# PN5180 SW Quick start guide

Rev. 1.6 — 7 May 2018 345816

User manual COMPANY PUBLIC

#### **Document information**

Info	Content
Keywords	PN5180, PN5180 SW design, PNEV5180B, NFC NXP Cockpit
Abstract	This user manual is related to the installation procedures of the PN5180 Evaluation board, which are related to the installation of the SW sample projects as well as the re-installation of the original LPC firmware to run the NFC Cockpit. It describes the steps to be done to become acquainted with the demo reader especially for SW development.



# **Revision history**

Rev	Date	Description
1.6	20180507	Editorial updates
1.5	20170511	MCUXpresso IDE description added
1.4	20170117	Updated description how to flash FW for the NFC Cockpit tool.
1.3	20170105	Updated examples descriptions, reworked firmware update
1.2	20161124	Updated examples descriptions
1.1	20160803	Note in section 5 regarding the LPCXpresso version added HCE, NFC Forum and MIFARE DESFire added to the Associated projects Cockpit version changed from 2.2 to 2.3 Required LPCXpresso version changed from 7.9 to 8.1.4 RTOS options added
1.0	20151126	Initial version

# **Contact information**

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

UM10954

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

# 1. Introduction

This document is the continuation of the "AN11744 - PN5180 Quick start guide" and describes the installation procedures of the SW development environment and handling SW example projects using the NFC Reader Library prepared for the PN5180 evaluation board.

It also describes how to re-install the original LPC firmware binary to use the NFC Cockpit again.

In this document the term "MIFARE Classic card" refers to a MIFARE Classic IC-based contactless card, the term "MIFARE DESFire card" refers to a MIFARE DESFire IC-based contactless card, the term "MIFARE Ultralight card" refers to a MIFARE Ultralight IC-based contactless card.

Projects used and explained in this documentation are:

#### Table 1. Example projects

Example projects delivered with the NFC Reader Library

Example	Description
NfcrdlilbEx1_Basic DiscoveryLoop	Explains how to poll for different technologies (Tag, P2P, HCE), detect and report them. Default configuration parameters are used.
NfcrdlilbEx2_AdvancedDi scoveryLoop	Explains how to poll for different technologies (Tag, P2P, HCE), detect and report them. All configuration parameters are used and explained.
NfcrdlilbEx3_ NFCForum	Explains how to configure the NFC Reader Library for different P2P modes such as Active Mode, Target Mode, Initiator Mode and SNEP Client/Server.
NfcrdlilbEx4_MIFARE Classic	Explains the usage of MIFARE Classic card commands.
NfcrdlilbEx5_ ISO15693	Explains the usage of this technology and provides an overview about the most common commands.
NfcrdlilbEx7_ EMVCo_Polling	Explains polling for EMVCo payment cards.
NfcrdlilbEx8_ HCE_T4T	Explains how to emulate a NFC Forum Type 4 Tag supporting read and write operations.
NfcrdlilbEx9_ NTagl2C	Explains NTAG-I2C specific commands.
NfcrdlilbEx10_ MIFAREDESFire	Explains the usage of MIFARE DESFire cards. (This example is delivered with the NFC Reader Library version available via NXP DocStore)
NfcrdlibEx11_ISO10373_ PCD	Example is used to perform ISO 10373-6 PCD compliance validation.
Nfcrdlib_SimplifiedAPI EMVCo	EMVCo loopback application with simplified API, which can be used for EMVCo level 1 digital certification.
Nfcrdlib_SimplifiedAPI EMVCo_Analog	Application is used to perform EMVCo2.6(L1) Analog compliance validation.
Nfcrdlib_SimplifiedAPI ISO	Explains how to use simplified API with different types of cards.

UM10954

# 2. Managing the PN5180 SW projects with MCUXpresso IDE

The PN5180 SW projects are delivered in a zip package and can be extracted, edited, compiled and linked with MCUXpresso IDE.

The MCUXpresso IDE is a low-cost highly integrated software development environment for NXP's LPC microcontrollers and includes all the tools necessary to develop highquality software solutions in a timely and cost-effective fashion. MCUXpresso 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 MCUXpresso IDE can build an executable of any size with full code optimization.

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

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

#### 2.1 Development environment

To use PN5180 prepared software package all components listed in the Table 2 are required.

Table 2. Development	Environment	
Item	Version	Description
PN5180EV5180B	1.0 or higher	PN5180 Customer Evaluation board (hardware)
LPC-Link 2	1.0	Standalone debug adaptor (hardware)
MCUXpresso IDE	10.0.0 or higher	Development IDE (PC software)

#### 2.2 Installation procedure of the MCUXpresso IDE

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

Multiple versions can be installed simultaneously without any issues.

The installation starts after double-clicking the installer file.

Setup - MCUXpresso IDE Optional debug driver selection These drivers are required when using	the debug probes listed below.
Windows may issue warnings when in These include drivers from Jungo Con NXP. If prompted, please allow these driver	nectivity, PE Micro, and SEGGER as well as
Image: With the second seco	
v10.0.0_344	< Back Next > Cancel

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

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

	Windows Security     Would you like to install this device software?     Name Philips (NXP) Universal Serial Bus contr     Publishen NXP Semiconductors USA. Inc.	
	Always trust software from "NXP Semiconductors     USA. Inc.".      You should only install driver software from publishers you trust. How can I decide which dee     software is soft to install?	
Fig 2. Windows sec	urity dialog	

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

#### PN5180 SW Quick start guide

UM10954



#### 2.3 Importing provided SW example projects

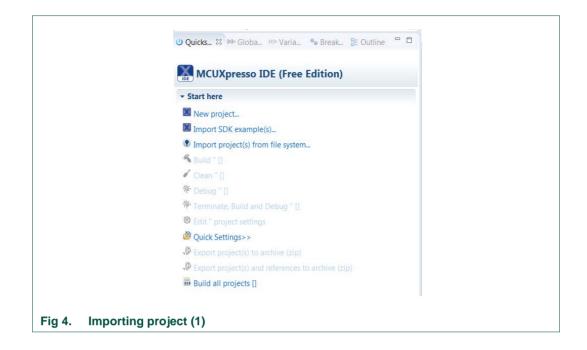
The use of quick start panel provides rapid access to the most commonly used features of the MCUXpresso IDE. Quickstart panel allows easy import projects, create new projects, build and debug projects.

The sequence of installing the software projects is indicated:

- Start the MCUXpresso IDE.
- Open new or dedicated workspace
- Select the option "Import project(s)" (see picture below).
- Browse the zip archive.
- MCUXpresso IDE unzips the software package.
- The software package is ready for use.

#### PN5180 SW Quick start guide

UM10954



In the Quickstart panel on the left-hand side, choose "Import projects(s)".

Import project(s) fr Select the examples a	-			
project archive or root wish to import, and pre		>. On the next page, s		
-	Open and 'legacy' examp	les are provided.		
Project archive (zip)				Browse
Project directory (unp	acked)			
Root directory				Browse
LPCOpen				
LPCOpen is the recom	mended code base for Co	ortex-M based NXP LP	C Microcontrollers.	
button in the Project a	des the LPCOpen package rchive (zip) section, above button below to Browse	e, and navigating to th	ne Examples/LPCOp	en directory.
Browse LPCOpen reso		the hxp.com website	for fatest resources.	
?	< Back	Next >	Finish	Cancel

Browse the desired package and click "Next".

M Import project(s) from file system	
Import project(s) from file system Select a directory to search for existing Eclipse projects.	
Projects:	
ApiDocumentation (ApiDocumentation/)     ApiDocumentation (ApiDocumentation/)     DAL (DAL/)     FreeRTOS (FreeRTOS)     Ipc_board_nxp_lpcxpresso_1769 (lpc_board_nxp_lpcxpresso_1769/)     Ipc_chip_175x_6x (lpc_chip_175x_6x/)     Vicrdlib_SimplifiedAPLEMVCo_(Ntcrdlib_SimplifiedAPLEMVCo/)     Vitcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (Ntcrdlib_SimplifiedAPLEMVCo_Analog (NtcrdlibEx1_BasicDiscoveryLoop/)     VitcrdlibEx1_BasicDiscoveryLoop (NtcrdlibEx1_BasicDiscove	Select All Deselect All Refresh
Copy projects into workspace Working sets Add project to working sets Working sets	New Select
(2) < Back Next > Finish	Cancel
Fig 6. Importing project (3)	

For a working demo project, you need to import at least four sub projects. One example project, the NFC Reader Library, FreeRTOS, one chip library and one board library.

When the import process has finished one can start browsing the code.

#### 2.4 Building projects

Building projects in a workspace is a simple case of using the Quickstart 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 7.

ſ

Γ

**UM10954** 

🛛 🗁 ApiDocumen	tation				
DAL					
FreeRTOS					
⊳ 🐸 lpc_board_nx	p_lpcxpresso_1769				
▷ 😂 lpc_chip_175	⊳ 😂 lpc_chip_175x_6x				
🛛 🖉 🖉 Nfcrdlib_Sim	plifiedAPI_EMVCo				
» Stordlib_S	New	•			
> Mfcrdlib_S	Go Into				
NfcrdlibEx	Open in New Window				
NfcrdlibE	Open in New Window				
🕨 🐸 NfcrdlibEx 🖺		Ctrl+C			
🕨 😂 NfcrdlibE		Ctrl+V			
🖻 🖉 NfcrdlibEs 🎽		Delete			
NfcrdlibE	Source	•			
NfcrdlibE	Move				
NfcrdlibEx	Rename	F2			
NfcrdlibEx	Import				
NifeedlibTr					
U Quickstart Pa	Build Project				
	Clean Project				
ΜΟυΧμ	Refresh	F5			
	Class Designt	.5			

As a part of the build output, the binary for the "User Flash" file is created. This binary file can be later also used to update LPC1769 Flash memory via USB mass storage interface.

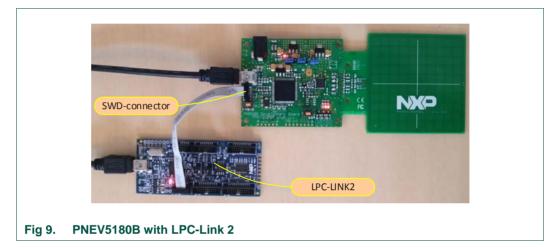
🕒 Project Explorer 🛛 🚼 Peripherals+ 🕮 Registers 🖾 Symbol
⊳ 🗁 FreeRTOS
Ipc_board_nxp_lpcxpresso_1769
▷ <sup>25</sup> Ipc_chip_175x_6x
» SimplifiedAPI_EMVCo
Mfcrdlib_SimplifiedAPI_EMVCo_Analog
» Stordlib_SimplifiedAPI_ISO
MfcrdlibEx1_BasicDiscoveryLoop
🔺 🞇 Binaries
> Distribution Structure Structur
Includes
DAL
🖻 🗁 DebugLPC1769
Fig 8. List of projects in "Project Explorer view

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".

#### 2.5 Running and debugging a project

This description shows how to run the "*NfcrdlibEx1\_basicDiscoveryLoop*" example application for the PN5180 evaluation development board. The same basic principles will apply for all other examples. In cases where example will need additional configuration this will be detailed described in the example description.

PN5180 evaluation board should be connected to the computer via LPC-Link 2, as shown in Fig 9.



When debug is started, the program is automatically downloaded to the target and it's programmed to the LPC1769 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.

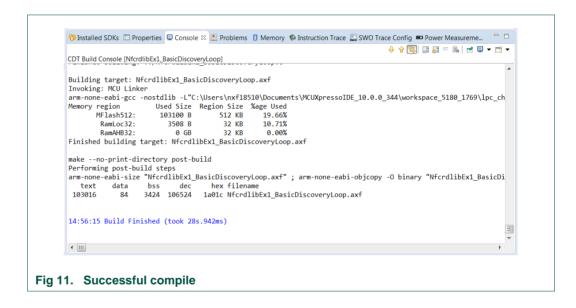
To start debugging your application on the PN5180, simply highlight the project in the Project Explorer and then in the Quickstart Panel click Debug, as shown in Fig 10. The MCUXpresso IDE will first build application, flash application binary and then will start with debugging.

Before running the project, please ensure that the correct microcontroller and the correct build configurations are chosen. Information about how to do this can be found in the Fig 23 and chapter 6.3.

#### PN5180 SW Quick start guide

UM10954

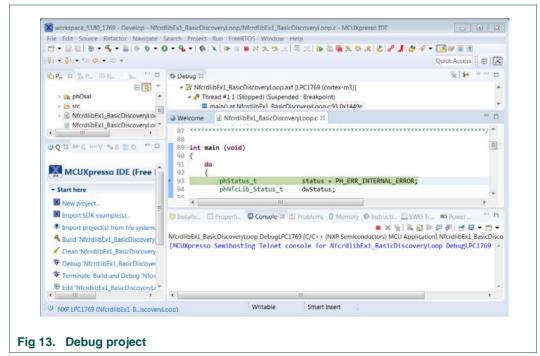




Select "LPC-Link 2" as a debug probe.

	nnect to target: LPC1769 probe found. Select the probe to us	e:			
Av	ailable attached probes				
	Name	Serial number/ID	Туре	Manufac	IDE Debug Mode
X	LPC-LINK2 CMSIS-DAP V5.182	D2G2ITKW	LinkServer	NXP Semic	Non-Stop
V V V	pported Probes (tick/untick to enab MCUXpresso IDE LinkServer (inc. C P&E Micro probes SEGGER J-Link probes obe search options				
Se	earch again				
✓ F	Remember my selection (for this Lau	inch configuration)			
?				ОК	Cancel

After successful software upload, the execution of the project starts immediately, but might halt at the initial breakpoint. To resume execution, click the resume button.



In the console window application debug outputs of the execution can be seen.



After the execution has reached the end of the main function please click the Terminate button to stop the execution. Otherwise rerun of the project will be possible.

		Run the program.	
	Ş	Step over C/C++ line.	
	ų	Step into a function.	
		Stop the debugger.	
	00	Pause execution of the running program.	
	i⇒	Instruction stepping mode (disassembly).	
Fig 15. Debug Button	s		

Buttons in the debug toolbar provide next functionalities:

# 3. Managing the PN5180 SW projects with Linux and Kinetis platform

Detailed description and guideline, how to import and manage NFC NXP Reader Library projects for Linux and Kinetis platform, check:

- AN11802 NFC Reader Library for Linux Installation Guidelines
- AN11908 NFC Reader Library for FRDM-K82F Installation Guidelines

UM10954

# 4. Associated projects

All example projects are available for download at the PN5180 product page in the documents section and are being distributed in one single file.

All projects are packaged into a single installer file. After downloading the zip file, extract it and run the installer. The installer makes a copy of all documents and SW on the hard disk.

By default, the projects are preconfigured to be run on the PNEV5180B development board. This is defined by preprocessor directive PHDRIVER\_LPC1769PN5180\_BOARD (properties-> settings->preprocessor) and by macro in "../intfs/ph\_NxpBuild\_App.h".

```
//#define NXPBUILD__PHHAL_HW_RC663
#define NXPBUILD PHHAL HW PN5180
```

#### Running the projects with, or without FreeRTOS

All projects described in the following sub chapters can be configured to run with or without FreeRTOS operating system. To enable/disable FreeRTOS support, define preprocessor directive (properties-> settings->preprocessor) PH\_OSAL\_FREERTOS or PH\_OSAL\_NULLOS.

#### 4.1 Example 1 – Basic Discovery Loop

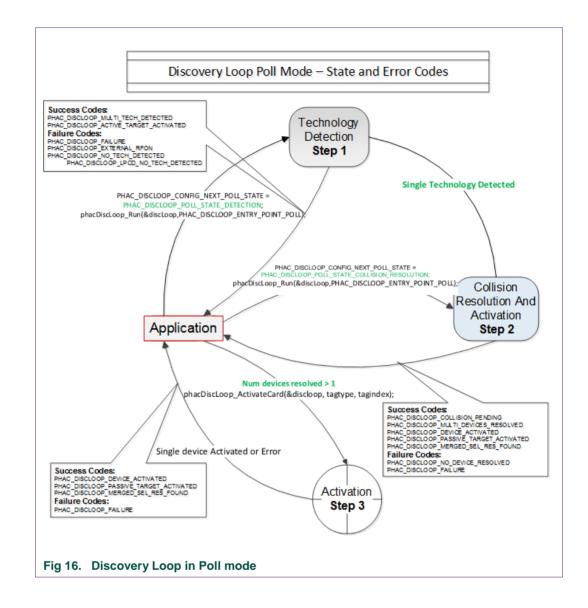
The Discovery Loop can be seen as the entry point when starting to communicate with an NFC tag or device. It scans the close environment for tags and devices of different technologies.

Example is implemented to work in POLL and LISTEN mode of the discovery loop. Information (like UID, SAK, and product type of MIFARE IC) of the detected tags are printed out and it also prints information when it gets activated as a target by an external initiator/reader. Whenever multiple technologies are detected, example select first detected technology and resolve it.

In passive poll mode, Low Power Card Detection (LPCD) is enabled.

#### PN5180 SW Quick start guide

UM10954



The core function of this example is "*BasicDiscoveryLoop\_Demo()*", where initialization of the NFC Reader library and polling for NFC technologies is implemented. After each polling loop, application is checking polling result and printout information about the detected tags or devices.

This example is using default DiscoveryLoop configuration, which enables all supported technologies and it is limited to one device for each technology.

#### Table 3. Supported technologies

ISO14443P3A	ISO15693- SLI	FeliCa	TYPEF_TARGET_PASSIVE
ISO14443P4A	ISO18000P3M3	TYPEA_TARGET_PASSIVE	TYPEF_TARGET_ACTIVE
ISO18092MPI	ISO14443P3B	TYPEA_TARGET_ACTIVE	

#### 4.2 Example 2 – Advanced Discovery Loop

Additionally to Example 1 the Advanced Discovery Loop example explains the different configuration options of the Discovery Loop and configure DiscoveryLoop with default values based on the interested profile, NFC or EMVCo.

The configuration of the "DiscoveryLoop" is implemented in "LoadProfile()" function.

#### 4.3 Example 3 – NFC Forum

Explains how to configure the NFC Reader Library for different P2P modes such as Active Mode, Target Mode, Initiator Mode and SNEP Client/Server.

In SNEP Server mode the example waits for a connection from a SNEP Client. When the connection between client and server is establish, client send a data and server read it. The application displays read data in the console window of the LPCXpresso IDE.

In SNEP Client mode, the application tries to connect to a SNEP Server. Once the connection is established, it transmits an NDEF message to the server.

#### 4.4 Example 4 – MIFARE Classic card communication

This example demonstrates how to configure "DiscoveryLoop" to poll for only one technology and how to resolve detected card, in this example MIFARE Classic is used.

Once MIFARE Classic card is activated, application printout information like UID, ATQA and SAK and perform the authentication with MIFARE Classic card default key. After successful authentication, basic read/write operations are implemented.

This example is good start in case of working with only one card or to see how to manage MIFARE Classic cards.

#### 4.5 Example 5 - ISO15693

Similar to the previous example, this one is also using only one technology, in that case ISO15693. "*DiscoveryLoop*" is configured to resolve only one device and in the example it is shown how to change Tx Guard Time for T5T cards, this is implemented in "phApp\_Init()" function.

Once ICODE SLI is resolved and activated, application printout card information like type of the card and UID, and it will read and write from/to the memory block.

This example is good start in case of working with only one card or to see how to manage ISO15693 type of the cards.

For a much more extensive example, demonstrating the use of ISO/IEC 15693 and ISE/IEC 18000-3 Mode 3 tags (ICODE SLI and ICODE ILT). In order to assure ICODE SLI and ILT detection please check HAL digital delay define settings as described in chapter 4.

#### 4.6 Example 7 – EMVCo Polling

The EMVCo Polling example it is demonstrated how to configure NFC Reader Library as specified by EMVCo specifications and starts polling for EMVCo cards.

Once an EMVCo compatible card is resolved and activated, it demonstrates the exchange of APDU commands. This example shall help the developers getting started more quickly when working with EMVCo cards.

#### 4.7 Example 8 – HCE T4T

Example 8 implements a Type 4 Tag card emulation according to NFC Forum Type 4 Tag specification. The example supports all specified commands such as *Select*, *ReadBinary*, *UpdateBinary*.

With this example our reader is in card emulation mode (HCE) and it support reading and writing data. Default data is configured as an NDEF message as a url <u>www.nxp.com</u>.

The maximum NDEF length the reader can write is limited by NDEF file size used in example (default configured as 1024 bytes).

#### 4.8 Example 9 – NTAG-I2C

The NTAG-I2C example demonstrates the use of special features which are supported by NTAG-I2C. By using POLL mode of the discovery loop, example detect the NTag I2C cards and displays detected tag information like UID, ATQA, SAK, Version info and perform "*Page Read*" and "*PageWrite*" commands.

For more details about the NTAG-I2C and its functionalities please consult the related product page [2]

#### 4.9 Example 10 – MIFARE DESFire card communication

The MIFARE DESFire example demonstrates how to use MIFARE DESFire EV1 cards.

Once MIFARE DESFire card is resolved and activated, it displays MIFARE DESFire applications created by this example previously and it displays 32bit signed integer which is incremented after each successful detection of tag.

In case no application is present on the tag, new application will be created with two new files to hold NXPNFCRDLIB version used to create this application and another file to hold 32bit signed integer.

**Note:** This example including the required modules of the NFC Reader Library is only available via NXP Docstore.

#### 4.10 Example 11 – ISO10373 PCD

This example is used to perform ISO 10373-6 PCD compliance validation. This example has to be executed in the DUT which has an ISO 14443 based PCD implementation. The ISO 10373-6 test methods verifies the compliance to the ISO 14443 protocols. An

external tool like Micropross MP300 implements the test methods for the ISO 10373-6 and is used as the counterpart for this testing.

#### 4.11 Simplified API EMVCo

This application will configure Reader Library as per EMVCo specification and start EMVCo polling. This loop back application will send SELECT\_PPSE command and is used to test EMVCo.3.1a(L1) digital compliance. Simplified approach, after library initialization, is using only three commands:

- phNfcLib\_Activate()
- phNfcLib\_Transmit()
- phNfcLib\_Receive()

#### 4.12 Simplified API EMVCo Analog

This example provides command line interface so user can choose one of three operation modes as below.

- EMVCo LoopBack Application
- Trans send Type A application
- Trans send Type B application

Above Application modes are used to perform EMVCo2.6(L1) Analog compliance validation.

#### 4.13 Simplified API ISO

This example is a reference application to demonstrate the usage of Simplified API with ISO profile. Application contains example of Type A Layer 4, Type B Layer 4, ISO/IEC15693 and ISO/IEC180003m3 and MIFARE DESFire card, MIFARE Ultralight card and MIFARE Classic card communication.

Example demonstrates how to use simplified API, which require, after successful library initialization, only three commands:

- phNfcLib\_Activate()
- phNfcLib\_Transmit()
- phNfcLib\_Receive()

# 5. Flashing Firmware on the LPC1769

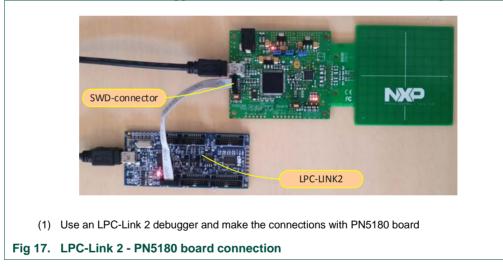
Flashing VCOM firmware is the procedure needed to prepare PN5180 evaluation board to be used with the Cockpit tool. The Cockpit tool is software design tool for prototyping NXP NFC card reader applications.

VCOM firmware which needs to be flashed on the LPC1769 is provided with the installer package of the NXP NFC Cockpit application and it can be found in "\NxpNfcCockpit\_v3.10.0.0\firmware\Secondary\_PN5180" folder.

This folder contains several binaries which can be used as VCOM firmware application:

- BootLoader\_And\_Nfcrdlib\_SimplifiedAPI\_EMVCo\_Secondary.bin
- BootLoader\_And\_phRfOnOff\_Secondary.bin
- BootLoader\_And\_phUcBal\_Secondary.bin

Steps required to flash VCOM application: Connect the LPC-LINK2 debugger to the PN5180 board as shown in the Fig 17.



🔀 workspace\_5180\_1769 - Develop - Nfcrdlib\_SimplifiedAPI\_EMVCo\_Analog/Readme.txt - MCUXpresso IDE File Edit Navigate Search Project Run FreeRTOS Window Help ◊ 🤩 👢 💊 (୬. ୭. 厘 🖬 🐠 (文 트 ) 🗴 ୭. ୮ 🖉 = 🛛 🛸 (४ = ۵. – ۷ = ۷ = ۵. – ۷ = ۴. 🐐 📾 – 🖉 – ೮ = ۲. 🔓 Project Ex... 🕺 🚼 Peripheral... 🕮 Registers 🖾 Symbol Vi... 🧮 🗖 💿 Welcome 🛛 🗟 NfcrdlibEx1\_BasicDiscoveryLoop.c E 🔄  $\nabla$ 37 ApiDocumentation X LinkServer GUI Flash programmer X 🖻 🗁 DAL FreeRTOS Program Flash using LPC-LINK2 CMSIS-DAP V5.182 Ipc\_board\_nxp\_lpcxpress Program target flash: LPC17xx (NXP LPC1769) Ipc chip 175x 6x Nfcrdlib\_SimplifiedAPI\_E Options » Stordlib\_SimplifiedAPI\_E Display progress log » SimplifiedAPI IS Reopen on completion MfcrdlibEx1\_BasicDiscove Repeat on completion Run flash command and copy to clipboard Image: NfcrdlibEx11 ISO10373 Just copy flash command to clipboard Confirm command before executing » StordlibEx2 AdvancedDi MicrollibEx3 NFCForum **Connection Options** MicrollibEx4\_MIFAREClass Use JTAG interface MfcrdlibEx5\_ISO15693 Additional options » StordlibEx7\_EMVCo\_Polli NfcrdlibEx8\_HCE\_T4T LinkServer connect script Browse... NfcrdlibEx9\_NTagI2C MfcrdlibTst12\_Rc663Lpcc Flash Driver Description NxpNfcRdLib Flash driver LPC175x\_6x\_512.cfx b Description of the phose o Browse... Program flash memory Erase flash memory Select file C:\nxp\NxpNfcCockpit\_v3.10.0.0\firmware\ 
Browse... Base address 0x00 Reset target on completion **Erase Options** Mass erase Erase only required sectors OK Cancel (U) Quicks... ⋈ 🗵 🖾 Global... (2) Create (or Open an existing MCUXpresso project) and set the MCU type to LPC1769 (3) Chose "Program flash" from menu bar (4) Select bootloader binary, e.g. NxpNfcCockpit\_v3.10.0.0\firmware\Secondary\_PN5180\BootLoader\_And\_Nfcrdlib\_SimplifiedAPI \_EMVCo\_Secondary.bin<sup>3</sup> (5) Set base address to 0x00 (6) Press "OK" button and flash the bootloader Fig 18. Flashing Bootloader.bin

Use the MCUXpresso Program Flash utility flash to flash bootloader binary to the MCU.

Flash the binary on the PNEV5180B and ensure that the process succeeds.

MCUXpresso IDE  Program Flash  crt_emu_cm_redlinkflash-load-exec "C: xp\NxpNfcCockpit_v3.10.00\firmware\Secondary_PN5180\BootLoader_And_Nfcrdlib_SimplifiedAPI_EMVCc gdebug 2vendor NXP -o LPC1769load-base=0x00 -ProbeHandle=1 -CoreIndex=0reset vectresetflash-driver LPC175x 6x 512.cfx	Secondary.bin" -
Nt: Erased/Wrote page 0-16 with 65992 bytes in 675msec Nc: Closing flash driver LPC175x, <u>6x</u> _512.cfx Pb: (100) Finished writing Flash successfully. Nt: Flash Mrite Done Nt: Loaded 0x101C8 bytes in 810ms (about 81kB/s) Nt: Reset target (core)	Ĩ
OK	
(7) Inspect program flash output	
Fig 19. Flash report	

Disconnect the USB cable, remove LPC-Link 2 connection and reset the board.

The board appears as a VCOM device:

Hardware and Sound Devices and Printers	<ul> <li>✓</li> <li>✓</li></ul>	×
File         Edit         View         Tools         Help           Add a device         Add a printer         Add a printer	₹. •	0
<ul> <li>Unspecified (2)</li> <li>Logitech USB Headset</li> <li>NxpNfcCockpit</li> </ul>		
20. PN5180 Board Correctly installed		

The PN5180 based evaluation board is ready now to be used with the NXP NFC Cockpit tool.

# 6. Supplementary Notes

#### 6.1 General Software Architecture

The software of the reference reader is based on the NFC Reader Library Fig 21. It intends to be simple, modular, easily readable and quickly portable by all the customers. This philosophy is reflected in its architecture which is divided into 4 layers:

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

ryLoop demo ation Application	MIFARE Classic application ISO/IEC 1569 application cation Layer) - Commandse Jewel/ NFC Forum ISO	demo Emu. T4T application Application	NTAG I2C demo application NFC Activity HCE La	wer NFC P2P Package	EMIVCo Loop back app for Compliance
Part of Part of MIFARE FeliCa Cmd DESFire set		ts	NFC Activity HCE La	ver NEC P2P Package	
MIFARE FeliCa Cmd DESFire set	Jewel/ NEC Forum ISO			SNEP 1.0 Network	Simplified API
		MEC ISO/IEC ICODE 693 18000 3m3 SLI	Discovery Loop T4T-A	SiveP 1.0 Protocols	
Sw Sw	Sw Sw Sw	Sw Sw	Sw Sw	Su	
	PAL (Protocol Abst	traction Layer) - Activation an	d Exchange		
14443 14	4443 144	43 compliant 18092 Initiate			
Ser Sw	Sw Stub Sw	Sw Sw	Sw Sw Sw		
	HAL (Hard	ware Abstraction Layer) - Rea	ders		
PN7462AU					
	BAL (Bu	s Abstraction Laver) - Interface	19		
Deserboury Di					
LinuxSPI					
	Cor	mmon (Layer independent)			
		oSym CryptoRng	phPlatform		
		Sw PN7462AU L	C1789 K82 Raspberry		
3 ic 3 srin	3         1           44.4         1           50         0           6         3           3         PN7462AU           MR         RaspborryPI           LinuxSPI         LinuxSPI           13.4         Tools	3         1443 4         144 4         144 4           9         5         5         5           10         5         5         5         5           13         PM7452AU         BAL (Burnes)         BAL (Burnes)         BAL (Burnes)           16         5         5         5         5         5           16         5         5         6         5         5           17         RaspborryPP LinuxSPH         5         5         6           134         Tools         Log         08AL Units         Crypt	2         1443 4         1443 4         complexit 4         1402 4         complexit 4         1002 4         10	a     i443 4 4 4     image: 1443 4 4     a    <	2         3         3         3         44.3 4         0

#### 6.1.1 Bus abstraction layer

This layer offers functions to abstract the hardware parts of the LPC1XXX microcontroller.

These functions use the specific libraries available for the LPC1XXX family microcontroller. Based on these stacks, the communication routines for the relevant physical media I2C/SPI can be easily designed. These drivers are specific for the LPC1XXX family and therefore cannot be ported to other microcontrollers.

#### 6.1.2 Hardware abstraction layer

This layer offers functions to abstract the hardware parts of the transceiver CLRC663.

#### 6.1.3 Protocol abstraction layer

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

UM10954

The following PAL modules are available in this software package:

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

#### 6.1.4 Application layer

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

#### 6.2 Build configuration

All the projects mentioned in Chapter 0 are available in debug configuration. Additionally, the Polling project comprises the release configuration.

• Debug configuration

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

• Release configuration

Once the project is debugged and mature, it might be interesting to use the release configuration, to use the hardware stand alone. No debug messages are displayed in the console window.

The build configuration can be selected as follows:

- Click on the project in the project window of the MCUXpresso IDE,
- Right click of the mouse → Select Build Configuration,
- Set active DebugLPC1769 build (or ReleaseLPC1769 build) for LPC1769.

	New Go Into	+	44 45 This example contains three mode of operations within it 46 choose as below.
	Open in New Window		47
	Copy Paste Delete Source Move Rename	Ctrl+C Ctrl+V Delete	<ul> <li>48 1.EMVCo LoopBack Application</li> <li>49</li> <li>50 2.Trans send Type A application</li> <li>51</li> <li>52 3.Trans send Type B application</li> <li>53</li> <li>54 Above Application modes are used to perform EMVCo2.6(L1)</li> </ul>
2 2	Import Export		55 validation. The CPU frequency configured for the Host c 56 should be sufficient enough to meet EMVCo timing require 57
	Build Project Clean Project Refresh Close Project Close Unrelated Projects	F5	58 593 Restrictions on Nfcrdlib_SimplifiedAPI_EMVCo_Analog 60 61 62 - Nfcrdlib_SimplifiedAPI_EMVCo_Analog is restricted to b 63 Boards/ICs mentioned in Section-14. 64 - The APIs are intended for NXP contact_less reader ICs.
	Build Configurations	•	Set Active I DebugLPC1769 (Debug build for LPC1769)
	Build Targets Index	+ +	Manage         2 ReleaseLPC1769 (Release build for LPC1769)           Project requires NxpNtcKdL1D V05.02.00 o         Build All           The default used MCU Type is LPC1769. W
	Run As Debug As Profile As	) }	Clean All Build Selected

#### 6.3 Setting the MCU

There are many LPC microcontrollers supported by the MCUXpresso IDE build in compiler. Before compiling a project, the correct MCU need to be set.

- Right click the project  $\rightarrow$  choose properties (at the bottom)
- C/C++ build → MCU settings → expand desired LPC700 MCU group → choose the correct microcontroller → click OK

ype filter text	MCU settings	← < ⇒ <
<ul> <li>Resource</li> <li>Builders</li> <li>4 C/C++ Build</li> </ul>	Available parts	
Build Variables Environment	SDK MCUs	
Logging	✓ Preinstalled MCUs	
MCU settings	MCUs from preinstalled LPC and generic Cortex-M part support	
Settings	NXP LPC1769	
Tool Chain Editor	LPC1765	
C/C++ General	LPC1766	
> Code Analysis	LPC1767	
Documentation	LPC1768	=
File Types	LPC1769	
Formatter	> LPC177x_8x	
Indexer	> LPC18xx	-
Language Mapping Paths and Symbols	Target architecture: cortex-m3	

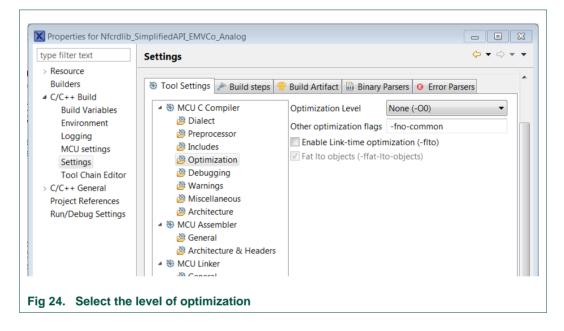
UM10954

#### 6.4 Level of compiler optimization

When the code size at the current compiler level overloads the FLASH size of the target board (512K for the ARM-based microcontroller LPC1769), a higher compiler optimization level can be selected to reduce the code size of the project.

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

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



#### 6.4.1 Optimization issues

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

#### 6.5 Removing the initial breakpoint on debug startup

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

1. Right click on the project and choose Launch Configurations → Edit current...

	Team Compare With Restore from Local History Launch Configurations Smart update Utilities Tools Run C/C++ Code Analysis	+ + + + +	 orm	Console [MfcrdlibEAG_EMVCo_] no-print-directory post- ing post-build steps e-abi-size "MfcrdlibEx6 data Edit current Create new Create and edit new Delete current Delete JTAG configuration	build	
L	Properties	Alt+Enter				



Create, manage, and run conf	igurations	Ň
	Name: NfcrdlibEx1_BasicDiscoveryLoop DebugLPC1769	
type filter text C++ (NXP Semiconductors) MCU A NfcrdlibEx1_BasicDiscoveryLoop E NfcrdlibEx1_BasicDiscoveryLoop R C++ Application C++ Application C++ Postmortem Debugger C++ Remote Application 18 Hardware Debugging 18 PEMicro Interface Debugging 18 Segger Interface Debugging unch Group	<ul> <li>Main</li> <li>Common</li> <li>Debugger</li> <li>Stop on startup at: main</li> <li>Force hardware breakpoint</li> <li>Debugger Options</li> <li>Target configuration</li> <li>Main</li> <li>GDB debugger: arm-none-eabi-gdb</li> <li>GDB command file:</li> <li>(Warning: Some commands in this file may interfere with the startup of example "run".)</li> <li>Non-stop mode (Note: Requires non-stop GDB)</li> <li>Enable Reverse Debugging at startup using:</li> <li>Software Reverse Debugging at startup using:</li> </ul>	
Filter matched 11 of 11 items	Revert Apply	•
?	Debug	

PN5180 SW Quick start guide

### 7. References

#### [1] MCUXpresso download website http://www.nxp.com/products/software-and-tools/run-time-software/mcuxpressosoftware-and-tools:MCUXPRESSO

[2] NTAG-I2C http://www.nxp.com/products/identification\_and\_security/nfc\_and\_reader\_ics/conne cted\_tag\_solutions/series/NT3H1101\_NT3H1201.html

- [3] AN11744 PN5180 evaluation board quick start guide, <u>www.nxp.com</u>
- [4] AN11742 PN5180 Dynamic Power Control, <u>www.nxp.com</u>

UM10954

345816

#### PN5180 SW Quick start guide

# 8. Legal information

#### 8.1 Definitions

**Draft** — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

#### 8.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

**Right to make changes** — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

**Applications** — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary

testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

**Export control** — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

**Evaluation products** — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US\$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

#### 8.3 Licenses

#### Purchase of NXP ICs with 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.

#### 8.4 Trademarks

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

MIFARE— is a trademark of NXP B.V.

MIFARE Classic - is a trademark of NXP B.V.

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

DESFire — is a trademark of NXP B.V

ICODE - is a trademark of NXP B.V

Kinetis - is a trademark of NXP B.V

# 9. List of figures

Fig 2.Windows security dialogFig 3.MCUXpresso IDEFig 4.Importing project (1)Fig 5.Importing project (2)Fig 6.Importing project (3)Fig 7.Building the projectFig 8.List of projects in "Project Explorer viewFig 9.PNEV5180B with LPC-Link 2Fig 10.Launch debug sessionFig 11.Successful compileFig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 23.Select the level of optimization	Fig 1.	Check box for NXP debug drivers	5
Fig 4.Importing project (1)Fig 5.Importing project (2)Fig 6.Importing project (3)Fig 7.Building the projectFig 8.List of projects in "Project Explorer viewFig 9.PNEV5180B with LPC-Link 2Fig 10.Launch debug sessionFig 11.Successful compileFig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll mode	Fig 2.	Windows security dialog	5
Fig 5.Importing project (2)Fig 6.Importing project (3)Fig 7.Building the project (3)Fig 8.List of projects in "Project Explorer viewFig 9.PNEV5180B with LPC-Link 2Fig 10.Launch debug sessionFig 11.Successful compileFig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 23.Select of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 3.	MCUXpresso IDE	6
Fig 6.Importing project (3)Fig 7.Building the project (3)	Fig 4.	Importing project (1)	7
Fig 7.Building the project	Fig 5.	Importing project (2)	7
Fig 8.List of projects in "Project Explorer viewFig 9.PNEV5180B with LPC-Link 2Fig 10.Launch debug sessionFig 11.Successful compileFig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 23.Select the build configurationFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 6.	Importing project (3)	8
Fig 9.PNEV5180B with LPC-Link 2Fig 10.Launch debug sessionFig 11.Successful compileFig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 23.Select the build configurationFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 7.	Building the project	9
Fig 10.Launch debug session	Fig 8.	List of projects in "Project Explorer view	9
Fig 11.Successful compileFig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 19.Flash reportFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 23.Select the build configurationFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 9.	PNEV5180B with LPC-Link 2	10
Fig 12.Select the launch configurationFig 13.Debug projectFig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 19.Flash reportFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 23.Select the build configurationFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 10.	Launch debug session	11
Fig 13.Debug project.Fig 14.Application printouts in Console window.Fig 15.Debug ButtonsFig 16.Discovery Loop in Poll mode.Fig 17.LPC-Link 2 - PN5180 board connection.Fig 18.Flashing Bootloader.binFig 19.Flash report.Fig 20.PN5180 Board Correctly installed.Fig 21.Architecture of the NFC Reader Library.Fig 22.Select the build configuration.Fig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimization.Fig 25.Edit Launch Configurations.	Fig 11.	Successful compile	11
Fig 14.Application printouts in Console windowFig 15.Debug ButtonsFig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 19.Flash reportFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Select the level of optimizationFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 12.	Select the launch configuration	12
Fig 15.Debug ButtonsFig 16.Discovery Loop in Poll mode.Fig 17.LPC-Link 2 - PN5180 board connection.Fig 18.Flashing Bootloader.binFig 19.Flash report.Fig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader Library.Fig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimization.Fig 25.Edit Launch Configurations.	Fig 13.	Debug project	12
Fig 16.Discovery Loop in Poll modeFig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 19.Flash reportFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 14.	Application printouts in Console window	13
Fig 17.LPC-Link 2 - PN5180 board connectionFig 18.Flashing Bootloader.binFig 19.Flash reportFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 15.	Debug Buttons	13
Fig 18.Flashing Bootloader.binFig 19.Flash report.Fig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 16.	Discovery Loop in Poll mode	15
Fig 19.Flash reportFig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 17.	LPC-Link 2 - PN5180 board connection	19
Fig 20.PN5180 Board Correctly installedFig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 18.	Flashing Bootloader.bin	20
Fig 21.Architecture of the NFC Reader LibraryFig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 19.	Flash report	21
Fig 22.Select the build configurationFig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 20.	PN5180 Board Correctly installed	21
Fig 23.Selection of the LPC1769 MCUFig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 21.	Architecture of the NFC Reader Library	22
Fig 24.Select the level of optimizationFig 25.Edit Launch Configurations	Fig 22.	Select the build configuration	24
Fig 25. Edit Launch Configurations	Fig 23.	Selection of the LPC1769 MCU	25
	Fig 24.	Select the level of optimization	26
Fig 26. Disable break point at main()	Fig 25.	Edit Launch Configurations	27
	Fig 26.	Disable break point at main()	27

# 10. List of tables

Table 1.	Example projects3	
Table 2.	Development Environment4	
Table 3.	Supported technologies15	

# 11. Contents

1.	Introduction3
2.	Managing the PN5180 SW projects with MCUXpresso IDE4
2.1	Development environment4
2.2	Installation procedure of the MCUXpresso IDE4
2.3	Importing provided SW example projects6
2.4	Building projects8
2.5	Running and debugging a project
3.	Managing the PN5180 SW projects with Linux
	and Kinetis platform13
4.	Associated projects14
4.1	Example 1 – Basic Discovery Loop14
4.2	Example 2 – Advanced Discovery Loop
4.3	Example 3 – NFC Forum16
4.4	Example 4 – MIFARE Classic card
	communication16
4.5	Example 5 - ISO1569316
4.6	Example 7 – EMVCo Polling17
4.7	Example 8 – HCE T4T17
4.8	Example 9 – NTAG-I2C17
4.9	Example 10 – MIFARE DESFire card
4.4.0	communication
4.10	Example 11 – ISO10373 PCD
4.11 4.12	Simplified API EMVCo
4.12	Simplified API EMVCo Analog18 Simplified API ISO18
5.	Flashing Firmware on the LPC176919
6.	Supplementary Notes
6.1	General Software Architecture
6.1.1	Bus abstraction layer22
6.1.2	Hardware abstraction layer
6.1.3 6.1.4	Protocol abstraction layer
6.1.4 6.2	Application layer
6.3	Build configuration
6.4	Setting the MCU24 Level of compiler optimization26
0.4	

6.4.1 6.5	Optimization issues Removing the initial breakpoint on debug	26
	startup	26
7.	References	28
8.	Legal information	29
8.1	Definitions	29
8.2	Disclaimers	29
8.3	Licenses	29
8.4	Trademarks	29
9.	List of figures	30
10.	List of tables	31
11.	Contents	32

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

#### © NXP B.V. 2018.

#### All rights reserved.

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

Date of release: 7 May 2018