system service commands could be irreversible or reversible depending on the application mode configured by:

```
#define ENABLE_IR_REVERSIBLE_COMMANDS 0 //1
```

If the macro ENABLE\_IR\_REVERSIBLE\_COMMANDS is defined to 0, example will not run irreversible commands, if defined to 1 example will run irreversible commands but user confirmation is needed.

**Table 16. PN7462AU\_ex\_phSystemServices features**

Feature	description
SECROW Lock	The HW SecRow contains the SWD access bits, code write-protection bits and RSTN pin behavior bits. For blocking any further writes to SecRow, the phhalSysSer_OTP_SetSecrowLock() is used. It prevents further usage of phhalSysSer_OTP_SecrowConfig() function.
Code write protection	It is required to lock flash memory from write at HW level. It is locked possibly at a stage when secure secondary upgrade is not planned for the remaining lifecycle of the product. For such use cases, phhalSysSer_OTP_SecrowConfig() is used to lock flash memory from any further write. Any flash programming after locking the flash results in hard fault. Once SECROW functionality is locked, this feature cannot be used anymore.
Block SWD debugging	This command disables PN7462AU SWD debug interface. When the PN7462AU IC is delivered from production to user, the default SWD access level enables the user to view and debug user flash memory, user EEPROM memory, user RAM memory, and peripheral registers. The access level can be irreversibly changed to prevent view/debug access to any memory region or peripheral registers, before deploying the IC to the field. phhalSysSer_OTP_SecrowConfig() can be used to lock the SWD against any further access. Once SECROW functionality is locked, this feature cannot be used anymore.
Disable primary download	Command is used to irreversibly disable the ROM primary download feature. On subsequent boots, the ROM boot never enters ROM primary download mode, even if DWL_REQ pin and USB_VBUS pin is high. This feature is typically used after development and flashing of secondary downloader in the flash memory, for subsequent code/data upgrades.
Update Product ID	USB Product ID PID update
Update Vendor ID	USB vendor ID update
Perform In Application Programming	Application asks for FLASH page number. Page is 128bytes long, for 158kb of the flash memory, the page number is in range 0-1263. The selected flash page is updated from user programmable values.
Set internal PVDD	PVDD is pad voltage reference and supply of the host interface (HSU, USB, I2C, and SPI) and the GPIOs. This command sets PVDD configuration to internal.
Get ROM version	Commands returns current ROM firmware version

*Note:*

*For irreversible commands: Secrow lock, code write protection, block SWD debugging and disable primary download the undo is not possible!*

**7.13 PN7462AU\_ex\_phExVCom example**

This example application features TypeA card detection, RF filed control and communication with the PC host over USB CDC interface (VCOM).

**7.13.1 Demo setup**

Customer demo board is connected to the PC host via USB interface. USB micro socket X3 on the Customer demo board needs to be connected to the PC’s USB port. The proper USB host interface configuration on the Customer Demo board is described in 3.2.1.1. Application outputs debug traces to and receives commands from the terminal emulation application running on the PC host. Serial port parameters are 9600/8/1/N/1.

**7.13.2 Command set**

Commands are entered by the terminal emulation program. Each command is one character long.

- a) T command (character T) , example application enters Type A detection mode and the terminal output is as follows:

```

Poll command received
Entering TypeA Polling mode..
Type A Polling
Type A Polling
Card UID=0424566AF13B80
Card UID=0424566AF13B80
Card UID=0424566AF13B80
Type A Polling
Type A Polling
    
```

LED8-10 lit in circular pattern.

- b) O command (character O), example application turns RF filed permanently ON, the corresponding output on the terminal console is :

```

RF ON command received
LED9 (Green) and LED10 (Blue) lit.
    
```

- c) F command (character F) , example application turns RF filed permanently OFF, output on the terminal console is :

```

RF OFF command received
LED7 (Red) and LED8 (Yellow) lit.
    
```

**7.14 PN7462AU\_ex\_phExDoorAccess example**

The example application demonstrates card detection, card authentication and HSU interface communication with the PC host. The application is running a NFC polling loop and goes to standby mode after each polling loop in case no CL card is detected by the RF field of the reader. The polling loop is implemented for Type A, B, F, ISO15693 and ISO 1800-P3M3. The application prints out the detected type of the card and UID if

available. In case the NFC device is detected, the application sends a NDEF message containing the NXP webpage address.

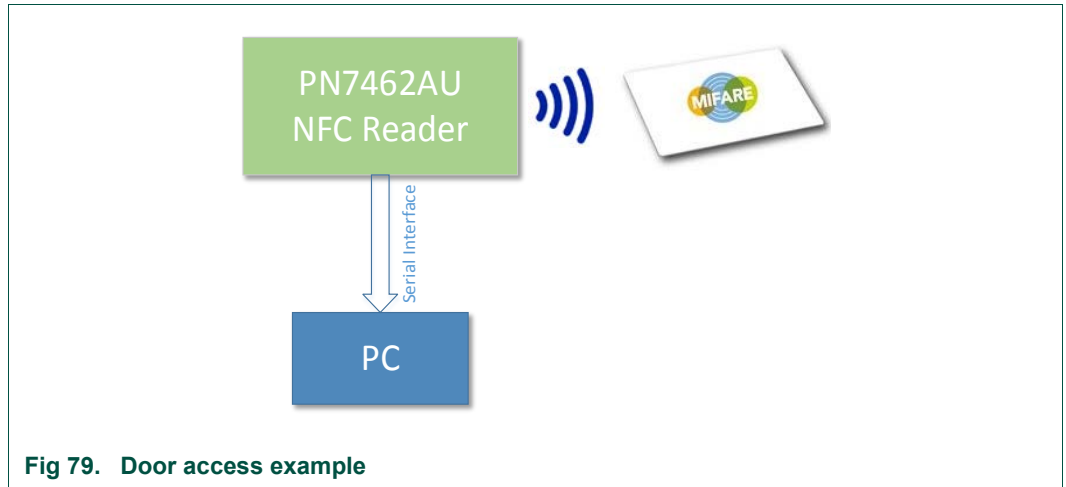


Fig 79. Door access example

After each polling loop the application goes to standby mode and remains for 500ms. A timer is used as a wakeup source from standby. This process continues until a card is detected by the RF field of the reader. If a card is detected, the card type with its UID is sent via HSU and will be printed on the PC console.

In case a MIFARE Classic card is detected, application tries to authenticate the card using the default MIFARE key. If the authentication is successful a block of data is read from the card. The type of the card, UID and data are sent via HSU and printed on the PC console.

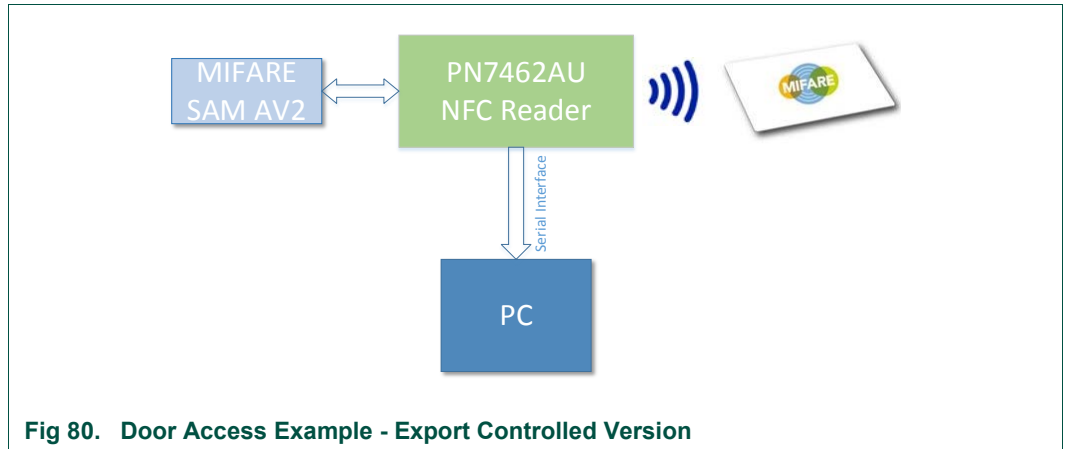
P2P functionality is integrated. When a NFC enabled phone is detected from the RF field as an active or passive target the LLCP SNEP will be activated and NDEF message will be sent to the mobile device.

If LPCD (Low Power Card Detection) is enabled, the reader checks during the wakeup time the presence of a card and enters the card detected mode when a card is present & repeats the cycle. If no card is present it goes back to standby mode. LPCD is enabled by default.

*Note:*  
Detailed description and how to use example is described in "Door Access User Manual".

### 7.15 PN7462AU\_ex\_phExDoorAccessEC example

This example is related to the 7.14 but in this version the application is using a MIFARE DESFire EV1 card for the authentication, data exchange is done over contactless interface and software key store or SAM (Secure Access Module) key store is used for storing the authentication key. By default, the software key store is enabled. The user can use the SAM key store by enabling corresponding. SAM is a key storage element and it should be inserted in the CT main.



**Fig 80. Door Access Example - Export Controlled Version**

On power up, NFC Reader starts polling for (PICC) cards (Type A, B or F, ISO15693, ISO18000-3M3) and if no card is present, the reader goes to standby mode. Timer is used as a wakeup source from standby - timer periodically every 500 ms. This process continues until a card is detected by the RF field of the reader. If a card is detected, the card type with its UID is sent via HSU & this is printed on the PC console

If a MIFARE DESFire EV1 card is detected, the reader tries to select a pre written custom application on the card. It tries to authenticate the card using the key stored in SAM. If SAM is not present, then software keys can be used for authentication. If authentication is successful a block of data will be read from the card.

*Note:*  
 Detailed description and how to use example is described in “Door Access User Manual”.  
 This example is available only with PSP package from NXP DocStore.

**7.16 PN7462AU\_ex\_phExNFCCcid example**

The NFC CCID is versatile demo application that features:

- Card detection for TypeA, TypeB, Felica, ISO15693, ISO18000p3m3 technologies
- Support for proprietary commands for Mifare Classic, Mifare UltraLight and Mifare UltraLightC for read and write.
- CCID USB device protocol implementation. Supports the Suspend Resume and Wakeup Feature.
- Communication of the CLIF information with the PC using a PCSC application.
- P2P passive initiator mode. Supports LLCP Initiator Mode for sending the NDEF message to the mobile.

Demo setup for this application is the same as described in 7.10

**7.17 PN7462AU\_ex\_phExMfCrypto example**

The PN7462AU\_ex\_phExMfCrypto demo application includes both CL and CT library components. Example is based on the Discovery Loop alongside Crypto layers and it is

intended to evaluate Mifare DesFire card functionalities. Application performs following operations:

- Application and Data and Value File creation inside the card.
- AES authentication of the application.
- Changing of the key.
- Enciphered Read of Value File and Plain Read of Data File.
- Enciphered Write of Value File and Plain write of Data File.
- Supports Different Keys for Read and write.
- CT part does not include any crypto example for CT but gives scope to include the CT example later

Example application trace in case when the application file is already created:

```
Entering Polling mode..
Type A Card - ISO14443-4A - UID : Len=7
      04 24 81 5A  47 21 80
Master Application of Desfire Card Selected,
Application A515A5 selected proceed for File Operations
APP Created By:
NxpNfcRdLib_v4.030.00.011627_2016
Value :   Len=4
      0D 00 00 00
Operation successful
Please remove the card.
```

*Note:*

*This example is available only with PSP package from NXP DocStore.*

## 7.18 PN7462AU\_ex\_phExRfPCDA

This example demonstrates simple low level API usage to perform detection, anti-collision, activation, authentication and R/W operation on the Type A cards according to ISO14443 and MIFARE standard.

Application is using low level Rf interface HAL implementation in Flash. There is limitation to only one card at the time. Supported TypeA cards are Type1 TOPAZ, MIFARE Ultralight, MIFARE Classic, MIFARE DESFire cards will pass through activation and anti-collision and. In the case of MIFARE Classic card also authentication with default key is demonstrated. In the case of MIFARE DESFire card L4 activation is demonstrated.

Typical application trace in case of MIFARE DESFire card is as follows:

```
Polling Start
Found TypeA
DesFire R/W PASS : =106
DesFire R/W PASS : =212
DesFire R/W PASS : =424
```

## 8. References

---

- [1] LPCXpresso webpage [www.nxp.com/redirect/lpcware.com/lpcxpresso/download](http://www.nxp.com/redirect/lpcware.com/lpcxpresso/download)
- [2] PN7462AU Datasheet
- [3] AN11706 PN7462 Antenna design guide
- [4] NFC-COCKPIT: NFC Cockpit configuration tool for NFC ICs  
<http://www.nxp.com/pages/nfc-cockpit-configuration-tool-for-nfc-ics:NFC-COCKPIT>

## 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 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 B.V.

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

**DESFire** — is a trademark of NXP B.V.

**I<sup>2</sup>C-bus logo** — is a trademark of NXP B.V.

**ICODE and I-CODE** — are trademarks of NXP B.V.



## 10. List of figures

Fig 1.	PN7462AU Customer demo board .....	4	Fig 38.	LPCXpresso installing LPC plugin - adding installation repository .....	35
Fig 2.	PNEV7462B supply .....	5	Fig 39.	LPCXpresso installing LPC plugin - adding LPXpresso .zip .....	36
Fig 3.	PN7462AU Customer board schematic (PN7462 part) .....	6	Fig 40.	LPCXpresso installing LPC plugin – enabling PN7462AU feature .....	36
Fig 4.	PNEV7462B contact slot schematic.....	7	Fig 41.	Importing project to LPCXpresso IDE (1) .....	37
Fig 5.	PNEV7462B TDA8026 part schematics.....	7	Fig 42.	Importing project to LPCXpresso IDE (2) .....	38
Fig 6.	PNEV7462 Smith chart default antenna matching 65x65.....	8	Fig 43.	Select project .....	39
Fig 7.	Matching circuit principle.....	9	Fig 44.	Project Workspace will all examples .....	40
Fig 8.	Antenna and matching .....	9	Fig 45.	Building a project.....	41
Fig 9.	PNEV746B V2.1 .....	11	Fig 46.	Build output - flash binary file .....	41
Fig 10.	PNEV7462B V2.2 .....	12	Fig 47.	Successful build .....	42
Fig 11.	Board Power settings.....	13	Fig 48.	Connecting LPC Link2 to the PN7462A Customer board .....	42
Fig 12.	VBUS supply jumper setting .....	14	Fig 49.	Launch debug session .....	43
Fig 13.	Supply indicator .....	14	Fig 50.	Select the launch configuration .....	43
Fig 14.	Default supply connection of the PN7462AU using all blocks.....	15	Fig 51.	Debug project.....	44
Fig 15.	Host Interface selection – USB mode .....	16	Fig 52.	Inserting breakpoints .....	45
Fig 16.	Host Interface selection - I2C mode.....	17	Fig 53.	Debug view and debugger traces of PN7462AU .....	46
Fig 17.	Host Interface selection - SPI.....	17	Fig 54.	Peripheral view.....	46
Fig 18.	Host Interface selection - HSU.....	18	Fig 55.	Debug view and debugger traces of PN7462AU .....	47
Fig 19.	JTAG/SWD debug probe connector.....	18	Fig 56.	HW setup for the flashing via LPC-Link2.....	48
Fig 20.	PN7460 NFC Cockpit: Activation of a MIFARE DESFire EV1 card + Get Application ID.....	20	Fig 57.	Selecting debug configuration.....	48
Fig 21.	PN7462 NFC cockpit register access .....	21	Fig 58.	Debugging PN7462AU FW .....	49
Fig 22.	PN7462 direct EEPROM access.....	22	Fig 59.	Build output .....	50
Fig 23.	PN7462 analog and digital test signals .....	23	Fig 60.	Build output - flash binary file .....	51
Fig 24.	Static overview of PN7462AU firmware .....	24	Fig 61.	PN7462AU as USB mass storage device .....	52
Fig 25.	Architecture diagram.....	25	Fig 62.	PN7462AU IC detected as USB mass storage device for FW/ EEPROM upgrade .....	52
Fig 26.	Contactless architecture view .....	27	Fig 63.	LEDs status.....	54
Fig 27.	Contact architecture view.....	28	Fig 64.	HW setup for the main example .....	55
Fig 28.	Project initialization order.....	28	Fig 65.	phExMain standby flow .....	58
Fig 29.	FreeRTOS usage diagram.....	29	Fig 66.	phExMain non standby flow .....	59
Fig 30.	Overview of PN7462AU development environment .....	30	Fig 67.	PN7462AU phExMain sequence diagram for standby scenario with RTOS.....	64
Fig 31.	LPCXpresso installation.....	31	Fig 68.	PN7462AU phExMain sequence diagram for non-standby scenario with RTOS.....	65
Fig 32.	Windows security dialog .....	32	Fig 69.	HW Setup for the “phExEMVCo”example .....	66
Fig 33.	LPCXpresso IDE.....	32	Fig 70.	HW setup for the “phExRF”example .....	68
Fig 34.	Product activation (1).....	33	Fig 71.	phExRF Example Application Flow .....	70
Fig 35.	Product activation (2).....	33	Fig 72.	HW setup for the “phExCT”example .....	71
Fig 36.	Product activation (3).....	34			
Fig 37.	LPCXpresso – install new SW .....	35			

Fig 73. phExCT Example Application Flow .....73  
Fig 74. HIF demo architecture .....76  
Fig 75. POS use case demo architecture .....79  
Fig 76. POS demo architecture.....80  
Fig 77. PC USB reader example block diagram .....81  
Fig 78. Example architecture .....81  
Fig 79. Door access example .....84  
Fig 80. Door Access Example - Export Controlled  
Version.....85

## 11. List of tables

---

Table 1.	Assembled matching components .....	10
Table 2.	Supply options .....	15
Table 3.	Development environment .....	30
Table 4.	Files found in USB mass storage.....	52
Table 5.	Code and data protection level .....	53
Table 6.	Status of read write operating code .....	53
Table 7.	“phExMain” Example features .....	56
Table 8.	“phExEMVCo” Example features .....	66
Table 9.	“phExRf” Example features .....	68
Table 10.	“phExRf” Example features .....	72
Table 11.	“phExRf” Example features .....	74
Table 12.	“phExCT7816” Example features.....	75
Table 13.	“phExHif” Example features .....	77
Table 14.	Host interface selection.....	77
Table 15.	Operation on the packet received on HIF .....	77
Table 16.	PN7462AU_ex_phSystemServices features...	82

## 12. Contents

<b>1.</b>	<b>Introduction</b> .....	<b>3</b>	5.1	Hardware abstraction layer – HAL.....	25
1.1	PNEV7462B concept .....	3	5.2	Protocol abstraction layer – PAL .....	25
<b>2.</b>	<b>Hardware overview of the PN7462AU Customer Demo Board</b> .....	<b>4</b>	5.3	Application layer – AL.....	26
2.1	PN7462AU Customer Demo Board.....	4	5.4	OSAL and utilities layer .....	26
2.1.1	Power supply.....	4	5.5	Component view.....	27
2.1.2	PN7462AU block.....	5	5.5.1	Contactless component view.....	27
2.1.3	LPCXpresso block.....	6	5.5.2	Contact component view .....	27
2.1.4	Smartcard reader and SAM slots extensions .....	7	5.6	Building a project from bottom to top.....	28
2.1.5	Antenna matching and EMC filter.....	8	5.7	RTOS and it's usage .....	29
2.2	Customer demo board available versions .....	10	<b>6.</b>	<b>Managing the PN7462AU SW projects with LPCXpresso IDE</b> .....	<b>30</b>
2.2.1	PNEV7462B V2.1.....	10	6.1	Development environment.....	30
2.2.2	PNEV7462B V2.2.....	12	6.2	Installation of the LPCXpresso IDE .....	31
2.2.2.1	Design changes V2.1 to V2.2:.....	12	6.3	Adding PN7462AU Plugin .....	34
<b>3.</b>	<b>Configuration of the PN7462AU Customer board</b> .....	<b>13</b>	6.4	Importing provided SW example projects.....	37
3.1	Board power settings .....	13	6.5	Building projects .....	40
3.1.1	PN7462AU supply options .....	13	6.6	Running and debugging the example projects .....	42
3.1.2	Power supply status LED .....	14	6.6.1	Break points .....	44
3.1.3	Supply options for PVDD, VUP_TX and TVDD .....	14	6.6.2	Debug traces .....	45
3.2	Host interface configuration.....	16	6.6.3	Peripheral view.....	46
3.2.1.1	USB Host Interface configuration .....	16	6.7	Updating Customer Board Firmware (Flash memory) .....	48
3.2.1.2	I2C Host Interface configuration.....	16	6.8	Updating Flash via SWD interface.....	48
3.2.1.3	SPI Host Interface configuration.....	17	6.9	Updating Flash via USB MSD interface.....	51
3.2.1.4	HSUART Interface configuration .....	17	<b>7.</b>	<b>Associated SW projects</b> .....	<b>54</b>
3.2.2	Debug interface.....	18	7.1	General overview .....	54
<b>4.</b>	<b>NFC Cockpit getting started</b> .....	<b>18</b>	7.1.1	Application messages – debug printouts.....	54
4.1	Board preparation .....	19	7.1.2	LEDs status specifications.....	54
4.2	Firmware and driver .....	19	7.2	PN7462AU_ex_phExMain – main example (CLIF + CTIF functionality) .....	55
4.3	NFC Cockpit installation.....	19	7.2.1	Demo setup .....	55
4.4	NFC Cockpit getting started .....	19	7.2.2	Features .....	55
4.5	PN7462 Register access.....	21	7.2.3	FreeRTOS usage by the example .....	56
4.6	PN7462 EEPROM access.....	22	7.2.4	Operation with standby and without standby.....	56
4.7	PN7462 analog and digital test signals .....	23	7.2.5	MF Classic.....	59
<b>5.</b>	<b>Software application stack</b> .....	<b>24</b>	7.2.6	MF Ultralight .....	60

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

7.2.7	MF DESFire .....	60	7.7.4	Apu transactions.....	74
7.2.8	Jewel reader.....	60	7.8	PN7462AU_ex_phExCT7816 example (CT functionality) .....	75
7.2.9	ISO15693 – ICODE SLIX.....	60	7.8.1	Demo setup .....	75
7.2.10	ISO18000-3.3 – ICODE ILT .....	61	7.8.2	Features .....	75
7.2.11	Type B eZLINK/ SLE card.....	61	7.8.3	ISO7816 activation .....	75
7.2.12	Type F (FeliCa tag) .....	61	7.8.4	APDU transactions .....	75
7.2.13	ISO14443-4 card mode (till activation).....	61	7.9	PN7462AU_ex_phExHif example.....	76
7.2.14	Passive and active ISO18092 initiator (till activation).....	62	7.9.1	Demo setup .....	76
7.2.15	Passive ISO18092 target (till activation) .....	62	7.9.2	Features .....	77
7.2.16	Contact Example.....	62	7.9.3	HIF selection .....	77
7.2.17	RTOS task management.....	62	7.9.4	Operation selection .....	77
7.2.18	No-RTOS management .....	65	7.9.5	EEPROM configuration dependencies .....	78
7.3	PN7462AU_ex_phExEMVCo example (CLIF + CTIF functionality).....	66	7.9.6	LPCExpresso projects provided for LPC1769 ..	79
7.3.1	Demo setup.....	66	7.10	PN7462AU_ex_phExPos example.....	79
7.3.2	Features.....	66	7.11	PN7462AU_ex_phExCcid example.....	80
7.3.3	EMVCo polling loop.....	67	7.12	PN7462AU_ex_phSystemServices example.....	81
7.3.4	EMV transaction.....	67	7.13	PN7462AU_ex_phExVCom example .....	83
7.3.5	RTOS task management.....	67	7.13.1	Demo setup .....	83
7.3.6	No RTOS management.....	67	7.13.2	Command set.....	83
7.4	PN7462AU_ex_phExRf example (CL functionality).....	68	7.14	PN7462AU_ex_phExDoorAccess example.....	83
7.4.1	Demo setup.....	68	7.15	PN7462AU_ex_phExDoorAccessEC example.....	84
7.4.2	Features.....	68	7.16	PN7462AU_ex_phExNFCCcid example.....	85
7.4.3	Application Flow.....	69	7.17	PN7462AU_ex_phExMfCrypto example.....	85
7.5	PN7462AU_ex_phExRFPoll example (CL functionality).....	70	7.18	PN7462AU_ex_phExRfPCDA .....	86
7.6	PN7462AU_ex_phExCT example (CT functionality).....	71	<b>8. References .....</b>	<b>87</b>	
7.6.1	Demo setup.....	71	<b>9. Legal information .....</b>	<b>88</b>	
7.6.2	Features.....	72	9.1	Definitions.....	88
7.6.3	Application Flow.....	72	9.2	Disclaimers.....	88
7.6.4	EMVCo activation.....	73	9.3	Licenses .....	88
7.6.5	SELECT master card .....	73	9.4	Trademarks .....	88
7.7	PN7462AU_ex_phExCTEMVCo example (CT functionality).....	74	<b>10. List of figures .....</b>	<b>89</b>	
7.7.1	Demo setup.....	74	<b>11. List of tables .....</b>	<b>91</b>	
7.7.2	Features.....	74	<b>12. Contents .....</b>	<b>92</b>	
7.7.3	EMVCo activation.....	74			

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