UM11495

PN7160 NFC controller

Rev. 1.3 — 13 September 2021 **User manual**
COMPANY PUBLIC

Document information

Information	Content
Keywords	PN7160, user manual, NFC, NFCC, NCI
Abstract	This is a user manual for the PN7160 NFC controller. The aim of this document is to describe the PN7160 interfaces, modes of operation and possible configurations.

Revision history

Rev	Date	Description
1.3	20210913	Security status changed into "Company public", no content change
1.2	20210820	• PMU_CFG configuration parameter description updated • Security status changed into "Company restricted"
1.1	20210705	Editorial updates, configuration parameters default values updated and FW update mechanism description added
1.0	20210302	Initial version

1 Introduction

The PN7160 is a highly integrated transmission/reception module for contactless communication at 13.56 MHz.

The user manual describes the software interfaces (API), based on the NFC Forum NCI specification. A dedicated chapter highlights the differences between NCI 2.0 and NCI 1.0. See →Section 5.4.

Note: this document includes cross-references, which can be used to directly access the section/chapter referenced in the text. These cross-references are indicated by the following sign: '→'. This sign is positioned right before the section/chapter reference. The way to jump to the referenced section/chapter depends on the file format:

As this document assumes pre-knowledge on certain technologies, please check section →Section 3 to find the appropriate documentation.

For further information, refer to the PN7160 data sheet [PN7160_DS].

2 Abbreviations

Acronym	Description
DH	Device Host
DH-NFCEE	NFC Execution Environment running on the DH
HCI	Host Controller Interface
HCP	Host Controller Protocol
HDLL	Host Data Link Layer
LPCD	Low Power Card Detector
NCI	NFC Controller Interface
NFC	Near Field Communication
NFCC	NFC Controller
NFCEE	NFC Execution Environment
RF	Radio Frequency
RFU	Reserved For Future Use

3 References

[PN7160] PN7160 IC

[PN7160_DS] PN7160 data sheet

[NCI1.0] NFC Forum NFC Controller Interface, version 1.0

[NCI2.0] NFC Forum NFC Controller Interface, version 2.0

[NCI] Refers to both [NCI1.0] or [NCI2.0] for common generic parts

[PN7160-NCI1.0] NXP proprietary extensions to [NCI1.0]

[PN7160-NCI2.0] NXP proprietary extensions to [NCI2.0]

[NCI_Table1] Status Codes table: table 128 in [NCI1.0].

[NCI_Table2] RF technologies table: table 129 in [NCI1.0].

[NCI_Table3] RF Technology and Mode table: table 130 in [NCI1.0]

[NCI_Table4] Bit Rates table: table 131 in [NCI1.0]

[NCI_Table5] RF protocols table: table 132 in [NCI1.0]

[NCI_Table6] RF Interfaces table: table 133 in [NCI1.0]

[NCI_Table7] RF Interfaces extensions table: table 134 in [NCI]

[NCI_Table8] Configuration parameters table: table 136 in [NCI]

[NCI_Table9] CORE_RESET_NTF table: table 5 in [NCI]

[NCI_Table10] NFCEE Protocols table: table 135 in [NCI]

[NCI_Table11] Value Field of Pwr Modes: table 60 in [NCI]

[NCI_Table13] GID and OID Definitions table: table 137 in [NCI]

[NCI_Chap2] State Machine: chapter 5.2 in [NCI]

[DIGITAL] NFC Forum Digital Protocol Specification **v1.1**

[ACTIVITY1.0] NFC Forum Activity Specification **v1.0**

[ACTIVITY1.1] NFC Forum Activity Specification **v1.1**

[ACTIVITY] Refers to both [ACTIVITY1.0] or [ACTIVITY1.1] for common parts

[LLCP] Logical Link Control Protocol v1.3

[I²C] I²C -bus specification and user manual Rev 03, defined by NXP. Last revision from June 2007 can be found here: http://ics.nxp.com/support/documents/interface/pdf/i2c.bus.specification.pdf

[SPI] SPI Block Guide, Freescale — [V04.0114 July2004]

[AN13218] PN7160 RF settings guide, Application Note

[AN13224] PN7160 Dynamic power control guide, Application Note

[AN13223] PN7160 Dynamic load modulation amplitude guide, Application Note

[7816-4] ISO/IEC7816-4

4 PN7160 architecture overview

[PN7160] is an NFC controller, which is briefly described in the next figure:

- The top part describes the Device Host architecture with Higher Layer Driver (i.e Android stack) hosting the different kind of applications (Reader/Writer, Peer to Peer, Card Emulation in the DH-NFCEE), the NCI driver and the transport layer driver.
- PN7160 itself is described in the middle part. It is connected to the Device Host through a physical interface which can be either I²C or SPI. PN7160 firmware supports both [NCI1.0] and [NCI2.0] specifications, it can operate in one or the other. Depending on NCI version used, PN7160 firmware also provides support for additional extensions that are not contained in the NCI specification. These additional extensions are specific to the PN7160 chip and are proprietary to NXP. “NFCEE_NDEF” is an NFCC embedded NDEF tag emulation, configured by the DH.
- The bottom part of the figure contains the RF antenna connected to the PN7160, which can communicate over RF with a Tag (Card) and a Reader/Writer or a Peer.

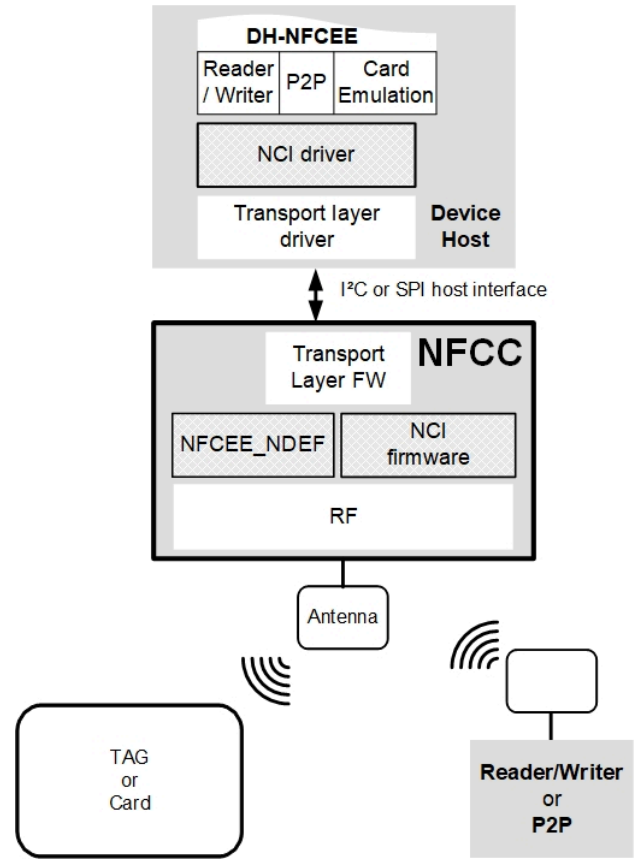

Figure 1. PN7160 system architecture

The PN7160 firmware is stored mainly in ROM, but one portion in stored in FLASH, which means that it can be updated when required.

The PN7160 user data is stored in RAM and in EEPROM. The data stored in EEPROM can also be updated.

NXP provides a download mechanism, described in chapter →Section 15, which allows downloading the binary content of the firmware stored in FLASH and the user data stored in EEPROM.

For contactless operation, several modes of operation are possible, based on the overall system described above.

4.1 Reader/Writer operation in poll mode

The Reader/Writer application running on the DH is accessing a remote contactless tag/card, through the PN7160.

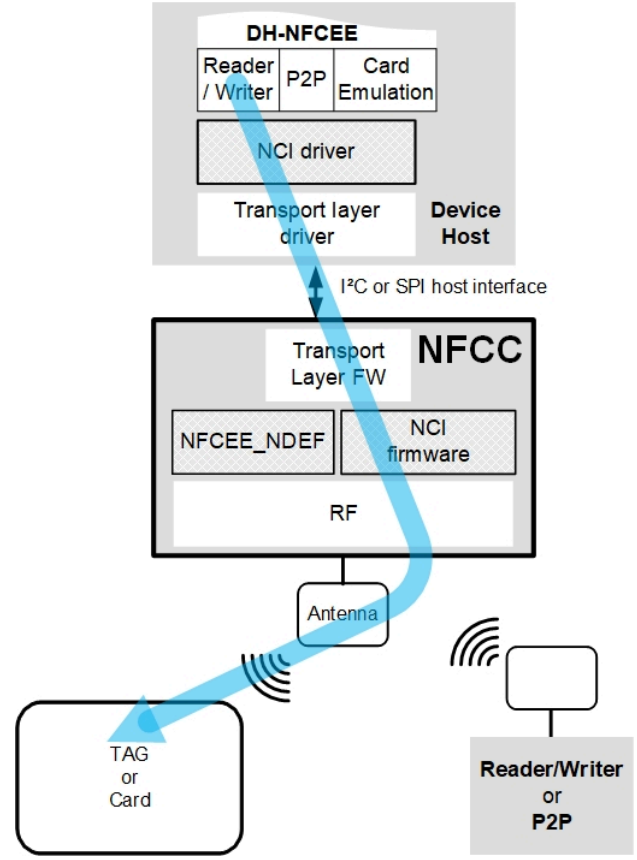

Figure 2. Reader/Writer scenario

This mode of operation is further detailed in chapter →Section 9.

4.2 Card emulation operation in listen mode

This mode of operation is further detailed in chapter →Section 10.

4.2.1 Card Emulated by the DH-NFCEE

An external Reader/Writer accesses the DH-NFCEE emulating a contactless card, through the PN7160.

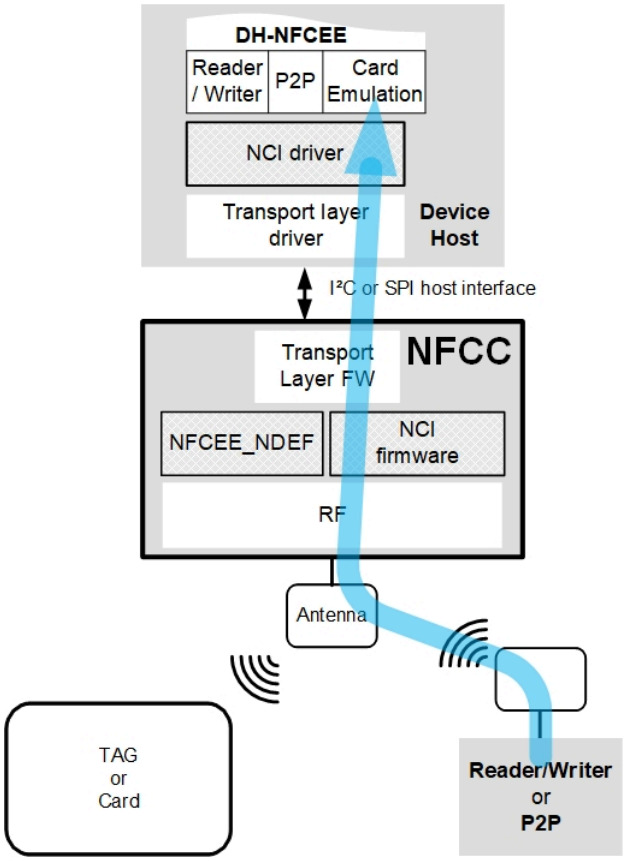

Figure 3. Card emulated by DH_NFCEE

4.2.2 Card Emulation over NFCC

A remote Reader/Writer is accessing the NDEF embedded tag hosted in “NFCEE_NDEF” through the PN7160.

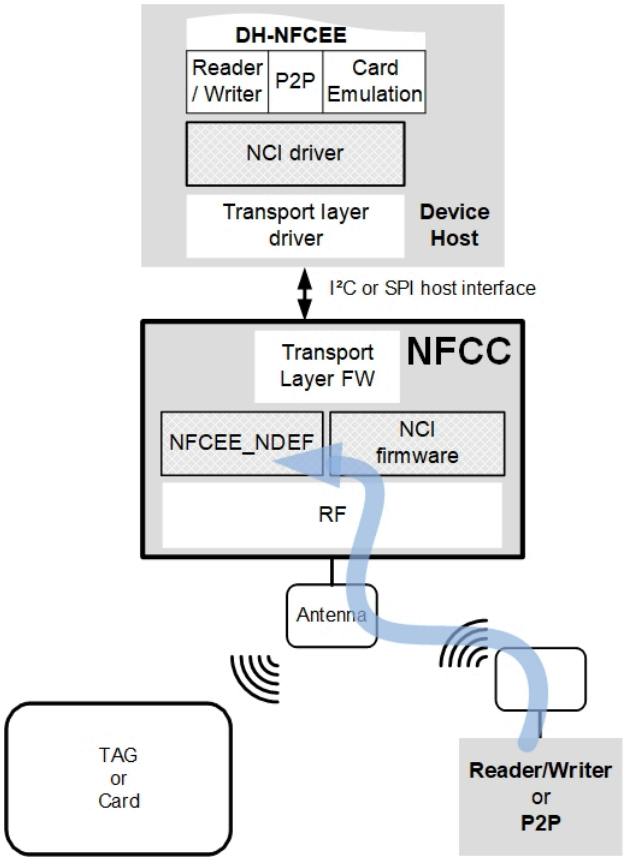

Figure 4. Card Emulation emulated by NFCCEE_NDEF

Tag of type 4 emulation is supported via NDEF T4T operation mechanism as depicted in chapter → Section 7.6

4.3 Peer to Peer operation in listen and poll mode

The P2P application running on the DH-NFCEE is accessing a remote Peer, through the PN7160.

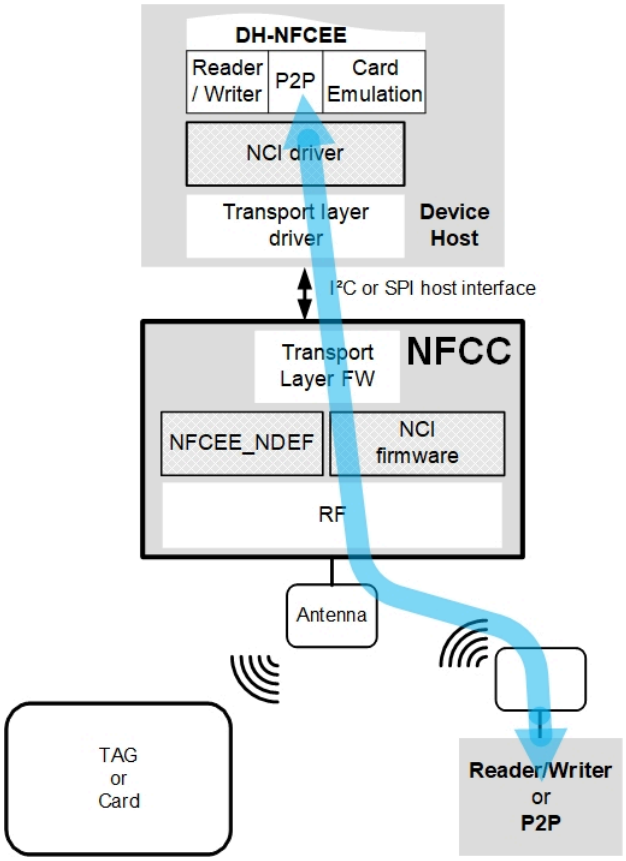

Figure 5. P2P scenario

This mode of operation is further detailed in chapter →Section 11

4.4 Combined modes of operation

Depending on the NCI version used by DH ([NCI1.0] or [NCI2.0]), the PN7160 can adapt its behavior and present different proprietary extensions. More details can be found in chapter →Section 13 about the startup sequence and the API used in each of these modes.

In any NCI mode, [PN7160] is capable of combining basic modes of operation described above, using the RF Discovery as defined in NCI. The principle used to combine the various modes of operation is to build a cyclic activity, which will sequentially activate various modes of operation:

1. Start a Polling sequence, to look for a remote Tag/Card or a remote Peer (Target). If several technologies are enabled by the DH, PN7160 will poll sequentially for all the enabled technologies.
2. If nothing was detected, enter a Listening sequence, to potentially be activated as a Card emulator or a P2P target by an external Reader/Writer or Peer.
3. If nothing happens after a programmable timeout, switch back to Poll Mode in step 1.

This cyclic activity is usually drawn as below example (where technologies NFC-A, NFC-B and NFC-F have been activated in Poll Mode):

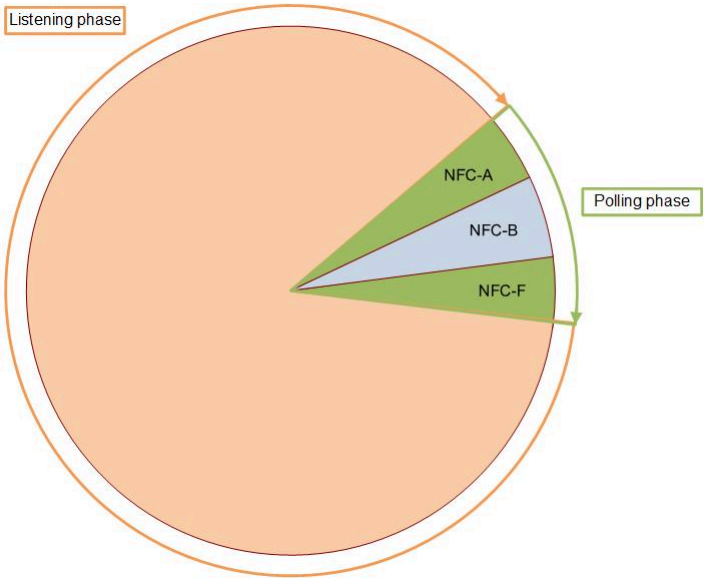

Figure 6. RF Discovery NFC Forum profile

Note that when [PN7160] is polling in Reader/Writer operation, it consumes a significant amount of current in the range of hundreds of mA (see [PN7160_DS] for accurate values). This applies for the 3 polling phases drawn on above figure and it is due to the fact that [PN7160] has to generate the RF carrier (13.56 MHz). However, during the Listen phase, [PN7160] current consumption is reduced to a few µA (see [PN7160_DS] for accurate values), due to the fact that it is waiting for the detection of an externally generated RF carrier.

Here is a figure illustrating such a cyclic RF Discovery, where polling is enabled only for NFC-A and NFC-B, for simplicity:

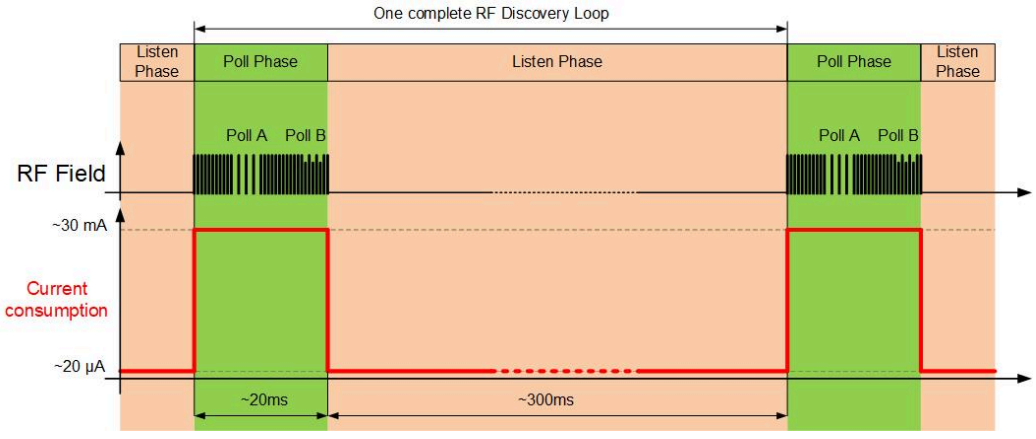

Figure 7. Power consumption during RF discovery

In a typical set-up, the polling phase is approximately 50 ms long while the listening phase is approximately 500 ms to 1 second long (this is configured thanks to the NCI parameter called TOTAL_DURATION) which delivers an average consumption of a few mA.

This average consumption can even be further optimized, using the PN7160 feature called “Tag Detector” or LPCD. See chapter →Section 12.4 for more details.

4.5 DH access to NDEF NFCEE over APDU

The PN7160 can emulate an NDEF T4T tag. The DH is in charge of filling the tag with any NDEF message. As such, the DH has a read and write access to the NDEF tag emulation in order to update the NDEF data stored in the NDEF T4T emulation tag.

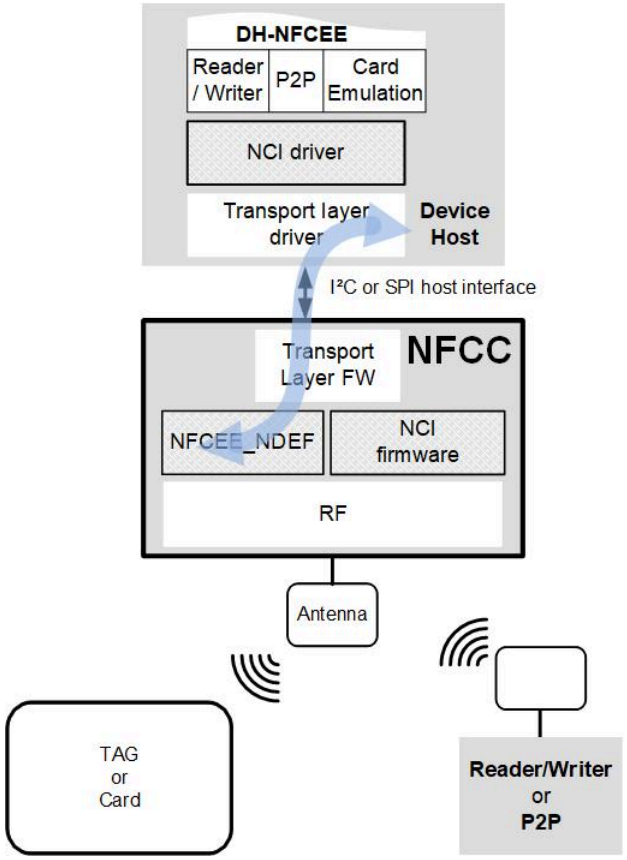

Figure 8. Exchanges between the NDEF_NFCEE and the DH over APDU NFCEE interface

NDEF message is exchanged according to [7816-4] protocol using the following Command APDU:

- select: instruction 0xA4 (Selection of applications, or files)
- Read Binary: instruction 0xB0
- Update Binary: instruction 0xD6

The DH exchanges NDEF data through the APDU NFCEE interface.

The NFCC gives exclusive access to the DH during NDEF data update procedure. This prevents external reader to access NDEF message in the same time.

A dedicated dynamic logical connection has to be created by the DH to the NFCEE_NDEF, in order to allow the transport of data between the DH and the NFCEE_NDEF.

5 NCI overview

The aim of this section is to give an overview of the key points of the [NCI] specification.

5.1 NCI components

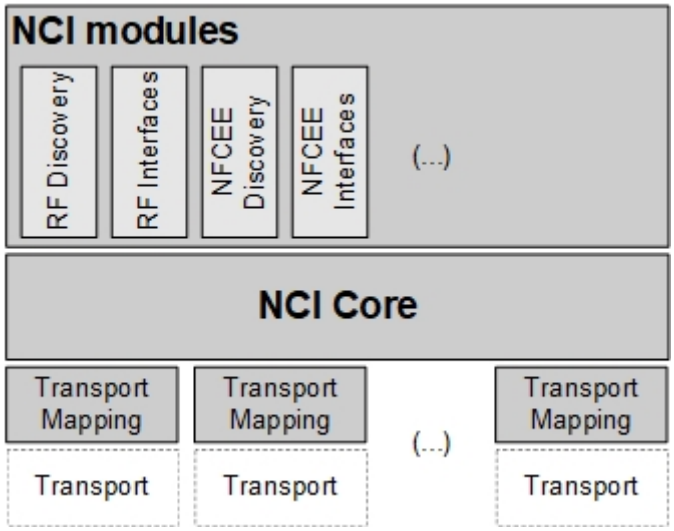

Figure 9. NCI components

5.1.1 NCI modules

NCI modules are built on top of the functionality provided by the NCI Core. Each module provides a well-defined functionality to the DH. NCI modules provide the functionality to configure the NFCC and to discover and communicate with Remote NFC Endpoints or with local NFCEEs.

Some NCI modules are mandatory parts of an NCI implementation, others are optional. There can also be dependencies between NCI modules in the sense that a module may only be useful if there are other modules implemented as well. For example, all modules that deal with communication with a Remote NFC Endpoint (the RF Interface modules) depend on the RF Discovery to be present.

5.1.2 NCI Core

The NCI Core defines the basic functionality of the communication between a Device Host (DH) and an NFC Controller (NFCC). This enables Control Message (Command, Response and Notification) and Data Message exchange between an NFCC and a DH.

5.1.3 Transport Mappings

Transport Mappings define how the NCI messaging is mapped to an underlying NCI Transport, which is a physical connection (and optional associated protocol) between the DH and the NFCC. Each Transport Mapping is associated with a specific NCI Transport.

5.2 NCI concepts

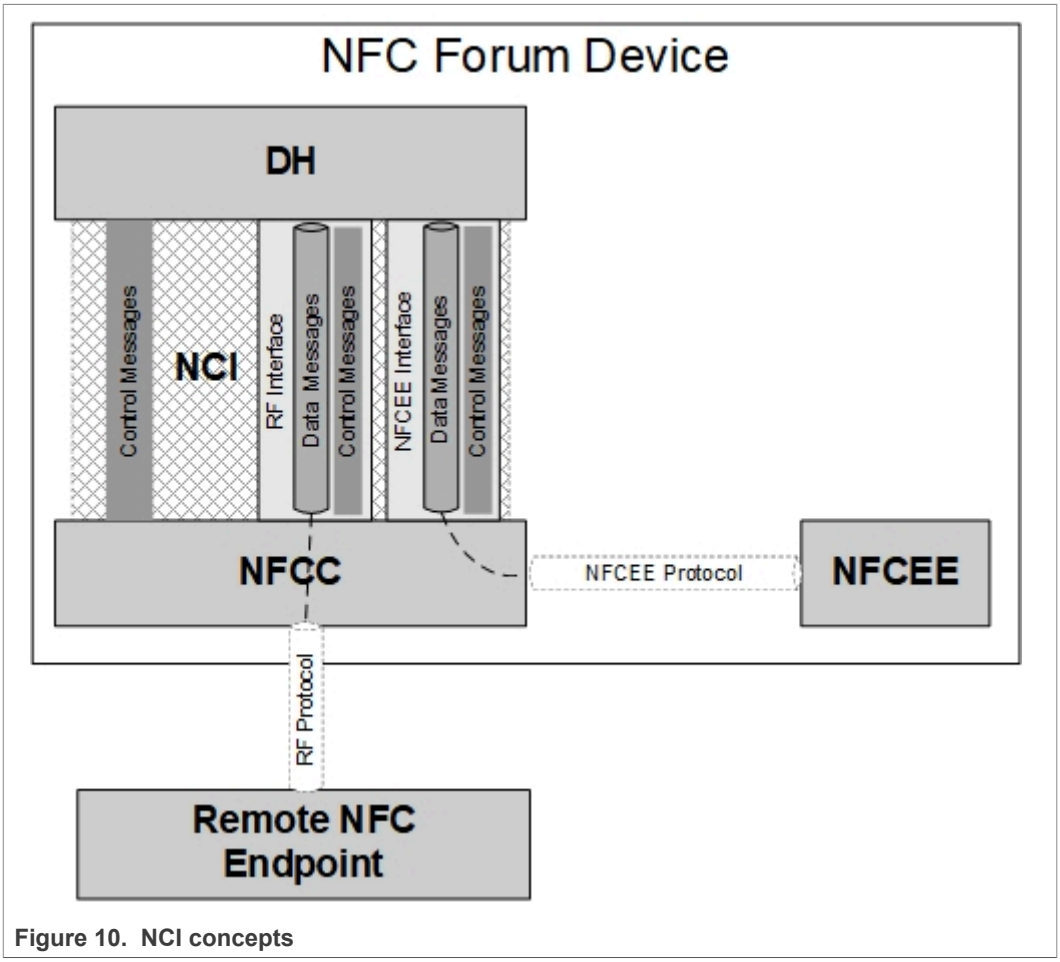

Figure 10. NCI concepts

5.2.1 Control Messages

A DH uses NCI Control Messages to control and configure an NFCC. Control Messages consist of Commands, Responses and Notifications. Commands are only allowed to be sent in the direction from DH to NFCC, Responses and Notifications are only allowed in the other direction. Control Messages are transmitted in NCI Control Packets, NCI supports segmentation of Control Messages into multiple Packets.

The NCI Core defines a basic set of Control Messages, e.g. for setting and retrieving of NFCC configuration parameters. NCI Modules can define additional Control Messages.

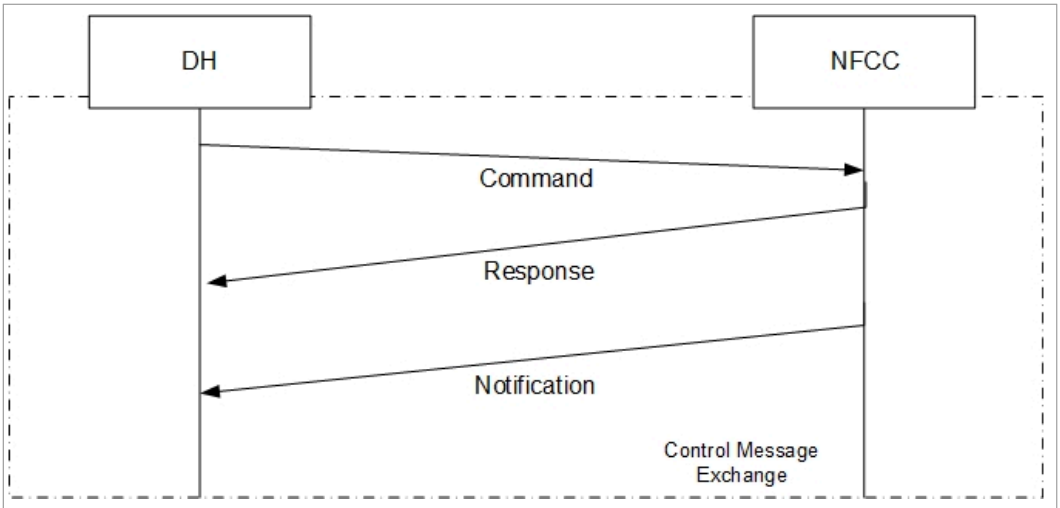

Figure 11. Control Message Exchange

5.2.2 Data Messages

Data Messages are used to transport data to either a Remote NFC Endpoint (named RF Communication in NCI) or to an NFCEE (named NFCEE Communication). NCI defines Data Packets enabling the segmentation of Data Messages into multiple Packets.

Data Messages can only be exchanged in the context of a Logical Connection. As a result, a Logical Connection must be established before any Data Messages can be sent. One Logical Connection, the Static RF Connection, is always established during initialization of NCI. The Static RF Connection is dedicated to be used for RF Communication. Additional Logical Connections can be created for RF and/or NFCEE Communication.

Since [NCI2.0], another logical connection, the Static HCI Connection, is always established during initialization of NCI. The Static HCI Connection is dedicated to be used for NFCEE communicating over HCI.

Logical Connections provide flow control for Data Messages in the direction from DH to NFCC.

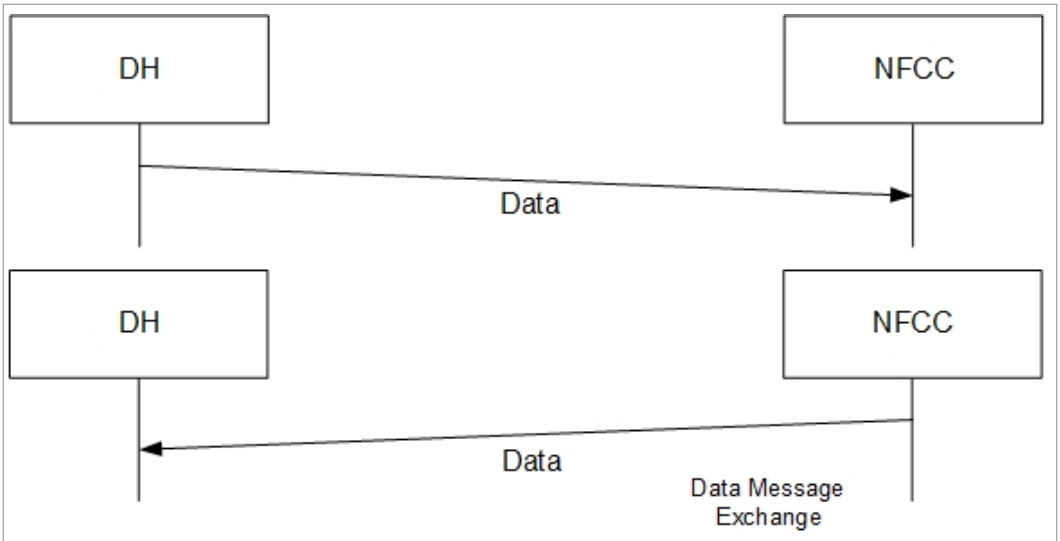

Figure 12. Data Message Exchange

5.2.3 Interfaces

An NCI Module may contain one Interface. An Interface defines how a DH can communicate via NCI with a Remote NFC Endpoint or NFCEE. Each Interface is defined to support specific protocols and can only be used for those protocols (the majority of Interfaces support exactly one protocol). NCI defines two types of Interfaces: RF Interfaces and NFCEE Interfaces.

Protocols used to communicate with a Remote NFC Endpoint are called RF Protocols. Protocols used to communicate with an NFCEE are called NFCEE Protocols.

An NFCEE Interface has a one-to-one relationship to an NFCEE Protocol, whereas there might be multiple RF Interfaces for one RF Protocol. The later allows NCI to support different splits of the protocol implementation between the NFCC and DH. An NCI implementation on an NFCC should include those RF Interfaces that match the functionality implemented on the NFCC.

Interfaces must be activated before they can be used and they must be deactivated when they are no longer used.

An Interface can define its own configuration parameters and Control Messages, but most importantly it must define how the payload of a Data Message maps to the payload of the respective RF or NFCEE Protocol and, in case of RF Communication, whether the Static RF Connection and/or Dynamic Logical Connections are used to exchange those Data Messages between the DH and the NFCC.

5.2.4 RF Communication

RF Communication is started by configuring and running the RF Discovery process. The RF Discovery is an NCI module that discovers and enumerates Remote NFC Endpoints.

For each Remote NFC Endpoint, the RF Discovery Process provides the DH with the information about the Remote NFC Endpoint gathered during the RF Discovery Process. One part of this information is the RF Protocol that is used to communicate with the Remote NFC Endpoint. During RF Discovery configuration, the DH must configure a mapping that associates an RF Interface for each RF Protocol. If only a single Remote

NFC Endpoint is detected during one discovery cycle, the RF Interface for this Endpoint is automatically activated. If there are multiple Remote NFC Endpoints detected in Poll Mode, the DH can select the Endpoint it wants to communicate with. This selection also triggers the activation of the mapped Interface.

Once an RF Interface has been activated, the DH can communicate with the Remote NFC Endpoint using the activated RF Interface. An activated RF Interface can be deactivated by either the DH or the NFCC (e.g. on behalf of the Remote NFC Endpoint). However, each RF Interface can define which of those methods are allowed. Depending on which part of the protocol stack is executed on the DH there are different deactivation options. For example, if a protocol command to tear down the communication is handled on the DH, the DH will deactivate the RF Interface. If such a command is handled on the NFCC, the NFCC will deactivate the Interface.

This specification describes the possible Control Message sequences for RF Communication in the form of a state machine.

5.2.5 NFCEE Communication

The DH can learn about the NFCEEs connected to the NFCC by using the NFCEE Discovery module. During NFCEE Discovery the NFCC assigns an identifier for each NFCEE. When the DH wants to communicate with an NFCEE, it needs to open a Logical Connection to the NFCEE (unless this logical connection is static) using the corresponding identifier and specifying the NFCEE Protocol to be used.

Opening a Logical Connection to an NFCEE automatically activates the NFCEE Interface associated to the protocol specified. When the NFCEE interface is static, it is available as soon as the NFCEE discovery process is completed. As there is always a one-to-one relationship between an NFCEE Protocol and Interface, there is no mapping step required (different as for the RF Communication).

Once the interface has been activated, the DH can communicate with the NFCEE using the activated Interface.

Closing the connection to an NFCEE Interface deactivates that NFCEE Interface.

NCI also includes functionality to allow the DH to enable or disable the communication between an NFCEE and the NFCC.

5.2.6 Identifiers

The NFCC might only be used by the DH but also by the NFCEEs in the device (in such a case the NFCC is a shared resource). NFCEEs differ in the way they are connected to the NFCC and the protocol used on such a link determines how an NFCEE can use the NFCC. For example, some protocols allow the NFCEE to provide its own configuration for RF parameters to the NFCC (similar to the NCI Configuration Parameters for RF Discovery) in other cases the NFCEE might not provide such information.

NFCCs can have different implementation in how they deal with multiple configurations from DH and NFCEEs. They might for example switch between those configurations so that only one is active at a time or they might attempt to merge the different configurations. During initialization NFCC informs the DH whether the configuration can only be under DH control or if the NFCC supports configuration by NFCEEs as well.

NCI includes a module, called Listen Mode Routing, with which the DH can define where to route received data when the device has been activated in Listen Mode. The Listen Mode Routing allows the DH to maintain a routing table on the NFCC. Routing can be

done based on the technology or protocol of the incoming traffic or based on application identifiers in case [7816-4] APDU commands are used on top of ISO-DEP.

In addition, NCI enables the DH to get informed if communication between an NFCEE and a Remote NFC Endpoint occurs.

5.3 NCI Packet Format

5.3.1 Common Packet Header

All Packets have a common header, consisting of an MT field and a PBF field:

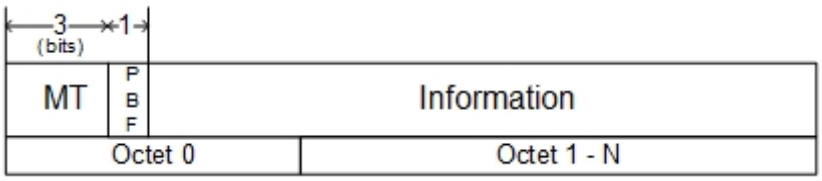

Figure 13. NCI Core Packet Format

- **Message Type (MT)**

The MT field indicates the contents of the Packet and SHALL be a 3-bit field containing one of the values listed in Table 2, below. The content of the Information field is dependent on the value of the MT field. The receiver of an MT designated as RFU SHALL silently discard the packet.

Table 1. MT values

MT	Description
000b	Data Packet
001b	Control Packet - Command Message as a payload
010b	Control Packet - Response Message as a payload
011b	Control Packet – Notification Message as a payload
100b-111b	RFU

- **Packet Boundary Flag (PBF)**

The Packet Boundary Flag (PBF) is used for Segmentation and Reassembly and SHALL be a 1-bit field containing one of the values listed in [NCI] specification.

Table 2. PBF Value

PBF	Description
0b	The Packet contains a complete Message, or the Packet contains the last segment of a segmented Message
1b	The Packet contains a segment of a Message which is not the last segment.

The following rules apply to the PBF flag in Packets:

- If the Packet contains a complete Message, the PBF SHALL be set to 0b.
- If the Packet contains the last segment of a segmented Message, the PBF SHALL be set to 0b.

- If the packet does not contain the last segment of a segmented Message, the PBF SHALL be set to 1b.

5.3.2 Control Packets

The Control Packet structure is detailed below.

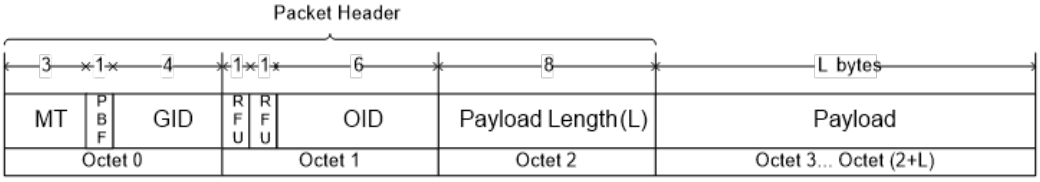

Figure 14. Control Packet Format

Each Control Packet SHALL have a 3 byte Packet Header and MAY have additional payload for carrying a Control Message or a segment of Control Message.

/!\	In the case of an 'empty' Control Message, only the Packet Header is sent.

- **Message Type (MT)**

Refer to section Section 5.3 for details of the MT field.

- **Packet Boundary Flag (PBF)**

Refer to section Section 5.3 for details of the PBF field.

- **Group Identifier (GID)**

NCI supports Commands, Responses and Notifications which are categorized according to their individual groups. The Group Identifier (GID) indicates the categorization of the message and SHALL be a 4 bit field containing one of the values listed in [NCI] specification.

All GID values not defined in [NCI] specification are RFU.

- **Opcode Identifier (OID)**

The Opcode Identifier (OID) indicates the identification of the Control Message and SHALL be a 6-bit field which is a unique identification of a set of Command, Response or Notification Messages within the group (GID). OID values are defined along with the definition of the respective Control Messages described in [NCI] specification.

- **Payload Length (L)**

The Payload Length SHALL indicate the number of bytes present in the payload. The Payload Length field SHALL be an 8-bit field containing a value from 0 to 255.

5.3.3 Data Packets

The Data Packet structure is detailed below.

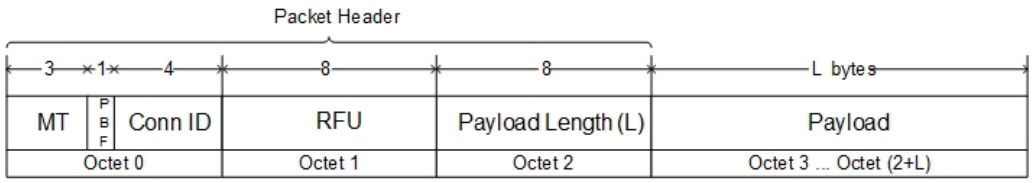

Figure 15. Data Packet Structure

Each Data Packet SHALL have a 3 byte Packet Header and MAY have additional Payload for carrying a Data Message or a segment of a Data Message.

/!\	In the case of an 'empty' Data Message, only the Packet Header is sent.

- **Message Type (MT)**

Refer to section Section 5.3 for details of the MT field.

- **Packet Boundary Flag (PBF)**

Refer to section Section 5.3 for details of the PBF field.

- **Connection Identifier (Conn ID)**

The Connection Identifier (Conn ID) SHALL be used to indicate the previously setup Logical Connection to which this data belongs. The Conn ID is a 4-bit field containing a value from 0 to 15.

- **Payload Length (L)**

The Payload Length field indicates the number of Payload bytes present. The Payload Length field is an 8-bit field containing a value from 0 to 255.

5.3.4 Segmentation and Reassembly

The Segmentation and Reassembly functionality SHALL be supported by both the DH and the NFCC.

Segmentation and Reassembly of Messages SHALL be performed independently for Control Packets and Data Packets of each Logical Connection.

Any NCI Transport Mapping is allowed to define a fixed Maximum Transmission Unit (MTU) size in bytes. If such a Mapping is defined and used, then if either DH or NFCC needs to transmit a Message (either Control or Data Message) that would generate a Packet (including Packet Header) larger than the MTU, the Segmentation and Reassembly (SAR) feature SHALL be used on the Message.

The following rules apply to segmenting Control Messages:

- For each segment of a Control Message, the header of the Control Packet SHALL contain the same MT, GID and OID values.
- **From DH to NFCC:** the Segmentation and Reassembly feature SHALL be used when sending a Command Message from the DH to the NFCC that would generate a Control Packet with a payload larger than the "Max Control Packet Payload Size" reported by the NFCC at initialization. Each segment of a Command Message except for the last SHALL contain a payload with the length of "Max Control Packet Payload Size".
- **From NFCC to DH:** when an NFCC sends a Control Message to the DH, regardless of the length, it MAY segment the Control Message into smaller Control Packets if needed for internal optimization purposes.

The following rules apply to segmenting Data Messages:

- For each segment of a Data Message, the header of the Data Packet SHALL contain the same MT and Conn ID.
- **From DH to NFCC:** if a Data Message payload size exceeds the Max Data Packet Payload Size, of the connection then the Segmentation and Reassembly feature SHALL be used on the Data Message.
- **From NFCC to DH:** when an NFCC sends a Data Message to the DH, regardless of the payload length it MAY segment the Data Message into smaller Data Packets for any internal reason, for example for transmission buffer optimization.

5.4 Differences between [NCI 2.0] and [NCI 1.0]

Here is a list of features which are changed or added in [NCI 2.0] compared to [NCI 1.0]

Table 3. Differences between NCI 2.0 and NCI 1.0

Item No.	Description	NCI 2.0	NCI 1.0
1	Backward compatibility to NCI 1.0 (Startup sequence: Reset and Init)	Supported	N/A
2	RF_INTF_ACTIVATED_NTF for T1T	Response to the RID is included in the _NTF	Response to the RID is NOT included in the _NTF
3	4-bit ACK/NACK for T2T	Differentiated from a 1-Byte RF Response, through a special Status code	Treated as a 1-Byte RF response, although it is not.
4	LLCP Low RF Extended Interface	Not Supported in PN7160	Not supported
5	Aggregated Frame RF Extended Interface	(Not Supported in PN7160)	Not supported
6	LF_PARAMETERS	18-Byte long	10-Byte-long
7	System Code-based routing (NFC-F)	Supported	Not supported
8	Poll Mode + *[Listen Mode in DH]* disabling in RFST_DISCOVERY (Screen Off)	Possible with a SET_CONFIG	Not possible
9	AFI parameter in RF_INTF_ACTIVATED_NTF for NFC-B/Listen	Supported	Not supported
10	P2P active	Supported	Proprietary implementation
11	ISO 15693/T5T	Supported	Proprietary implementation
12	Credit Piggybacking	Supported	Not supported
13	Reader 144443-4 Presence Check	Supported	Proprietary implementation
14	Tag deactivation in Frame RF Interface	Supported	Proprietary implementation

PN7160 implements the features as defined in NCI 2.0, including mechanism defined to ensure that PN7160 can be connected to a DH implementing NCI1.0.

If PN7160 is delivered in NCI2.0 mode by default, the mechanism to use NCI1.0 is then the following:

- PN7160 indicates it implements NCI 2.0 in the CORE_RESET_RSP
- The DH will then send the CORE_INIT_CMD using the format defined in NCI 1.0
- PN7160 understands that the DH is not NCI 2.0 ready, and automatically moves all the features which are changed from NCI 1.0 to NCI 2.0.

6 Device Host Interface Link

6.1 Overview

The PN7160 can support either I²C or SPI as physical interface used to connect to the DH, depending on the part number ordered.

Independent of the physical interface, the PN7160 has two main modes of operation to communicate with the DH:

1. NCI-based communications
2. HDLL-Based communications, only used when the PN7160 is triggered to enter the “download mode”, to update its firmware.

The description of the transport layer in the next chapters is limited to NCI-based communications. For further information on the HDLL-based communications, please refer to chapter →Section 15.

/!\	In the next chapters, the IRQ pin is drawn as being active High (=> a logical '1' on IRQ indicates that PN7160 is willing to trigger a Read sequence). Since the polarity of the IRQ pin is configurable (see parameter IRQ_POLARITY_CFG in →Section 13.1), the IRQ pin may be active with a logical '0' level.

6.2 I2C Interface

6.2.1 Introduction

The I²C interface of the PN7160 is compliant with the [I²C] Bus Specification, including device ID and Soft Reset. It is slave-only, i.e. the SCL signal is an input driven by the host.

/!\	NCI packets can be as long as 258 Bytes in both directions. The DH shall consider this constraint when implementing its I²C driver.

The PN7160 I²C interface supports standard (up to 100 kbps), fast-Speed mode (up to 400 kbps) and High-Speed mode (up to 3.4 Mbit/s).

I²C defines two different modes of addressing (7-bit and 10-bit). The PN7160 only supports the 7-bit addressing mode.

Following names are used in the document:

Table 4. I²C pins correspondence

Pin name	Pin correspondence
I2C_ADDR0	Equivalent to pin I2CADR0_SPINSS of PN7160 when using I²C
I2C_ADDR1	Equivalent to pin I2CADR1_SPIMOSI of PN7160 when using I²C
SDA	Equivalent to pin I2CSDA_SPIMISO of PN7160 when using I²C
SCL	Equivalent to pin I2CSCL_SPISCK of PN7160 when using I²C

The PN7160 I²C 7-bit address can be configured from 0x28 to 0x2B. The 2 least significant bits of the slave address are electrically forced by pins I2C_ADDR0 and I2C_ADDR1 of the PN7160.

So PN7160 slave 7-bit address is in binary format: “0 1 0 1 0 I2C_ADDR1 I2C_ADDR0”

Table 5. PN7160 I²C slave address

I2C_ADDR1 Pin	I2C_ADDR0 Pin	Address Value	I2C-bus address (R/W = 0, write)	I2C-bus address (R/W = 1, read)
0	0	0x28	0x50	0x51
0	1	0x29	0x52	0x53
1	0	0x2A	0x54	0x55
1	1	0x2B	0x56	0x57

This can be easily configured through direct connection of pins I2C_ADDR0 and I2C_ADDR1 to either GND or PVDD at PCB level.

6.2.2 NCI Transport Mapping

In the PN7160, there is no additional framing added for I²C: an NCI packet (either data or control message, as defined in chapter →Section 5.3) is transmitted over I²C “as is”, i.e. without any additional Byte (no header, no CRC etc…).

6.2.3 Write Sequence from the DH

As the I²C clock is mastered by the DH, only the DH can initiate an I²C exchange.

A DH write sequence always starts with the sending of the PN7160 I²C Slave Address followed by the write bit (logical ‘0’: 0b). Then the PN7160 I²C interface sends an I²C ACK back to the DH for each data byte written by the DH.

It may send an I²C NACK (negative acknowledge) when none of the reception buffers used by the NCI core in the PN7160 is free, which may happen in case PN7160 is in standby mode. If one single byte of a complete NCI frame is NACKed by the PN7160, the DH has to re-send the complete NCI frame and not only this single byte.

In case DH does not have the possibility to retransmit the complete frame, it can also activate a wake-up pin named WUP_REQ to wake up PN7160 and send the frame after Tboot=5 ms.

To activate this option and to use the WUP_REQ pin for host interface wake-up, one has to first set the proprietary parameter “IRQ_POLARITY_CFG” with a NCI CORE_SET_CONFIG_CMD command. This is described in details in chapter 14 “Configurations”. By default the option will be turned off and the wake-up via WUP_REQ will be disabled.

WUP_REQ also influences the standby mode: As long as the WUP_REQ pin is asserted high, the PN7160 will never go into standby.

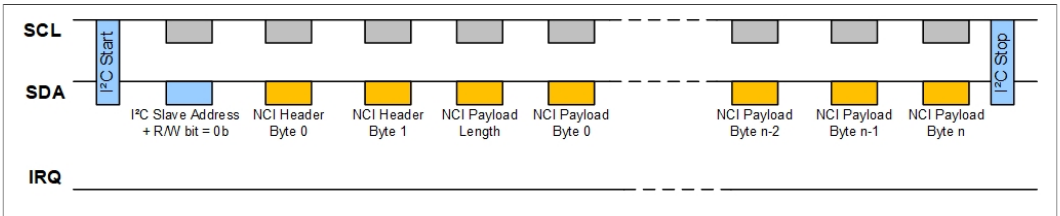

Figure 16. I²C Write sequence

/!\	It may happen that PN7160 has an NCI Message ready to be sent to the DH while it is receiving another NCI Message from the DH. In such a condition, the IRQ pin will be raised somewhere during the Write Sequence: this is not an error and has to be accepted by the DH: once the Write Sequence is completed, the DH has to start a Read Sequence (see →Section 6.2.4).

6.2.4 Read Sequence from the DH

The DH shall never initiate a spontaneous I²C read request. The DH shall wait until it is triggered by the PN7160. To trigger the DH, the PN7160 generates a logical transition from Low to High on its IRQ pin (if the IRQ pin is configured to be active High; see configuration chapter →Section 13). So after writing any NCI command, the DH shall wait until the PN7160 raises its IRQ pin. The DH can then transmit a Read request to fetch the NCI answer from the PN7160. When the PN7160 needs to send a spontaneous notification to the DH (for instance an RF Interface activation notification), the PN7160 raises the IRQ pin and the DH performs a normal read as described above.

A DH Read Sequence always starts by the sending of the PN7160 I²C Slave Address followed by the read bit (logical '1'). Then the DH I²C interface sends an ACK back to the PN7160 for each data Byte received.

The figure below is an example where the IRQ is raised so the DH can proceed a read.

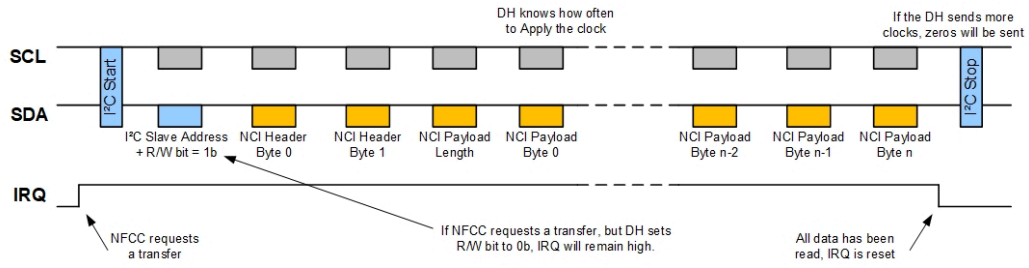

Figure 17. I²C Read sequence

As indicated on the figure above, in case the PN7160 requests a data transfer by raising the IRQ pin and the DH tries to initiate a write sequence by positioning the write bit to 0b, the PN7160 keeps the IRQ active until the DH starts a read sequence. The DH is not allowed to proceed with a write sequence once the PN7160 has set the IRQ pin to its active value (logical '1' in the figure above).

If PN7160 has another message ready to be sent to the DH before the end of the on-going Read Sequence, the IRQ pin will be first deactivated at the end of the on-going Read Sequence and then re-activated to notify to the DH that a new message has to be read.

6.2.5 Split mode

The PN7160 supports the interruption of a frame transfer, as defined in [I²C]. This feature is only available in Read Mode; it is RFU to use it in Write Mode.

This can be useful in a system where the I²C bus is shared between several peripherals: it allows the host to stop an on-going exchange, to switch to another peripheral (with a different slave address) and then to resume the communication with the PN7160.

Another typical use-case for the split mode is to have the DH reading first the NCI packet header, to know what the Payload length is. The DH can then allocate a buffer with an

appropriate size and read the payload data to fill this buffer. This use-case is represented on the following picture:

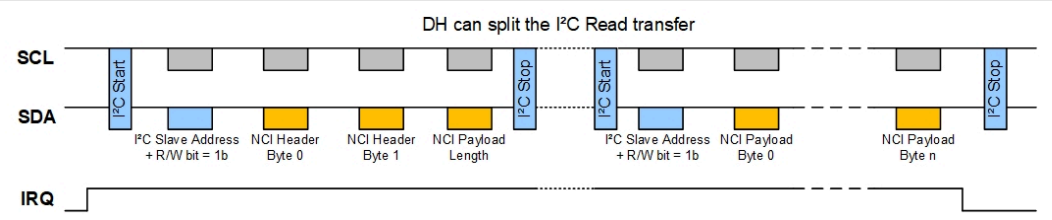

Figure 18. I²C Read sequence with split mode

6.3 SPI Interface

6.3.1 Introduction

The PN7160 slave-only SPI interface is compliant with the Freescale standard (see [SPI]).

- SPI speed up to 7 Mbit/s
- 8-bit data format only
- Supports all 4 modes of SPI (CPOL and CPHA)
- If no data is available the MISO line will be kept idle high (sends 0xFF bytes)
- Toggling the NSS line indicates a new frame

It is restricted to half-duplex communications.

Table 6. SPI pins correspondence

Pin name	Pin correspondence
NSS (HIF1)	Equivalent to pin I2CADR0_SPINSS of the PN7160 when using SPI
MOSI (HIF2)	Equivalent to pin I2CADR1_SPIMOSI of the PN7160 when using SPI
MISO (HIF3)	Equivalent to pin I2CSDA_SPIMISO of the PN7160 when using SPI
SCK (HIF4)	Equivalent to pin I2CSCL_SPISCK of the PN7160 when using SPI

6.3.2 NCI Transport Mapping

A header Byte is added to the NCI packets (either data or control message, as defined in chapter →Section 5.3).

So each data transfer over SPI starts with a header Byte which is called a “transfer direction detector”.

Here is the meaning of this Byte, depending on its value:

Table 7. PN7160 Transfer Direction Detector

Byte value	Meaning
0XXXXXXXb	DH Write access
11111111b (0xFF)	DH Read access

So this header Byte restricts the full-duplex capabilities of SPI to half-duplex.

6.3.3 Write Sequence from the DH

As the SPI clock is mastered by the DH, only the DH can initiate an SPI exchange.

A DH write sequence always starts with the sending of the Transfer Direction Detector Byte = 0XXXXXXXb. The PN7160 will consider all the following Bytes as part of an NCI packet.

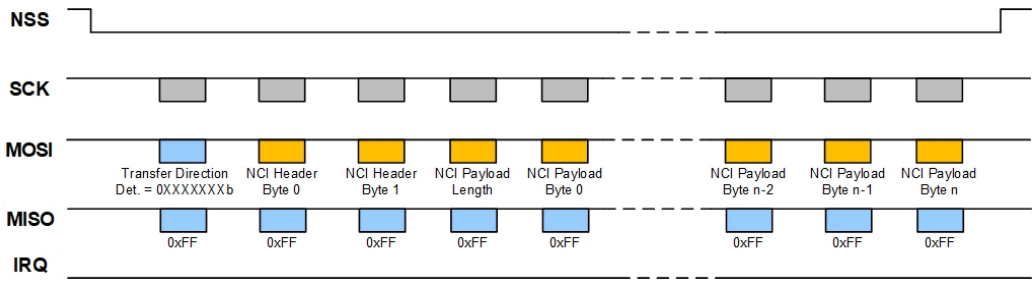

Figure 19. Example of a SPI Write access

/!\	It may happen that PN7160 has an NCI Message ready to be sent to the DH while it is receiving another NCI Message from the DH. In such a condition, the IRQ pin will be raised somewhere during the Write Sequence: this is not an error and has to be accepted by the DH: once the Write Sequence is completed, the DH has to start a Read Sequence.

Issues may happen, in case PN7160 is in standby mode when the DH is writing a new frame: in such a condition, PN7160 will not be able to catch the received frame.

In order to detect that PN7160 is not ready to receive a frame, the DH has to monitor the MISO line, to check the value of the 1st Byte received on MISO while it is writing data on the MOSI line. When the Write sequence starts:

- If the first Byte received on MISO is 0xFF, PN7160 is ready and the Write sequence will proceed with no issue.
- If the first Byte received on MISO is any other value than 0xFF, PN7160 is not ready to receive a new frame. The DH has to resend the whole NCI frame, after a typical timeout of 5ms.

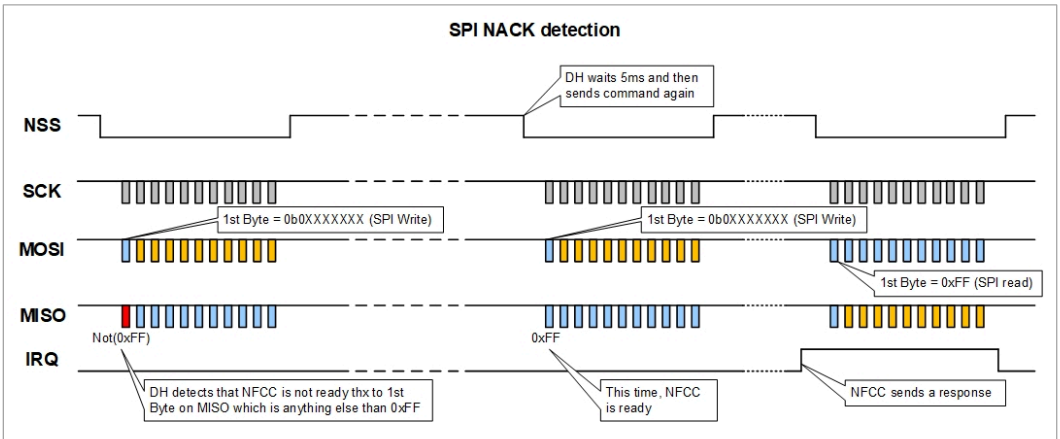

Figure 20. SPI Write access in case PN7160 is in standby

6.3.4 Read Sequence from the DH

The DH shall never initiate a spontaneous SPI read request. The DH shall wait until it is triggered by the PN7160. To trigger the DH, the PN7160 generates a logical transition from Low to High on its IRQ pin (if the IRQ pin is configured to be active High; see configuration chapter →Section 13). So after writing any NCI command, the DH shall wait until the PN7160 raises its IRQ pin. The DH can then transmit a Read request to fetch the NCI answer from the PN7160.

A DH Read Sequence always starts by the sending of the Transfer Direction Detector Byte = 0xFF.

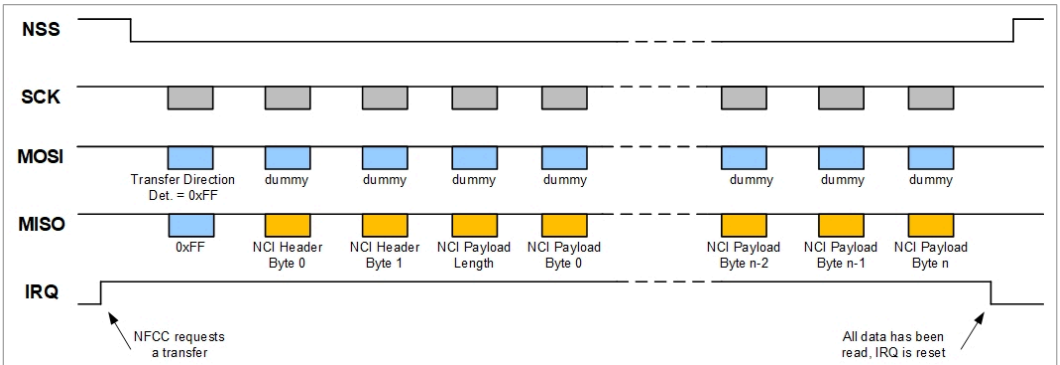

Figure 21. Example of a SPI Read access

As indicated on the figure above, in case the PN7160 requests a data transfer by raising the IRQ pin and the DH tries to initiate a write sequence by positioning the Transfer Direction Byte to 0b0XXXXXXX, the PN7160 keeps the IRQ to logical '1' until the DH starts a read sequence. The DH is not allowed to proceed with a write sequence once the PN7160 has set the IRQ pin to its active value (logical '1' in the figure above).

If PN7160 has another message ready to be sent to the DH before the end of the on-going Read Sequence, the IRQ pin will be first deactivated at the end of the on-going Read Sequence and then re-activated to notify to the DH that a new message has to be read.

6.3.5 Split mode

The PN7160 supports the interruption of a frame transfer over SPI. This feature is only available in Read Mode; it is RFU to use it in Write Mode.

A typical use-case for the split mode is to have the DH reading first the NCI packet header, to know what the Payload length is. The DH can then allocate a buffer with an appropriate size and read the payload data to fill this buffer. This use-case is represented on the following picture:

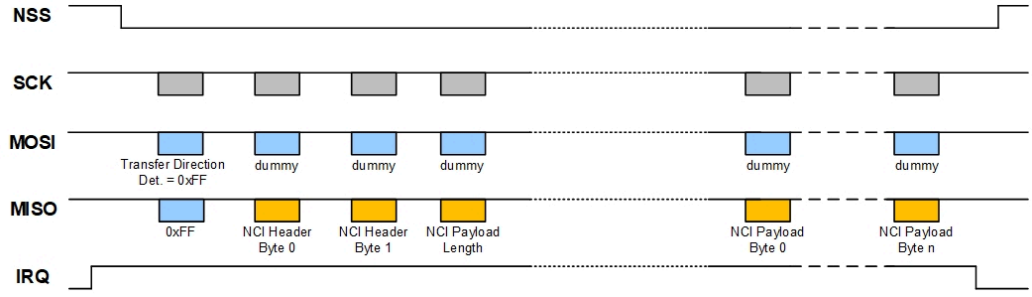

Figure 22. SPI Read sequence with split mode

6.3.6 Invalid Sequence from the DH

Any SPI data transfer starting by a Transfer Direction Detector Byte different from either 0XXXXXXXb or 11111111b is discarded by PN7160, as this is an invalid frame.

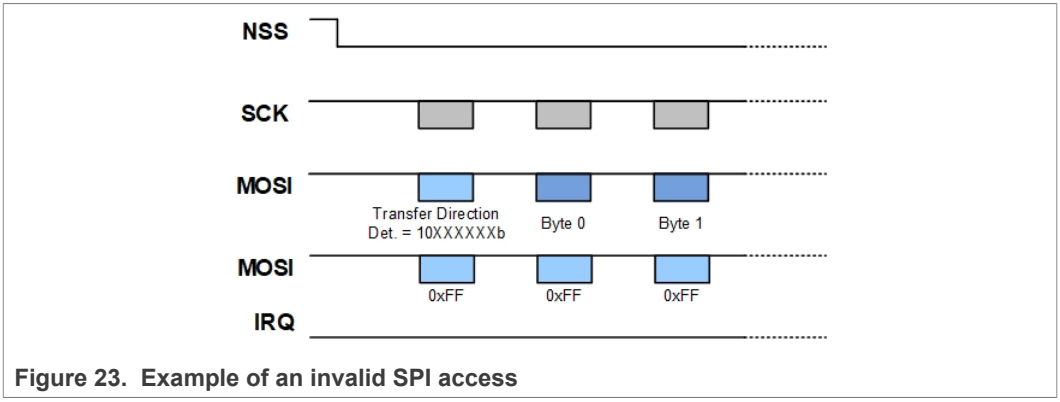

Figure 23. Example of an invalid SPI access

7 Compliance to [NCI] and PN7160 extensions

The PN7160 is a complex contactless System on Chip, which offers a lot of features. Unfortunately, [NCI] as defined by the NFC Forum refers in this document to both [NCI1.0] and [NCI2.0] but does not give full access to all possible features. Therefore, NXP had to extend [NCI] with proprietary extensions, and the PN7160 interface which includes [NCI] plus the PN7160 extensions is referenced in the present document as [PN7160-NCI1.0] when PN7160 runs in NCI1.0 mode and [PN7160-NCI2.0] when it runs in NCI2.0 mode.

7.1 Feature-based comparison of [NCI] and [PN7160-NCI]

The table below represents the features overview of the PN7160. It highlights the main differences between the NCI standard ([NCI]) and [PN7160-NCI]. The Chapter column contains shortcuts to the section in the document where the feature is described in details.

7.2 Features actually available in [PN7160]

Table 8. RF features list

Mode	Protocol	Techno	NFCEE	Other	Chapter
Card Emu.	ISO-DEP	NFC-A	DH-NFCEE	ISO-DEP RF IF 106 kb/s	→Section 10.1.2
			NFCEE-NDEF	ISO-DEP RF IF 106 kb/s	→Section 7.6
		NFC-B	DH-NFCEE	ISO-DEP RF IF 106 kb/s	→Section 10.1.2
			NFCEE-NDEF	ISO-DEP RF IF 106 kb/s	→Section 7.6
	T3T	NFC-F	DH-NFCEE	Frame RF IF	→Section 10.2.3
R/W	ISO-DEP	NFC-A	DH-NFCEE	Frame RF IF 106	→Section 9.3.1
				ISO-DEP RF IF 106, 212, 424, 848 kb/s	→Section 9.3.2
		NFC-B	DH-NFCEE	Frame RF IF 106	→Section 9.3.1
				ISO-DEP RF IF 106, 212, 424, 848 kb/s	→Section 9.3.2
	MIFARE Classic MIFARE Plus	NFC-A	DH-NFCEE	TAG-CMD IF 106 kb/s (proprietary interface)	→Section 9.1.1
	T1T	NFC-A	DH-NFCEE	Frame RF IF 106 kb/s	→Section 9.1.1
	T2T	NFC-A	DH-NFCEE	Frame RF IF 106 kb/s	→Section 9.1.1
	FeliCa / T3T	NFC-F	DH-NFCEE	Frame RF IF 212, 424 kb/s	→Section 9.2.1
	ISO15693 / T5T	ISO15693 / NFC-V	DH-NFCEE	Frame RF IF 26 kb/s	→Section 9.4.1
	Kovio	Kovio	DH-NFCEE	106 kb/s (proprietary interface)	→Section 9.5.1
	Specific Tag	NFC-A / NFC-B	DH-NFCEE	106 kb/s (proprietary interface)	
P2P Target	NFC-DEP	NFC-A Passive	DH-NFCEE	NFC-DEP IF 106 kb/s	→Section 11.1.1
		NFC-A Active	DH-NFCEE	NFC-DEP IF 106 kb/s	→Section 11.1.2
		NFC-F Passive	DH-NFCEE	NFC-DEP IF 212, 424 kb/s	→Section 11.1.1

Table 8. RF features list...*continued*

Mode	Protocol	Techno	NFCEE	Other	Chapter
		NFC-F Active	DH-NFCEE	NFC-DEP IF 212, 424 kb/s	→Section 11.1.2
P2P Initiator	NFC-DEP	NFC-A Passive	DH-NFCEE	NFC-DEP IF 106 kb/s	→Section 11.1.1
		NFC-A Active	DH-NFCEE	NFC-DEP IF 106 kb/s	→Section 11.1.2
		NFC-F Passive	DH-NFCEE	NFC-DEP IF 212, 424 kb/s	→Section 11.1.1
		NFC-F Active	DH-NFCEE	NFC-DEP IF 212, 424 kb/s	→Section 11.1.2

Table 9. Other features list

Feature	Chapter
RF Booster, DC-DC (also named “Config 2”)	→Section 8.6.2
RF Discovery Activity (NFC Forum, EMVCo)	→Section 12
Clock management	→Section 8.5
Power Management	→Section 13
Antenna self-Test	→Section 14.2
PRBS Test	→Section 14.3
Secure Firmware Upload mode	→Section 15
EMVCo profile for PICC	→Section 12.5.2
EMVCo profile for PCD	→Section 12.5.1
Low Power Card Detector (LPCD)	→Section 12.4
14443-4 Presence Check ([PN7160-NCI1.0])	→Section 9.3.3

7.3 [NCI] Implementation in the PN7160

[NCI] defines several features which are optional or configurable. For instance, data exchange can use an optional flow control, for which the number of credits is defined by the NFCC. The maximum number of simultaneous Dynamic logical connections is also up to the NFCC.

So the intent of this section is to describe those features in [NCI] which are optional or configured by the NFCC, to highlight how they are implemented in the PN7160.

7.3.1 Logical connections and credits

Here is a simplified overview of an NFC device as defined in the NFC Forum:

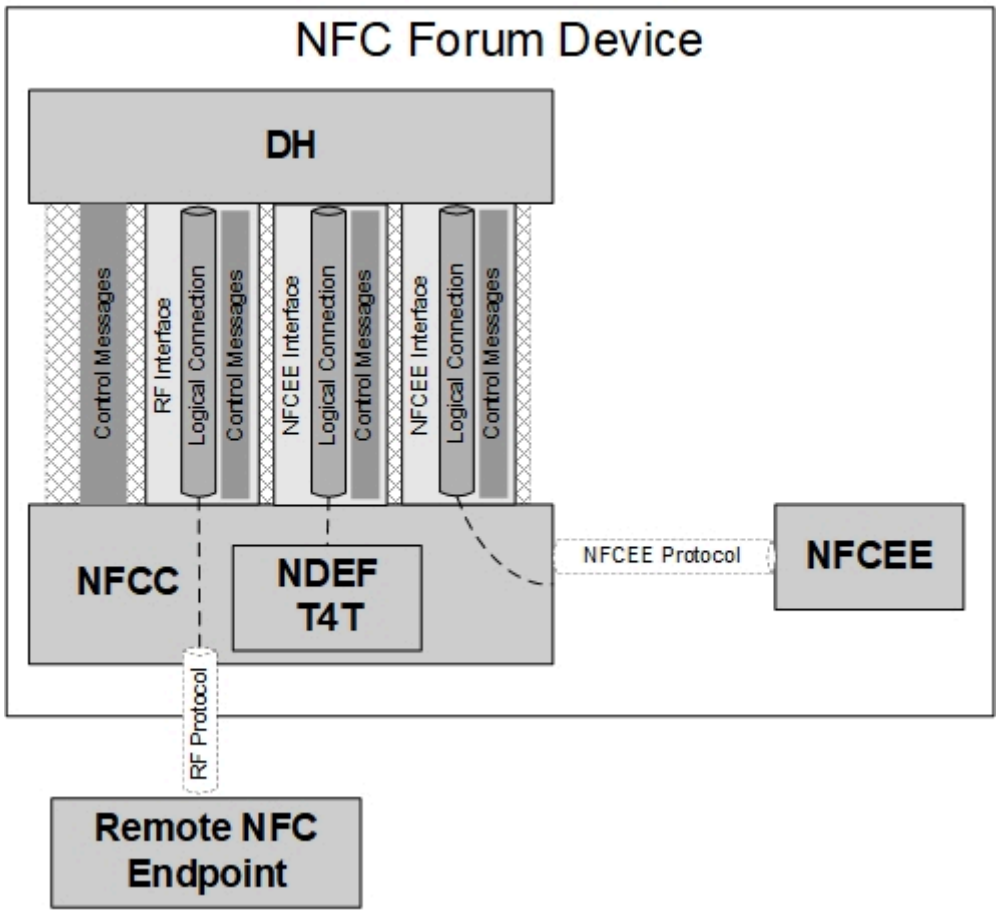

Figure 24. NFC Forum Device architecture

Logical connections are used to transport data between the DH and the NFCC. Although optional in [NCI], PN7160 implements data flow control based on credits management. In order to minimize the required buffer/memory size, the **number of credits is limited to 1** on each logical connection.

In addition to the mandatory static RF logical connection and static HCI connection (only for [NCI2.0]), PN7160 **allows to create only 1 additional** dynamic logical connection. So, the “Max Logical Connections” parameter reported in **CORE_INIT_RSP** equals **0x01** for PN7160 and the number of credits of the static HCI Connection equals to 0x01. That means that when the DH needs to create a new logical connection, it has first to close the currently opened one, if any.

Here is an overview of the logical connection and credits available in the PN7160:

Table 10. Logical Connections/Credits configuration

Logical connection	Number of connections	Number of credits	Max. Data Packet payload Size
Static connection	- 1 for RF End Point	1	[255]
Dynamic connection	- 1 for NDEF T4T emulation - 1 for NCI Loopback testing	1 1	[255] [255]

In order to optimize the number of PN7160 internal buffers, all the logical connections shall not be used at the same time. Therefore, only the following scenarios are possible at the same time depending on the RF State Machine:

Table 11. Logical Connections supported depending on RF State Machine

NCI RF States	NCI CMD	Static RF EndPoint	NDEF T4T / Loopback
RFST_IDLE	Supported	Not supported	Supported
Others RF States	Supported	Supported	Not supported

7.3.2 Compliance to [NCI] control messages

Here is a detailed status, for the current version PN7160:

Table 12. Status on the compliance to [NCI] control messages

Group	Control messages	Status
CORE	CORE_RESET_CMD / RSP / NTF (updated in [NCI2.0])	**Full Support** [1]
	CORE_INIT_CMD / RSP (updated in [NCI2.0])	**Full Support**
	CORE_SET_CONFIG_CMD / RSP	**Full Support**
	CORE_GET_CONFIG_CMD / RSP	**Full Support**
	CORE_CONN_CREATE_CMD / RSP	**Partial Support**[2]
	CORE_CONN_CLOSE_CMD / RSP	**Full Support**
	CORE_CONN_CREDITS_NTF	**Full Support**
	CORE_GENERIC_ERROR_NTF	**Full Support**
	CORE_INTERFACE_ERROR_NTF	**Full Support**
	CORE_SET_POWER_SUB_STATE_CMD/RSP (only in [NCI2.0])	**Full Support**
RF	RF_DISCOVER_MAP_CMD / RSP	**Full Support**
	RF_SET_LISTEN_MODE_ROUTING_CMD / RSP	**Full Support**
	RF_GET_LISTEN_MODE_ROUTING_CMD / RSP / NTF	**Full Support**
	RF_DISCOVER_CMD / RSP / NTF	**Full Support**
	RF_DISCOVER_SELECT_CMD / RSP	**Full Support**
	RF_INTF_ACTIVATED_NTF	**Full Support**
	RF_DEACTIVATE_CMD / RSP / NTF	**Full Support**
	RF_FIELD_INFO_NTF	**Full Support**
	RF_T3T_POLLING_CMD / RSP / NTF	**Full Support**
	RF_NFCEE_ACTION_NTF	**Full Support**
	RF_NFCEE_DISCOVERY_REQ_NTF	**Full Support**
	RF_PARAMETER_UPDATE_CMD / RSP	**No Support** [3]
	RF_INTF_EXT_START_CMD / RSP (new in [NCI2.0])	**No Support**
	RF_INTF_EXT_STOP_CMD / RSP (new in [NCI2.0])	**No Support**
	RF_EXT_AGG_ABORT_CMD / RSP (new in [NCI2.0])	**No Support**
	RF_NDEF_ABORT_CMD / RSP (new in [NCI2.0])	**No Support**

Table 12. Status on the compliance to [NCI] control messages...*continued*

Group	Control messages	Status
	RF_ISO_DEP_NAK_PRESENCE_CMD / RSP / NTF (new in [NCI2.0])	**Full Support**
NFCEE	NFCEE_DISCOVER_CMD / RSP / NTF	**Full Support**
	NFCEE_MODE_SET_CMD / RSP / NTF (NTF new in [NCI2.0])	**Full Support**

[1] CORE_RESET_NTF has sometimes an additional field, not compliant to [NCI]. See →Section 8.1.

[2] The Destination Specific parameter of Type 0x00 is not supported but it does not prevent any use case.

[3] RF_PARAMETER_UPDATE_CMD is not fully supported as corresponding scenarios can automatically be fulfilled when using ISO-DEP RF Interface and NFC-DEP RF Interface.

7.3.3 Compliance to [NCI] RF interfaces

Here is a drawing of the RF interfaces available in [NCI]:

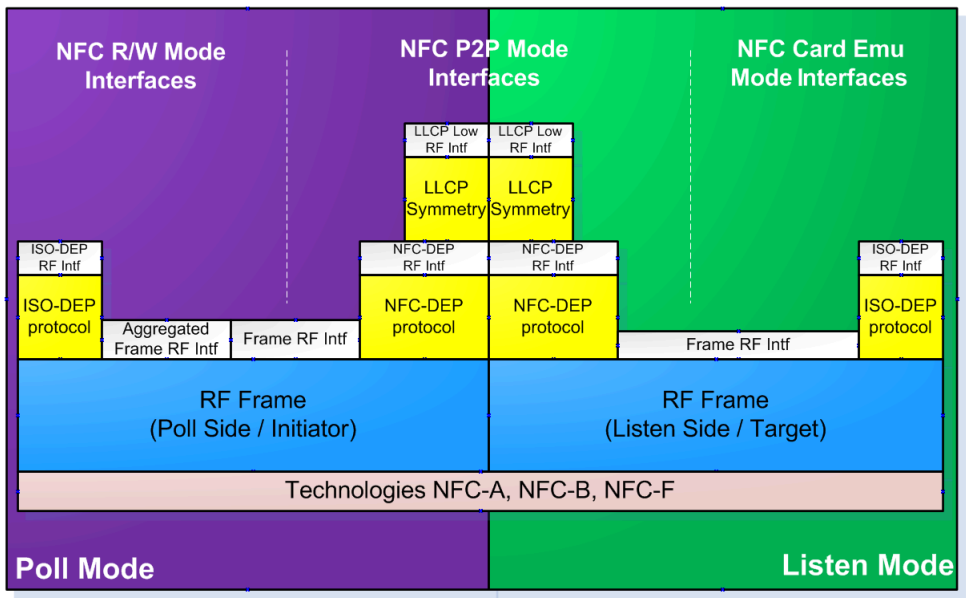

Figure 25. [NCI] RF Interface Architecture

This section details the status on the different RF interfaces supported by the PN7160.

Table 13. NCI Interface limitations

RF Interface present in [NCI]	Status
Poll side and Listen side Frame RF interface	**Partial Support**[1]
Poll side and Listen side ISO-DEP interface	**Full support**
Poll side and Listen side NFC-DEP interface	**Full support**
Poll side and Listen side NFC-DEP interface with LLCP-Low extension (new in 2.0)	**Full support**

Table 13. NCI Interface limitations...*continued*

RF Interface present in [NCI]	Status
Poll side Frame RF interface with frame Aggregated extension (new in 2.0)	**Full support**

[1] Only Poll side supported, P2P Passive and Active and ISO-DEP Listen side are not supported in Frame RF Interface.

7.3.4 Compliance to [NCI] RF Discovery

[NCI] relies on the [ACTIVITY] specification defined by the NFC Forum.

[ACTIVITY1.1] is applied by default on PN7160.

7.3.5 Compliance to [NCI] configuration parameters

[NCI] defines a set of configuration parameters, in [NCI_Table8] (see chapter →Section 3). Most of them are supported by PN7160; however, a subset of these parameters is not supported.

Here is a status for all these parameters, together with their default value in PN7160:

Table 14. Compliance to [NCI] configuration parameters

Config parameters	Status on PN7160	Coming from	Default value	Remark
TOTAL DURATION	**Full support**	[NCI]	0x03E8 (1s)	Minimum Value: 350 ms Maximum Value: 65 seconds
CON_DEVICE_ LIMIT	**Full support**	[ACTIVITY] [NCI1.0]	0x03	Parameter is Read Only, Value is set to 3, except for ISO15693 where it is limited to 2 VICCs
PA_DEVICES_ LIMIT	**Full support**	[ACTIVITY] [NCI 2.0]	0x02	Derived from CON_DEVICE_LIMIT in NCI 1.0
PB_DEVICES_ LIMIT	**Full support**	[ACTIVITY] [NCI 2.0]	0x02	Derived from CON_DEVICE_LIMIT in NCI 1.0
PF_DEVICES_ LIMIT	**Full support**	[ACTIVITY] [NCI 2.0]	0x02	Derived from CON_DEVICE_LIMIT in NCI 1.0
PV_DEVICES_ LIMIT	**Full support**	[ACTIVITY] [NCI 2.0]	0x02	Derived from CON_DEVICE_LIMIT in NCI 1.0
CON_ DISCOVERY_ PARAM	**Full support**	[ACTIVITY] [NCI 2.0]	0x01	
PA_BAIL_OUT	**No Support**	[ACTIVITY]	-	Bail Out is always activated in Poll/NFC-A
PB_AFI	**Full support**	[DIGITAL]	0x00	
PB_BAIL_OUT	**No Support**	[ACTIVITY]	-	Bail Out is always activated in Poll/NFC-B
PB_ATTRIB_ PARAM1	**Full support**	[DIGITAL]	0x00	
PB_SENSB_ REQ_ PARAM	**No Support**	[DIGITAL]	-	No support of advanced features in NFC-B, no support of the extended SENSB_RES.
PF_BIT_RATE	**Full support**	[DIGITAL]	0x01 (212kb/s)	

Table 14. Compliance to [NCI] configuration parameters ***...continued***

Config parameters	Status on PN7160	Coming from	Default value	Remark
PF_RC_CODE	**Full support**	[DIGITAL] [NCI1.0]	0x00	
PF_BAIL_OUT	**No Support**	[ACTIVITY] [NCI 2.0]	-	Bail Out is always activated in Poll/NFC-F
PB_H_INFO [NCI1.0] PI_B_H_INFO [NCI2.0]	**Full support**	[DIGITAL]	empty	Renamed to PI_B_H_INFO in NCI2.0
PI_BIT_RATE	**Full support**	[DIGITAL]	0x00 (106kb/s)	
PN_NFC_DEP_ SPEED [NCI1.0] PN_NFC_DEP_ PSL [NCI2.0]	**Full support**	[DIGITAL]	0x00 (106kb/s)	Renamed to PN_NFC_DEP_PSL in NCI2.0
PN_ATR_REQ_ GEN_BYTES	**Full support**	[DIGITAL]	empty	
PN_ATR_REQ_ CONFIG	**Full support**	[DIGITAL]	0x30	
LA_BIT_FRAME_ SDD	**Full support**	[DIGITAL]	0x01	
LA_PLATFORM_ CONFIG	**Full support**	[DIGITAL]	0x00	
LA_SEL_INFO	**Full support**	[DIGITAL]	0x00	
LA_NFCID1	**Full support**	[DIGITAL]	0x00000000	
LB_SENSB_INFO	**Full support**	[DIGITAL]	0x81	
LB_NFCID0	**Full support**	[DIGITAL]	0x00000000	
LB_ APPLICATION_ DATA	**Full support**	[DIGITAL]	Empty	
LB_SFGI	**Full support**	[DIGITAL]	0x00	Available for both [NCI 1.0] and [NCI 2.0]
LB_ADC_FO [NCI 1.0] LB_FWI_ADC_FO [NCI 2.0]	**Full support**	[DIGITAL]	0x05	LI_FWI from [NCI 1.0] has been split between Listen A and Listen B. Then LB_ADC_FO I n NCI 1.0 becomes LB_FWI_ADC_FO in [NCI 2.0]
LB_BIT_RATE	**Full support**	[NCI] [NCI 2.0]	0x00	
LF_T3T_ IDENTIFIERS_ 1..4	**Full support**	[DIGITAL]	0xFFFF02FE00 0000000000FF FFFFFFFFFFF FFF	By default, LF_T3T_PMM_DEFAULT is used (backward compatibility to NCI 1.0)
LF_T3T_ IDENTIFIERS_ 5..16	**No Support**	[DIGITAL]	-	

Table 14. Compliance to [NCI] configuration parameters ***...continued***

Config parameters	Status on PN7160	Coming from	Default value	Remark
LF_PROTOCOL_TYPE	**Full support**	[DIGITAL]	0x02	
LF_T3T_PMM_DEFAULT	**Full support**	[DIGITAL] [NCI 1.0]	0xFFFFFFFFFFFFFFFF	
LF_T3T_MAX	**Full support**	[NCI]	0x04	
LF_T3T_FLAGS	**Full support**	[NCI]	0x0000	
LF_T3T_RD_ALLOWED	**Full support**	[NCI] [NCI 2.0]	0x00	
LF_PROTOCOL_TYPE	**Full support**	[NCI] [NCI 2.0]	0x00	
LF_CON_BITR_F	**Partial Support**	[DIGITAL]	0x06	Always both 212 and 424 kb/s
LI_FWI	**Full support**	[DIGITAL] [NCI 1.0]	0x04	LI_FWI is supported in NCI1.0
LI_A_RATS_TB1	**Partial Support**	[DIGITAL] [NCI 2.0]	0x04	LI_A_RATS_TB1 implementation for NCI2.0 does only support FWI
LA_HIST_BY [NCI 1.0] LI_A_HIST_BY [NCI 2.0]	**Full support**	[DIGITAL]	empty	Renamed to LI_A_HIST_BY in NCI2.0
LB_H_INFO_RESP	**No Support**	[DIGITAL]	-	Consequence: the "Higher Layer Response" field in the ATTRIB Response is left empty
LI_BIT_RATE	**Full support**	[DIGITAL]	0x00 (106kb/s)	
LI_A_RATS_TC1	**Partial Support**	[NCI] [NCI 2.0]		Also available when working in NCI1.0 mode
LN_WT	**Full support**	[DIGITAL]	0x08	
LN_ATR_RES_GEN_BYTES	**Full support**	[DIGITAL]	Empty	
LN_ATR_RES_CONFIG	**Full support**	[DIGITAL]	0x30	
PACM_BIT_RATE	**Full support**	[NCI] [NCI 2.0]	0x01	Activation bit rate in P2P initiator active
RF_FIELD_INFO	**Full support**	[NCI]	0x00	
RF_NFCEE_ACTION	**Full support**	[NCI]	0x01	
NFCDEP_OP	**Full support**	[NCI]	0x0E	Minor deviation compared to NCI standard that specifies 0x00 for [NCI 1.0] and 0x1F for [NCI 2.0] by default
LLCP_VERSION	**Full support**	[LLCP] [NCI 2.0]	0x11	
NFCC_CONFIG_CONTROL	**Full support**	[NCI] [NCI 2.0]	0x00	In [NCI 1.0], similar proprietary API exists with proprietary configuration parameter 0xA09B

7.3.6 Compliance to [NCI] data messages

PN7160 is fully compliant to the [NCI] data messages.

7.4 Extensions added to [NCI] to allow full control of the PN7160

The [PN7160-NCI] Extensions section gives a quick overview of the numerous extensions required to [NCI] to give full access to all the features available in the PN7160.

7.4.1 [PN7160-NCI] extensions to [NCI] RF Protocols

PN7160 supports more protocols than handled today by [NCI].

It is required to extend the [NCI_Table5] defined in [NCI] (see chapter →3) such that these protocols can be configured in various commands/notifications:

Table 15. Proprietary RF protocols

Chapter	Value	Description
→Section 9.4	0x06	PROTOCOL_15693 (only for [PN7160-NCI1.0])
→Section 9.1	0x80	PROTOCOL_MIFARE_CLASSIC
→Section 9.5	0x81	PROTOCOL_KOVIO
	0x82-0x9F	Reserved for Proprietary protocols
	0xA0-0xFD	Reserved for Proprietary protocols

7.4.2 [PN7160-NCI] extensions to [NCI] Bit Rates in ISO15693 and NFC-F

PN7160 supports the Poll Mode for technology ISO15693. Unfortunately, [NCI1.0] does not define an appropriate bit rate (26kb/s) the NFCC has to report to the DH in the RF_INTF_ACTIVATED_NTF. NXP has defined a proprietary value for this bit rate.

PN7160 offers the possibility to poll for NFC-F @ 212 kb/s and NFC-F @ 424 kb/s, unfortunately, [NCI] only allows configuring one of these 2 bit rates, but not both in the same discovery sequence. The [NCI] parameter used to configure the bit rate in NFC-F is PF_BIT_RATE. By setting PF_BIT_RATE to the value of 0x80 “NFC_BIT_RATE_212 AND NFC_BIT_RATE_424”, polling is done for both 212 and 424k in the same discovery sequence.

Table 16. Proprietary Bit rates

Chapter	Value	Description
→Section 9.4	0x80	NFC_BIT_RATE_26 (only for [PN7160-NCI1.0])
	0x80	NFC_BIT_RATE_212 AND NFC_BIT_RATE_424

7.4.3 [PN7160-NCI] [NCI-2.0] Bit Rates in NFC-V

In NCI2.0, PN7160 supports the Poll Mode for technology [NCI2.0] defines an appropriate bit rate (26kb/s) the NFCC has to report to the DH in the RF_INTF_ACTIVATED_NTF.

Table 17. T5T Bit rates

Chapter	Value	Description
→Section 9.4	0x20	NFC_BIT_RATE_26

/!\	For NCI2.0, ISO15693 is renamed T5T and bit rate is the one mentioned above.

7.4.4 [PN7160-NCI] extensions to [NCI] RF Interfaces

PN7160 offers some features which are not accessible using the currently defined RF interfaces in [NCI]. So the [NCI_Table6] (see chapter →Section 3) needs to be extended with some proprietary RF interfaces, as described in the table below:

Table 18. RF Interfaces extension

Chapter	New RF Interface	Value	Brief description
→Section 9.1.3	TAG-CMD	0x80	This new interface adds a header to the data payload, in order to encode commands such as: - T2T/MFUL sector select command - MIFARE Classic Authenticate command
		0x81-0xFE	Reserved for proprietary RF Interfaces

These three proprietary RF Interfaces are reported inside the CORE_INIT_RSP.

7.4.5 [PN7160-NCI] extensions to [NCI] Control messages

This section contains all the additional commands/notifications in [PN7160-NCI].

Some commands mentioned with [PN7160-NCI1.0] are NCI1.0 specific extensions.

Table 19. PN7160-NCI additional commands/notifications

Chapter	PN7160-NCI Control message	Brief description	Support
→Section 8.4.1	NCI_PROPRIETARY_ ACT_ CMD/RSP	Command might be used in the future by the DH to activate the proprietary functions inside the NFCC	**Full Support**
→Section 9.3.3	RF_ PRES-CHECK_ CMD/RSP/ NTF	Command used to check if a T4T or an ISO-DEP tag is still in the field [PN7160-NCI1.0]	**Full Support**
→Section 13.3	RF_GET_TRANSITION_ CMD/ RSP	To read out an RF register setting for a given RF Transition	**Full Support**
→Section 12.8	RF_PLL_UNLOCKED_NTF	Notification used to indicate that the PLL has been started but could not have been locked. This might be due to a missing or unstable input clock.	**Full Support**
→Section 12.8	RF_TXLDO_ERROR_NTF	Notification used to indicate that TxLdo (RF Transmitter) could not start. This is usually due to a missing or bad power supply on VUP/TVDD or a bad clock/power configuration	**Full Support**
→Section 12.7	SCREEN_STATE_ CMD/RSP	Command used to notify NFCC about a screen state change. This may be used to allow different configurations in listen mode. [PN7160-NCI1.0]	**Full Support**
→Section 12.6.1	CORE_SET_POWER_ MODE_ CMD/RSP	Command allowing the DH to configure the power mode (standby or idle mode).	**Full Support**

Table 19. PN7160-NCI additional commands/notifications*...continued*

Chapter	PN7160-NCI Control message	Brief description	Support
→Section 9.3.5	RF_WTX_NTF	Notification sent by NFCC to inform DH about a WTX exchanged in 14443-4 PCD	**Full Support**
→Section 12.4.3	RF_LPCD_TRACE_NTF	Notification to collect the measurements performed by the Tag Detector	**Full Support**
→Section 14.2	TEST_ANTENNA_ CMD/RSP	Command allowing the DH to check the presence of the antenna components on the PCB.	**Full Support**
→Section 14.3	TEST_PRBS_CMD/RSP	Command allowing the DH to send data over RF at different baud rates in order to verify the contactless part without any interaction with the NCI RF Discovery.	**Full Support**
→Section 14.4	TEST_GET_REGISTER_CMD/ RSP	Command to receive the Value of the AGC_VALUE_ REGISTER used to configure the dynamic LMA lookup tables	**Full Support**

[NCI] defines some rules which constraint the use of the control messages. That means that depending on the state the NCI RF State Machine is in, depending on the RF Interface used, depending on some parameters, the control messages are valid or incorrect, and sometimes they trigger state transitions.

NXP has extended these rules for the [PN7160-NCI1.0] and [PN7160-NCI2.0] extensions.

The following tables give the full picture of these rules:

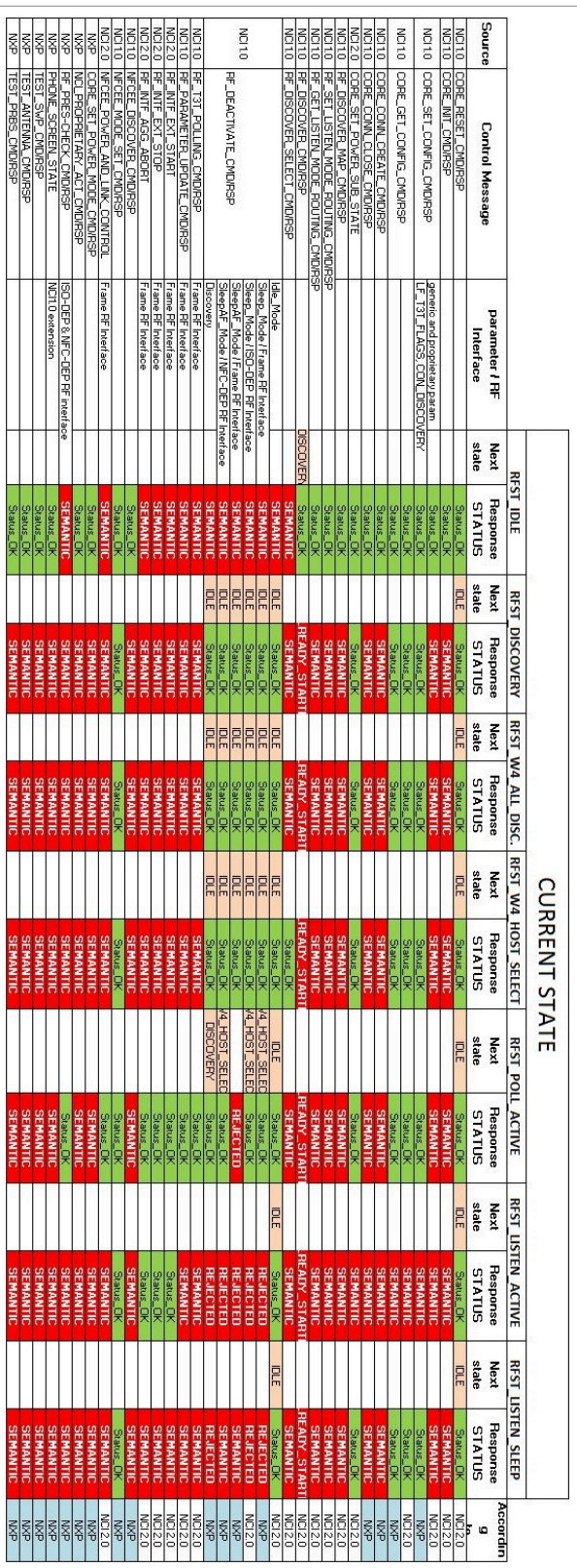

Figure 26. CMDs/RSPs versus the current state of the NCI RF State Machine

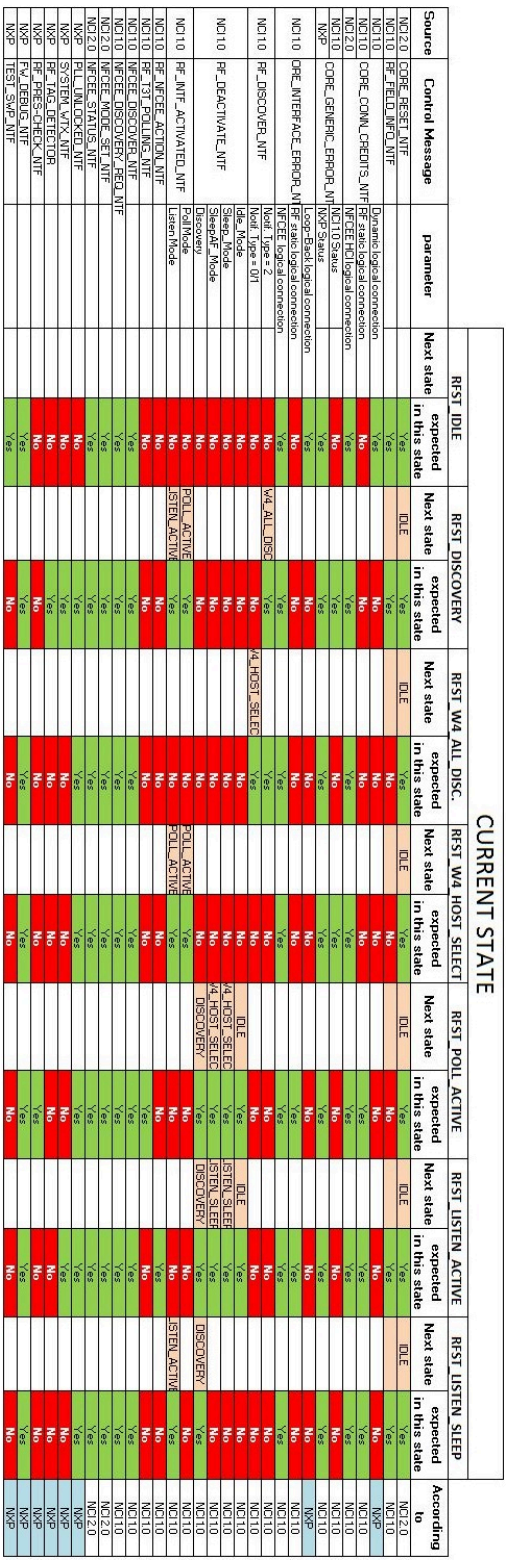

Figure 27. NTFs versus the current state of the NCI RF State Machine

PN7160 defines additional states to the RF state machine defined in [NCI_Chap2], to ensure a correct implementation of the “atomic behavior” of the pair of commands made

by CORE_RESET_CMD and CORE_INIT_CMD and also to correctly handle wrong RF protocol to RF interface mapping through the RF_DISCOVER_MAP_CMD. The drawing below illustrates these additional states, linked to the [NCI]-defined RFST_IDLE:

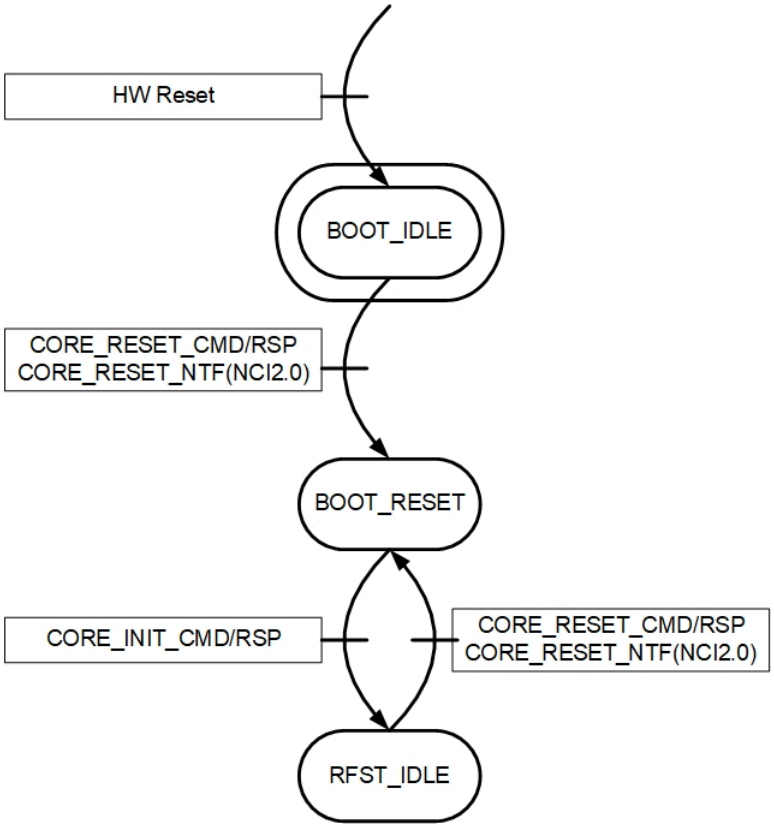

Figure 28. States added to the [NCI] State Machine.

7.4.6 [PN7160-NCI] extensions to [NCI] Configuration parameters

[NCI] lists a number of parameters, which are necessary to set up the RF discovery. But the PN7160 requires a lot more parameters, for instance to configure some RF protocols which are not supported by [NCI], to configure the power and clock management …

Here is a list of sets of parameters, sorted out by features to configure:

Table 20. Overview of additional Configuration parameters

Chapter	Feature to configure	Comment
→Section 13.1	System	Parameters allowing the DH to configure the System: Clock management, IRQ and CLOCKREQ pins management, MIFARE Classic Keys handling…
→Section 13.2	RF Discovery	Parameters allowing the DH to configure the Discovery activity (Tag Detector, Discovery profile between: NFC Forum, EMVCo…).
→Section 13.3	Contactless Front-End	Parameters allowing the DH to configure all internal HW settings in the ContactLess InterFace (CLIF).

Please refer to the chapters listed in the first column to have all the detailed information on the parameters.

7.4.7 [PN7160-NCI] extensions to [NCI] proprietary parameters space

[NCI] defines a parameter space with a size of 255 parameters, in which around 96 tags are allocated for proprietary parameters:

Table 21. Parameter space

Parameters space sub-sections	Tag
Assigned and reserved for NCI 1.0/2.0	0x00-0x9F
Reserved for Proprietary Use	0xA0-0xFE
RFU (Reserved for Extension)	0xFF

Regarding the PN7160 needs, this reserved area is not sufficient. To extend this space, the solution chosen is to define a space of Tags coded on 16 bits, instead of 8 bits. These extended Tags will always start by the value 0xA0, which is the first value available in the Proprietary range. This allows adding 256 new parameters.

Remark: If this is not sufficient in the future, we might use 16-bit tag values starting by 0xA1, 0xA2 …

Table 22. Extended TLV for proprietary parameters

Payload	Field	Length	Description
m+3 bytes	Tag = 0xA0XX	2 bytes	Extended tag identifier
	Len	1 byte	The length of Val (m)
	Val	m bytes	Value of the configuration parameter

This is illustrated by the following picture:

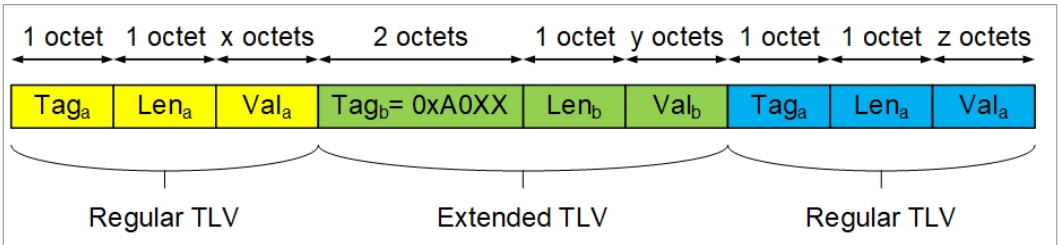

Figure 29. Regular and Extended TLVs comparison

7.4.8 [PN7160-NCI] extensions to [NCI] Status Codes

[NCI] defines a set of standard Status Codes in [NCI_Table1] (see chapter →Section 3).

NXP has extended this set of status codes with the following values:

Table 23. Proprietary Status Codes

Status code	Description	Used in
0xA3	STATUS_LPCD_FAKE_DETECTION	CORE_GENERIC_ERROR_NTF
0xE1	STATUS_BOOT_TRIM_CORRUPTED	CORE_GENERIC_ERROR_NTF
0xE4	STATUS_EMVCO_PCD_COLLISION	CORE_GENERIC_ERROR_NTF

7.4.9 [PN7160-NCI] extensions to [NCI] Reason Code in CORE_RESET_NTF

[NCI] defines a set of standard Reason Codes in the CORE_RESET_NTF. Please refer to [NCI_Table9] (see chapter →Section 3).

NXP has extended this set of reason codes with the following value:

Table 24. Proprietary Reason Codes in CORE_RESET_NTF

Reason code	Description
0xA0	An assert has triggered PN7160 reset/reboot
0xA1	An over temperature has triggered the reset of PN7160
0xA3	Arm subsystem reset / Watchdog reset has occurred

/!\	When the proprietary reason code is used, the CORE_RESET_NTF is out of [NCI] compliance.

Indeed, PN7160 appends one parameter at the end of the CORE_RESET_NTF, to provide some information for debug purposes. The CORE_RESET_NTF format is then:

7.4.9.1 [PN7160-NCI] extensions CORE_RESET_NTF with Reason Code 0xA0

Table 25. CORE_RESET_NTF when reason code = 0xA0 is used

Payload Field(s)	Length	Description	Default
Reason Code	1 byte	0xA0: NXP proprietary	0xA0
Configuration Status	1 byte	See [NCI]	0x00
Program Counter	4 bytes	Program counter for assertion (field present only when reason code is 0xA0)	

7.4.9.2 [PN7160-NCI] extensions CORE_RESET_NTF with Reason Code 0xA1

The sequence followed by PN7160 when an over temperature is detected is the following:

- PN7160 forces pins CLK_REQ to logical '0'
- PN7160 waits then until the chip temperature comes down to an internal threshold (the power consumption in this mode is around 100µA, since the temperature monitoring circuit is still alive)
- When the internal temperature is low enough, PN7160 reboots, disabling the RF discovery if it was previously enabled

PN7160 sends then a CORE_RESET_NTF (0xA1) to inform the DH that an over temperature event occurred.

7.4.9.3 [PN7160-NCI] extensions CORE_RESET_NTF with Reason Code 0xA3

This notification is not expected in normal operating mode but may occur in rare cases if a connection is broken on the platform or a bad configuration has been entered. In such case, local NXP support may be needed and some specific NCI CORE_GET_CONFIG_CMD with proprietary parameters ‘A01B’ and ‘A027’ may be sent by DH in order to get some additional information for further debugging by R&D.

7.4.10 [PN7160-NCI] extensions to [NCI] RF Technology & Mode

PN7160 supports more RF Technology & Mode parameters than handled today by [NCI].

It is required to extend the [NCI_Table3] defined in [NCI] (see chapter →Section 3) such that these RF Technology & Mode parameters can be used in RF_DISCOVER_CMD:

Table 26. Proprietary RF Technology & Mode parameters

Chapter	Value	Description
→Section 11.1.2	0x03	NFC_A_ACTIVE_POLL_MODE ([PN7160-NCI1.0])
→Section 11.1.2	0x05	NFC_F_ACTIVE_POLL_MODE ([PN7160-NCI1.0])
→Section 9.4	0x06	NFC_15693_POLL_MODE ([PN7160-NCI1.0])
→Section 9.5	0x70	NFC_A_KOVIO_POLL_MODE ([PN7160-NCI)
	0x71-0x7F	Reserved for Proprietary Technologies in Poll Mode
→Section 11.1.2	0x83	NFC_A_ACTIVE_LISTEN_MODE ([PN7160-NCI1.0])
→Section 11.1.2	0x85	NFC_F_ACTIVE_LISTEN _MODE ([PN7160-NCI1.0])

/!\	PN7160 supports full RF Technology & Mode parameters as defined in [NCI2.0] when initialized in NCI2.0 mode.

7.4.11 [PN7160-NCI1.0] extensions to [NCI1.0] Power Modes

In order to configure the Listen Mode Routing table in a specific way when the screen of the Device is turned Off, PN7160 needs a specific Power mode which does not exist in [NCI1.0]. So the Value Field of Power States as defined in [NCI_Table11] is extended:

Table 27. Modified Value Field of power states:

Chapter	Value								Description
→Section 12.7	b7	b6	b5	b4	b3	b2	b1	b0	
	x								Screen Off ([PN7160-NCI1.0])
		x							Screen Locked ([PN7160-NCI1.0])
			0	0	0				RFU
						0			Battery off
							x		Switched off
								x	Switched on

A new command is also added, to allow the DH inform the NFCC that it has entered one of these 2 additional power modes: see →Section 12.7.

7.5 Tag deactivation in RF frame interface

In NCI2.0 mode, when using the NCI command NCI_DEACTIVATE_CMD (Sleep) in Frame RF Interface, the NFCC will only change the internal state machine. It is the responsibility of DH to properly handle the tag deactivation by sending the appropriate RF commands to the tag. In NCI1.0 mode, then NFCC will handle the deactivation tag itself and also update the NCI state machine.

7.6 NDEF emulation

The NFCC provides an NDEF T4T emulation feature compliant with the Type 4 Tag Operation Technical Specification. As such, the NFCC embedded an NDEF NFCEE.

The NDEF T4T emulation supports the following Command-APDU:

- Select: 0xA4 (Selection of applications, or files)
- ReadBinary: 0xB0
- UpdateBinary: 0xD6

The maximum NDEF message length is 136 bytes.

7.6.1 NDEF NFCEE discovery

The feature is disabled by default, the DH shall send a CORE_SET_CONFIG with parameter 0xA095 in RF_IDLE_STATE in order to activate the NDEF T4T tag emulation feature.

The NDEF T4T tag emulation is seen by the DH as an NDEF NFCC using the NFCEE_DISCOVER_CMD.

Table 28. NFCEE_DISCOVER_NTF for the NDEF tag emulation

Payload Field(s)	Length	Value/Description
NFCEE ID	1 byte	0x10 (NDEF NFCEE)
NFCEE Status	1 byte	0x00 if NDEF NFCEE is enabled, 0x01 if disabled
Number of Protocol Information Entries	1 byte	1
Supported NFCEE Protocols [0..n]	1 byte	0x00 (ISO7816-4 APDU protocol)
Number of NFCEE Information TLVs	1 byte	1
NFCEE Information	8 bytes	• NDEF type: 0x04 • Length: 0x06 • NDEF: 0x00000088 • Supported power state: 0xC3 • Persistent characteristics: 0x01

Once detected by the DH, the NDEF NFCEE does not require any negotiation, its capabilities are fixed. Then, after the DH has sent a NFCEE_MODE_SET command to enable the NDEF NFCEE, the NFCC will immediately send a NFCEE_DISCOVERY_REQ_NTF:

Table 29. NFCEE_DISCOVER_REQ_NTF for the NDEF tag emulation

Payload Field(s)	Length	Value/Description
Number of entries	1 byte	0x02
First information entry	5 bytes	• Information Type: 0x00 if enabled, 0x01 if disabled • Length: 0x03 • NFCEE ID: 0x10 • RF Technology and Mode: 0x80 (NFC_A_PASSIVE_ LISTEN_MODE) • RF Protocol: 0x04 (PROTOCOL_ISO_DEP)

Table 29. NFCEE_DISCOVER_REQ_NTF for the NDEF tag emulation...*continued*

Payload Field(s)	Length	Value/Description
Second information entry	5 bytes	• Information Type: 0x00 if enabled, 0x01 if disabled • Length: 0x03 • NFCEE ID: 0x10 • RF Technology and Mode: 0x81 (NFC_B_PASSIVE_ LISTEN_MODE) • RF Protocol: 0x04 (PROTOCOL_ISO_DEP)

7.6.2 NDEF NFCEE access from DH

After the NDEF NFCEE has been discovered and enabled by the DH, the DH can activate a logical connection with the NDEF NFCEE:

Table 30. CORE_CONN_CREATE_CMD for NDEF tag emulation

Payload Field(s)	Length	Value/Description
Destination type	1 byte	0x03 (NFCEE)
Number of parameters	1 byte	0x01
Destination parameters	4 bytes	• Destination Type: 0x01 • Length: 0x02 • NFCEE ID: 0x10 • NFCEE Interface Protocol: 0x00 (ISO7816-4 APDU protocol)

Then NFCC answers with a CORE_CONN_CREATE_RSP with a status OK.

Using the logical channel #5 and according to the Type 4 Tag Operation Technical Specification, the DH can exchange NCI data packets with the NDEF NFCEE in order to select the NDEF AID, read the Capability Container, read and write the content of the NDEF message.

7.6.3 NDEF NFCEE access from RF End Point

For read access from the RF End Point, the routing table shall contain an AID-based routing entry with the NDEF AID:

Table 31. RF_SET_LISTEN_MODE_ROUTING_CMD for NDEF tag emulation

Payload Field(s)	Length	Value/Description
More	1 byte	Last message: 0x00
Number of routing entries	1 byte	0x01
Routing Entry	2 bytes	• Qualifier Type: 0x02 (AID-based routing entry) • Length: 0x9 • Route: 0x10 (NDEF NFCEE) • Power State: see →Power Modes • AID: [0xD2 0x76 0x00 0x00 0x85 0x01 0x01]

Then NFCC answers with a RF_SET_LISTEN_MODE_ROUTING_RSP with a status OK.

Here are the commands and configuration parameters to prepare the NDEF card emulation in Listen Mode for ISO-DEP through the ISO-DEP Interface with technology NFC-A or NFC-B:

Table 32. Configuration sequence for NDEF CE of NFC-A / ISO-DEP through the ISO-DEP interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Listen
	RF Interface	ISO-DEP
CORE_SET_CONFIG_CMD	LA_BIT_FRAME_SDD	
	LA_PLATFORM_CONFIG	
	LA_SEL_INFO	
	LA_NFCID1	
	LB_SENSB_INFO	
	LA_NFCID1	
	LB_NFCID0	
	LB_APPLICATION_DATA	
	LB_SFGI	
	LB_ADC_FO	
	LI_FWI	
	LI_BIT_RATE	
	RF_NFCEE_ACTION	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A/B_PASSIVE_LISTEN_MODE

7.6.4 T4T operation: detection of the NDEF message from the DH

The DH exchanges APDU with the NDEF NFCEE through NCI data packet with logical channel #5.

The NDEF message detection from the DH consists of the following sequence:

Table 33. Detection of the NDEF message from the DH

C-APDU	Description
NDEF AID select	Select the NDEF application
CC file select	Select the capability container file
CC file length read	Get the capability container length
CC file read	Get the capability container value

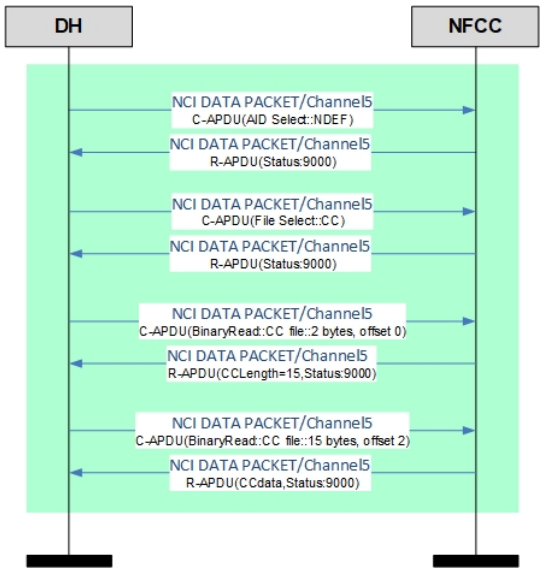

Figure 30. Detection of the NDEF message from the DH

NDEF tag emulation capabilities are hardcoded, thus cannot be updated, but can be retrieved from the CC file using the binary read C-APDU.

7.6.5 T4T operation: read/write data from the NDEF message from the DH

The NDEF message access for a binary read or a binary write operation can only be done after detection of the NDEF AID and binary read of the capability container information.

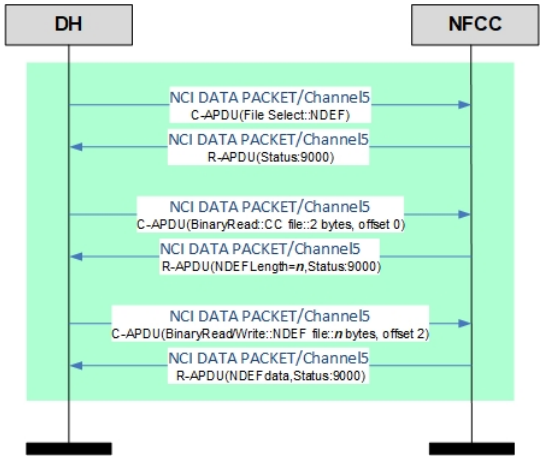

Figure 31. Read/Write raw data from the NDEF message with the DH

7.6.6 T4T operation: detection of the NDEF message from the RF End Point

The RF End Point reader exchanges APDU with the NDEF NFCEE through the ISO-DEP RF interface and PROTOCOL_ISO_DEP protocol.

The NDEF message detection consists on the following sequence:

Table 34. Detection of the NDEF message from an RF Reader

C-APDU	Description
NDEF AID select	Select the NDEF application
CC file select	Select the capability container file
CC file length read	Get the capability container length
CC file read	Get the capability container value

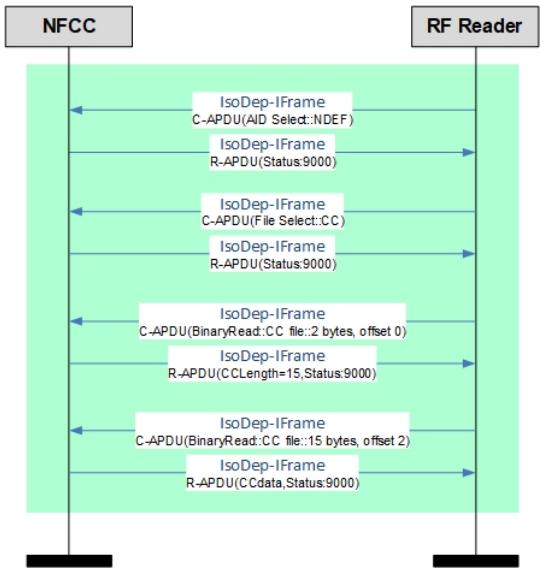

Figure 32. Detection of the NDEF message from an RF Reader

NDEF tag emulation capabilities are hardcoded, thus cannot be updated, but can be retrieved from the CC file using the binary read C-APDU.

7.6.7 T4T operation: read/write data from the NDEF message from the RF End Point

The NDEF message access for a binary read operation can only be done after detection of the NDEF AID and binary read of the capability container information.

/!\	NDEF message cannot be written from the RF End Point, it can only be read.

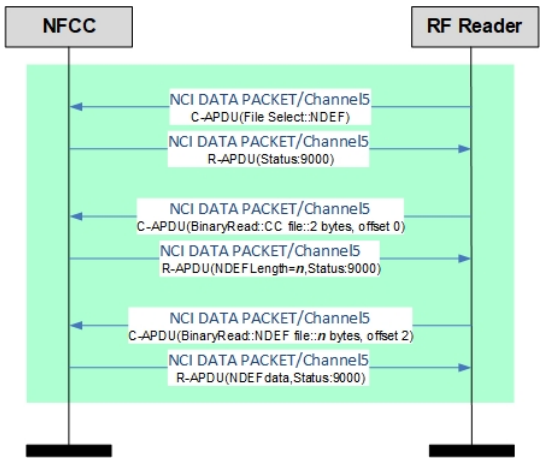

Figure 33. Read of the NDEF message with a RF Reader

8 Initialization and operation configuration

8.1 Reset / Initialization

[NCI] defines a Reset/Init sequence, which is based on two different commands:

- CORE_RESET_CMD
- CORE_INIT_CMD

These two commands have to be called by the DH in an “atomic” way: there cannot be any other command in-between and the PN7160 operation cannot start any operation (Reader/Writer, Card Emulation, P2P, Combined modes etc…) if it does not first receive these 2 commands.

[NCI] defines 2 modes for the Reset command: Keep Configuration and Reset Configuration. Here is the detail of the difference between the 2 reset modes:

Table 35. Comparison of the 2 Reset Modes

Features	Reset Configuration	Keep Configuration
MCU reboot	Yes	Yes
Listen Mode Routing table	Lost	Kept
NCI Configuration parameters	Back to default	Kept
Proprietary Configuration parameters	Kept	Kept
Interface Mapping Table	Lost	Kept
Discovery activity	Lost	Lost
Dynamic connections	Lost	Kept
NFCEE status (enabled/disabled)	Kept	Kept

/!\	PN7160 may delay the CORE_RESET_RSP

If the DH sends a CORE_RESET_CMD while PN7160 has already indicated that it has some data available to be read by the DH (IRQ pin activated), the DH has first to read the data available from PN7160 before it can get the CORE_RESET_RSP. The reason is that the NCI output buffer in PN7160 needs to be flushed before PN7160 can apply a Reset and then send the CORE_RESET_RSP.

8.2 Switch between NCI1.0 and NCI2.0

PN7160 supports NCI2.0 with backward compatibility to NCI1.0.

As such it always sends CORE_RESET_NTF, after CORE_RESET_RSP answer to CORE_RESET_CMD command from the DH, indicating NCI supported version.

Table 36. NCI2.0 CORE_RESET_NTF triggered from CORE_RESET_CMD

Payload Field(s)	Length	Value/Description
Reset Trigger	1 byte	0x02 (CORE_RESET_CMD was received)
Configuration Status	1 byte	0x00 if configuration has been kept, 0x01 if configuration has been reset

Table 36. NCI2.0 CORE_RESET_NTF triggered from CORE_RESET_CMD...*continued*

Payload Field(s)	Length	Value/Description
NCI Version	1 byte	0x20 (NCI Version 2.0)
Manufacturer ID	1 byte	0x04 (NXP)
Manufacturer Specific Information Length	1 byte	0x04
Manufacturer Specific Information	4 bytes	See →Manufacturer Specific Information

The NCI version to be used is then determined by the CORE_INIT_CMD sent by the DH that provides 2 extra bytes in NCI2.0.

Table 37. NCI2.0 CORE_INIT_CMD

GID	OID	Numbers of parameter(s)	Description
0b	0x01	1	Feature Enable (2 bytes)

When the PN7160 receives NCI2.0 CORE_INIT_CMD from the DH, it switches current NCI version execution mode to NCI2.0, while when receiving NCI1.0 CORE_INIT_CMD it switches to NCI1.0.

/!\	When a DH is operating in NCI1.0, it might expect a CORE_RESET_NTF notification from the NFCC depending on the current NCI version execution mode of the NFCC.

8.3 Manufacturer Specific Information in [NCI] CORE_INIT_RSP / CORE_RESET_NTF

The NCI1.0 CORE_INIT_RSP message and the NCI2.0 CORE_RESET_NTF notification contains a field “Manufacturer Specific Information” with 4 bytes.

Here is the meaning of these 4 Bytes and the conditions when they are incremented / updated:

Table 38. Manufacturer Specific Information in NCI1.0 CORE_INIT_RSP and NCI2.0 CORE_RESET_NTF

Byte	Meaning	Condition to increment
0	Hardware Version number	New silicon
1	ROM Code Version number	New ROM Code
2	FLASH Major version	New Firmware, adding features
3	FLASH Minor version	New Firmware, solving bugs on existing features.

8.4 Whole sequence to prepare the PN7160 operation

After the Reset/Init sequence is passed, PN7160 requires several other steps before it is ready to start operating as Reader/Writer, Card Emulator …

The simplest case is when the DH issues CORE_RESET_CMD command with Reset Type = Keep Configuration.

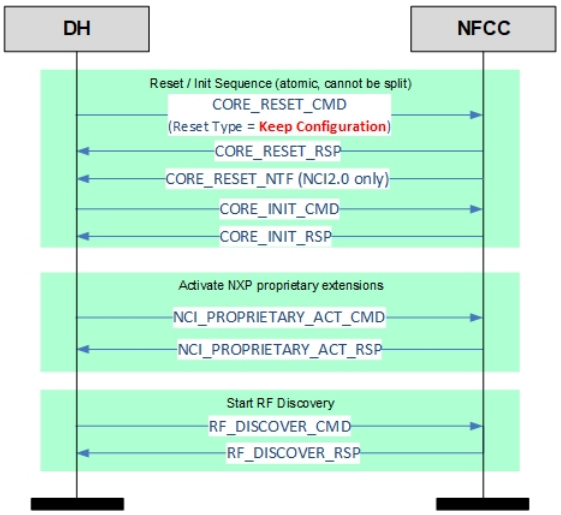

Figure 34. Initialization sequence to prepare PN7160 operation (Keep Configuration)

In case of CORE_RESET_CMD command issued by the DH with Reset Type = Reset Configuration, the entire configuration is lost so PN7160 needs to be reconfigured and various optional steps might be needed depending on the targeted use case.

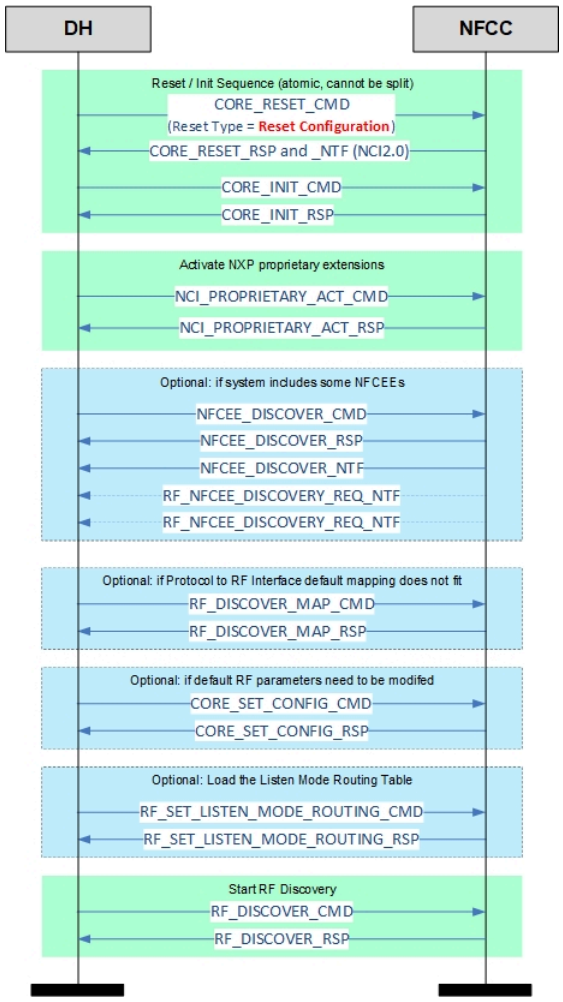

Figure 35. Full initialization sequence to prepare the PN7160 operation (Reset configuration)

8.4.1 Proprietary command to enable proprietary extensions

It is visible on the previous flowchart that NXP has introduced a proprietary command sent by the DH to enable the proprietary extensions to [NCI] offered by the PN7160. So, when PN7160 receives NCI_PROPRIETARY_ACT_CMD command, it knows that the DH is aware of the proprietary extensions and may therefore send proprietary notifications (see the list in Table 18).

Table 39. NCI_PROPRIETARY_ACT_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x02	0	DH informs the PN7160 that it knows the proprietary extensions

Table 40. NCI_PROPRIETARY_ACT_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x02	2	PN7160 indicates that it understood the command

Table 41. NCI_PROPRIETARY_ACT_RSP parameters

Payload Field(s)	Length	Value/Description
Status	1 byte	One of the following Status codes, as defined in [NCI_Table1] 0x00: STATUS_OK 0x03: STATUS_FAILED Others: RFU
FW_Build_Number	4 bytes	NXP internal firmware build number

8.4.2 Configuration template

In order to help issuing the right configuration sequence for a given mode of operation, the present document details typical configuration sequence, based on the following template:

Table 42. Template for a typical configuration sequence

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	…
	Mode	…
	RF Interface	…
CORE_SET_CONFIG_CMD	Depends on Technology & Mode	…
RF_SET_LISTEN_MODE_ ROUTING_CMD *(for Listen Mode only)*	Technology-based routing	…
	Protocol-based routing	…
	AID-Based routing	…
RF_DISCOVER_CMD	RF Technology & Mode	…

8.5 PLL input Clock Management

The PN7160 support 2 clock sources scheme:

- a 27.12 MHz quartz ("XTAL MODE")
- or a clean clock signal available on the platform on which PN7160 is connected ("PLL MODE"). A PLL inside PN7160 converts this input clock signal into an internal 27.12 MHz used to generate the RF carrier. The input clock frequency has to be one of the predefined set of input frequencies: 13 MHz, 19.2 MHz, 24 MHz, 26 MHz, 32 MHz, 38.4 MHz or 48 MHz.

The DH has to configure the parameter CLOCK_SEL_CFG (see chapter →Section 13.1) to configure what is the clock source as used in the current application. When "PLL MODE" is set as clock source, this parameter also indicates which clock frequency is used as an input to the PLL.

If the clock source is the 'PLL', then the DH shall also configure the appropriate PLL_SETTING and DPLL_SETTING (see chapter →Section 13.1) depending on the input clock frequency.

In order to optimize system power consumption, it may be required to switch OFF the PLL input clock when the PN7160 does not need it (when PN7160 has to generate the 13.56 MHz RF carrier or when downloading a new firmware). A dedicated pin (CLKREQ) is used to inform the DH or a clock generating chip that the PN7160 requires to get the PLL input clock, such that it can generate the 13.56 MHz RF carrier.

PN7160 assumes that the PLL input clock is ON and stable after a programmable time-out, which is configured thanks to the parameter CLOCK_TO_CFG (see chapter →Section 13.1).

8.6 TVDD configurations

PN7160 supports 2 different TVDD configurations, called CFG1 and CFG2 (with or without external DC-DC).

See PMU_CFG in →Section 13.1 for the configuration options.

8.6.1 CFG1: VDD(UP) connected to VBAT

In CFG 1, VDD(UP) and VBAT are connected to an external supply delivering between 2.8 V and 5.5 V.

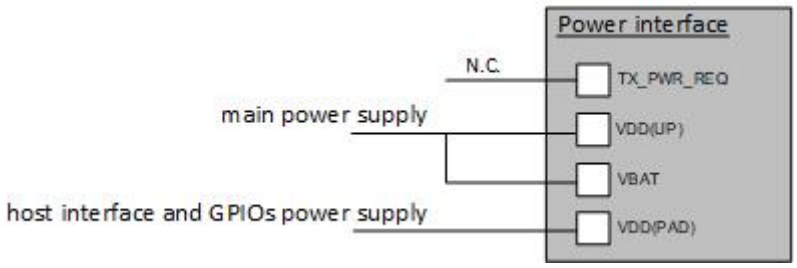

Figure 36. CFG1 - VDD(UP) connected to VBAT

8.6.2 CFG2: VDD(UP) connected to external power supply

In CFG2, the VDD(UP) pin is connected to an external power supply. The internal TXLDO is used to generate configurable TVDD.

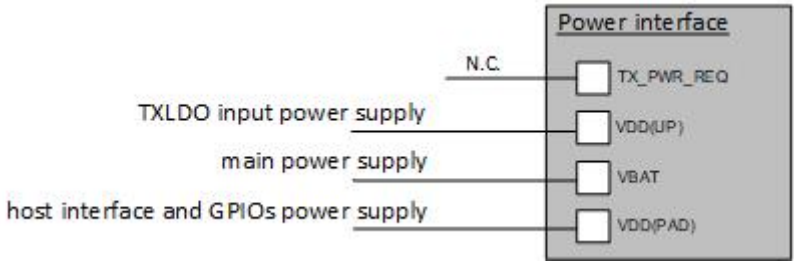

Figure 37. CFG2 - VDD(UP) connected to external power supply

An external DC-DC can be alternatively be used to supply VDD(UP).

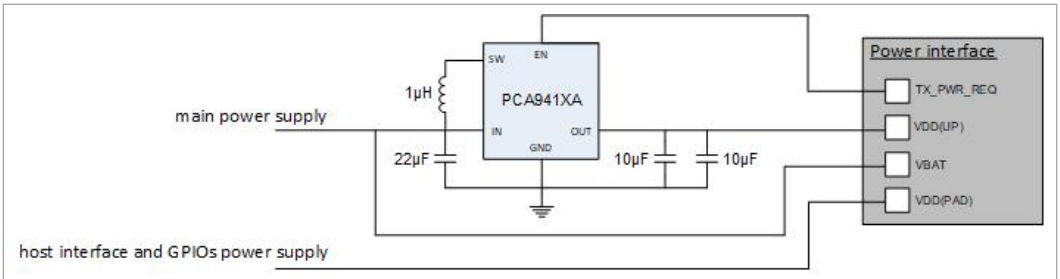

Figure 38. CFG2 - VDD(UP) connected to external DC-DC

8.7 Dynamic LMA

The PN7160 supports Dynamic Load Modulation Amplitude for P2P Target passive and Card Emulation. This feature is set thanks to DLMA_CTRL, DLMA_RSSI and DLMA_TX parameters (see Section 13.3).

Below is the detailed description of these 3 parameters.

Table 43. DLMA_CTRL parameter description

Byte	Description	Default value
byte 0	bit [7] = Type A&B DLMA enable/disable bit [6:0] = 0000011 must not be modified	0x83
byte 1	bit [7] = Type A&B SINGLE driver possibility enable/disable bit [6] = Type A&B BPSK possibility enable/disable bit [5] = Type A&B TX SHIFT sign (0 - positive, 1 - negative) bit [4:0] = Type A&B TX SHIFT absolute value	0xc5
bytes 3-2	bit [15:12] = Type A&B TXLDO limit (linearized via LUT: 0->2.70V, 1->3V, 2->3.3V, 3->3.6V, 4->3.9V, 5->4.2V, 6->4.5V, 7->4.7V, 8->4.75V, 9->5V, 10->5.25V) bit [11:10] = RFU bit [9:0] = Type A&B RSSI SCALING value	0x908f
byte 4	RFU - must not be modified	0x00
byte 5	bit [7] = Type F DLMA enable/disable bit [6:0] = 0000011 must not be modified	0x83
byte 6	bit [7] = Type F SINGLE driver possibility enable/disable bit [6] = Type F BPSK possibility enable/disable bit [5] = Type F TX SHIFT sign (0 - positive, 1 - negative) bit [4:0] = Type F TX SHIFT absolute value	0xc5
bytes 8-7	bit [15:12] = Type F TXLDO limit (linearized via LUT: 0->2.70V, 1->3V, 2->3.3V, 3->3.6V, 4->3.9V, 5->4.2V, 6->4.5V, 7->4.7V, 8->4.75V, 9->5V, 10->5.25V) bit [11:10] = RFU bit [9:0] = Type F RSSI SCALING value	0x908f
byte 9	RFU - must not be modified	0x00

Table 43. DLMA_CTRL parameter description*...continued*

Byte	Description	Default value
byte 10	bit [7] = RFU bit [6] = Type F AGC phase compensation enable/disable bit [5] = Type F TX phase compensation enable/disable bit [4] = Type F trimmed phase compensation enable/disable bit [3] = RFU bit [2] = Type A&B AGC phase compensation enable/disable bit [1] = Type A&B TX phase compensation enable/disable bit [0] = Type A&B trimmed phase compensation enable/disable	0x77
byte 11	RFU - must not be modified	0x08

Table 44. DLMA_RSSI parameter description

Byte	Description	Default value
bytes 1-0	Periodic timer to reapply RSSI (0423h -> ~10 ms)	0x0423
byte 2	Number of Entries for type A&B	0x18
bytes 4-3	Entry 01 type A&B RSSI value: bit [15:13] = RFU bit [12:0] = RSSI value	0x0035
byte 5	Entry 01 type A&B Phase offset value: [-127°: +127°] bit [7] = sign (0 - positive, 1 - negative) bit [6:0] = phase offset value 0° to 127°	0x00
bytes 7-6	Entry 02 type A&B RSSI value: bit [15:13] = RFU bit [12:0] = RSSI value	0x004b
byte 8	Entry 02 type A&B Phase offset value: [-127°: +127°] bit [7] = sign (0 - positive, 1 - negative) bit [6:0] = phase offset value 0° to 127°	0x00
bytes 9...74	Following type A&B entries	...
byte 75	Number of Entries for type F	0x18
bytes 77-76	Entry 01 type F RSSI value: bit [15:13] = RFU bit [12:0] = RSSI value	0x0035
byte 78	Entry 01 type F Phase offset value: [-127°: +127°] bit [7] = sign (0 - positive, 1 - negative) bit [6:0] = phase offset value 0° to 127°	0x00
bytes 80-79	Entry 02 type F RSSI value: bit [15:13] = RFU bit [12:0] = RSSI value	0x004b
byte 81	Entry 02 type F Phase offset value: [-127°: +127°] bit [7] = sign (0 - positive, 1 - negative) bit [6:0] = phase offset value 0° to 127°	0x00

Table 44. DLMA_RSSI parameter description*...continued*

Byte	Description	Default value
bytes 82...147	Following type F entries	...

Table 45. DLMA TX setting register decoding

Byte	Description	Default value
bytes 3-0	Entry 01 TX settings value	0x0a00c100
bytes 7-4	Entry 02 TX settings value	0x0a418001
bytes 8…155	Following TX settings entries	...
bytes 159-156	Entry 03 TX settings value	0x01f0531f

This feature and the recommendation procedure to use it are described in [AN13223], as referenced in →Section 3.

9 Poll side: Reader/Writer Mode

9.1 T1T, T2T, MIFARE Ultralight, MIFARE Classic and MIFARE Plus tags

All the tags/cards in this category are based on NFC-A technology, but they do not support the ISO-DEP Protocol.

MIFARE Plus product-based cards support the ISO-DEP protocol, but only when they are configured in Security Level3, which is out of scope for this section.

9.1.1 The [NCI] Frame RF Interface

[NCI] allows the data exchange with tags T1T, T2T using the Frame RF Interface.

Most of the commands of the MIFARE Classic and MIFARE Plus can also be mapped on the Frame RF Interface, but NXP decided to use a separate RF interface (TAG-CMD, see →Section 9.1.3) because the MIFARE Classic Authenticate command is split in 2 steps and has a tight response timeout (1ms) which can hardly be monitored by the DH through the NFCC.

Here is a summary of the tags/card based on technology NFC-A that can be accessed through the Frame RF interface.

Table 46. Tag/Cards accessible over the [NCI] Frame RF Interface

Tag/Card	Access through the Frame RF Interface
T1T	Supported
T2T	Supported
MIFARE Ultralight, Ultralight C	Supported
MIFARE Classic	Not supported
MIFARE Plus for Security levels 1 and 2	Not supported

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for T1T and T2T through the Frame RF Interface:

Table 47. Configuration sequence for Reader/Writer of T1T or T2T through the Frame RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD*	RF Protocol (choose between the 2 possible protocols)	PROTOCOL_T1T
		PROTOCOL_T2T
		PROTOCOL_T5T (NCI2.0)
	Mode	Poll
	RF Interface	Frame RF Interface
CORE_SET_CONFIG_CMD	PA_BAIL_OUT[1]	
RF_DISCOVER_CMD	RF Technology and Mode	NFC_A_PASSIVE_POLL_MODE

* Note: RF_DISCOVER_MAP_CMD is optional since the mapping to Frame RF Interface is done by default.

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-A, whatever the value written by the DH to that parameter.

Important Note: In the setting “AGG_INTF_CONFIG”, the NCI standard includes a special field called “NFCC Aggregation Enabled” which NFCCs may follow or not. If the “NFCC Aggregation Enabled” field is set to 1b, an NFCC may aggregate RF frames received from the Remote RF Endpoint. This means in that case, the NFCC would collect all responses from a Remote RF Endpoint first before forwarding them to the DH at once. This feature is not supported by PN7160, instead it will always forward responses immediately, similar to pipelining. Therefore the value of the field “NFCC Aggregation Enabled” is ignored by PN7160.

9.1.2 The [NCI] Frame Aggregated RF Interface Extension

NCI2.0 allows the data exchange with tags T1T, T2T using the Frame Aggregated RF Interface Extension.

Here is a summary of the Tags/Card based on technology NFC-A that can be accessed through the Aggregated Frame RF interface.

Table 48. Tag/Cards accessible over the [NCI] Frame RF Interface

Tag/Card	Access through the Aggregated Frame RF Interface
T1T	Supported
T2T	Supported
MIFARE Ultralight, Ultralight C	Supported
MIFARE Classic	Not supported
MIFARE Plus for Security levels 1 and 2	Not supported

Pre-requisite: the Frame Aggregated RF interface extension is only available after activation of the Frame RF Interface with protocol T1T and T2T. Here is the command with the parameters to activate the Frame Aggregated RF interface mechanism:

Table 49. RF_INTF_EXT_START_CMD

GID	OID	Main Parameters	Values
0001b	0x0A	Extension	Frame Aggregated RF Interface Extension
		Start parameter length	2
		Start parameter	1 byte retry count (1) (0-7) 1 byte Command Timeout

/!\	[1] The aggregation by the NFCC of several responses from the RF End Point is not supported. So if bit 7 of the first bye of the start parameter of the RF_INTF_EXT_START_CMD is set to 1b, it will be ignored by the NFCC.

After the DH has sent an aggregated list of commands via an NCI data packet, the NFCC will proceed the list and execute all command one after the other starting from the first one in the aggregated list of commands.

After processing a T1T or T2T command to the RF End point, the NFCC forwards immediately to the DH T1T or T2T response received from the RF End Point and processes the next command. If the DH does not fetch T1T or T2T response, the processing of T1T or T2T command will be stalled until the previous T1T or T2T response is successfully read by the DH.

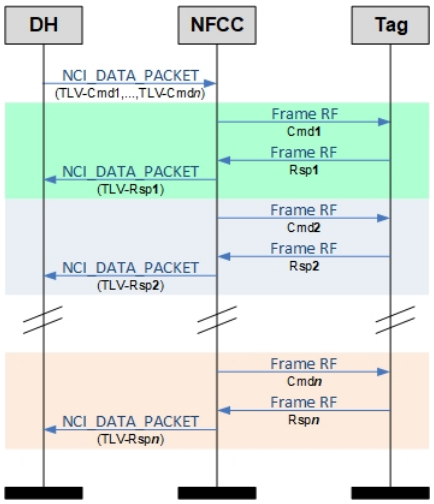

Figure 39. Messaging during Frame Aggregated Rf Interface extension

The NFCC does not support NCI chaining in NCI data packet after the Frame Aggregated RF interface extension has been started.

/!\	If the DH sends an NCI data packet with the PBF bit set to one, the NCI data packet will be rejected by the NFCC.

If the DH needs to send a collection of commands which does not fit a single NCI data packet payload capacity, the commands shall be subdivided per set of commands so that each set fit a single NCI data packet payload capacity.

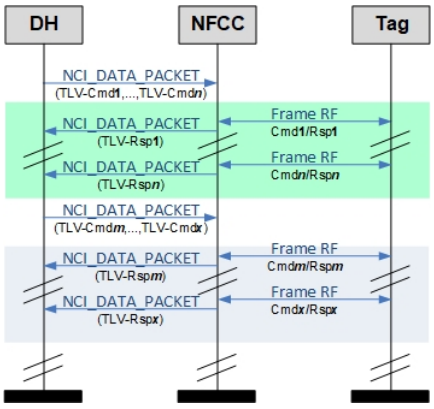

Figure 40. Hudge set of commands during Frame Aggregated Rf Interface extension

Is a T1T or T2T response is more than 252 bytes, it cannot be embedded in single NCI data packet because the TLV aggregated response forwarded by NFCC to the DH has 3 bytes of overlay: the TLV-object type, the TLV-object length and the NCI status. As a

consequence, the NFCC will forward it to the DH within 2 chained NCI data packet where the PBF bit field of the first NCI data packet is set to 1.

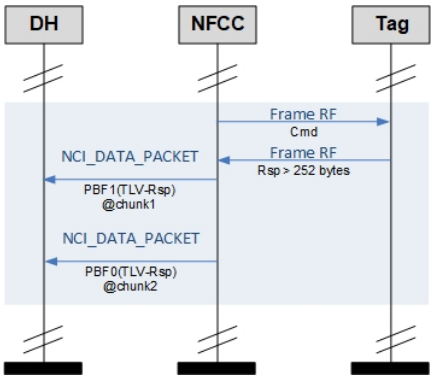

Figure 41. Response chaining during Frame Aggregated Rf Interface extension

9.1.3 [PN7160-NCI] extension: TAG-CMD Interface

In addition to the incompatibility of the Frame RF Interface with the MIFARE Classic Authenticate command described in the previous chapter, the intention when introducing the TAG-CMD interface was to add some commands such as ReadN/WriteN which would allow to read/write multiple bytes, and would rely on the NFCC to call several times the basic read/write commands defined in the T1T, T2T or MIFARE Classic protocols.

Unfortunately, we had to withdraw this concept and the TAG-CMD as implemented in PN7160 is limited to MIFARE Classic operation in Reader/Writer and T2T operation in Reader/Writer when the Sector Select command is required.

The figure bellow represents the location of the TAG-CMD RF Interface:

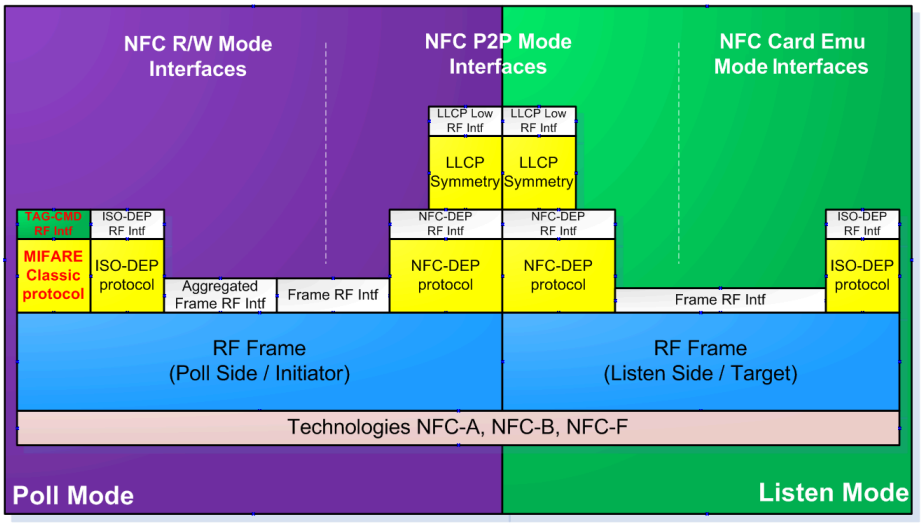

Figure 42. TAG-CMD RF Interface

9.1.4 [PN7160-NCI] extension: Payload structure of the TAG-CMD RF Interface

The TAG-CMD RF Interface is using the same data mapping as the one defined for the [NCI] Frame RF Interface (see section 8.2.1 in [NCI]). However, for the TAG-CMD RF Interface, the Payload is defined differently.

Two different structures are defined:

- REQ (requests): these are commands from the DH to the NFCC
- RSP (responses): these are responses from the NFCC to the DH.

The diagram below details how the Payload is modified to insert a header, which carries the REQ ID or the RSP ID and some parameters, if required.

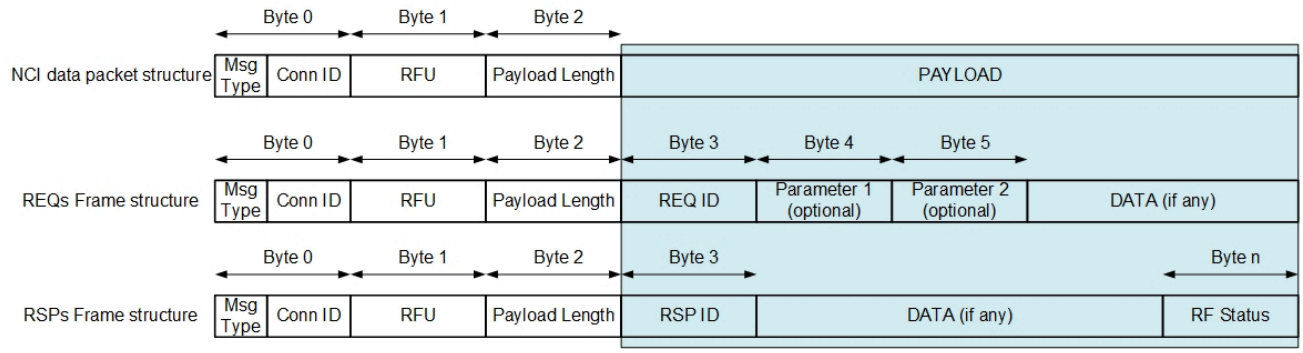

Figure 43. Data message payload for the TAG-CMD Interface

REQs and RSPs do not share exactly the same structure:

- REQs: Although illustrated with 2 parameters on the figure above, REQs may have no parameters or only one. Some REQuests might also need parameters bigger than 1 Byte. Parsing The REQ ID is the way to know how many parameters follow and how long they are.
- RSPs: there are no parameters in ReSPonses. A Byte is added at the end of the payload (after the DATA field) to inform the DH on the RF status (to report RF errors if they were some). The Status codes used are the following:

Table 50. TAG-CMD RF Status code

Value	Description
0x00	STATUS_OK
0x03	STATUS_FAILED
0xB0	RF_TRANSMISSION_ERROR
0xB1	RF_PROTOCOL_ERROR
0xB2	RF_TIMEOUT_ERROR
Others	RFU

9.1.5 [PN7160-NCI] extension: REQs and RSPs rules

A REQ command is always going from DH to RF, through the NFCC.

An RSP response is always going from the RF to the DH, through the NFCC.

The DH SHALL wait until it has received an RSP associated to a REQ before it can send a new REQ.

9.1.6 [PN7160-NCI] extension: List of REQs and RSPs

In this section, the following acronyms are used:

Table 51. Acronyms definition

Acronym	Description
T1T	NFC Forum Type 1 Tag (based on Topaz/Jewel)
MF	MIFARE family, not ISO-DEP compliant, including T2T, MIFARE Ultralight (std or C), MIFARE Classic and MIFARE Plus for Security Level 1 and 2.
MFC	MIFARE Classic and MIFARE Plus for Security Level 1 and 2.

The added REQuests/ReSPonses pairs are listed in the following table:

Table 52. List of REQuests and RESPonses

REQ/RSP Name	ID	Param 1	Param 2	Param 3	Data	Description
XCHG_DATA_REQ	0x10	None	None	None	Yes	MFC: DH sends Raw data to the NFCC, which encrypts them before sending them to MFC. T1T/T2T: DH sends Raw data to the NFCC, which forwards them in plain to the Tag.
XCHG_DATA_RSP	0x10	N/A	N/A	N/A	Yes	MFC: DH gets Raw data once RF data from MFC are decrypted by the NFCC, if successful. T1T/T2T: DH gets Raw plain data once the NFCC receives RF data from the Tag, if successful.
MF_SectorSel_REQ	0x32	Sector Address	None	None	No	T2T and MFU only: DH Sends the address of the Block to select.
MF_SectorSel_RSP	0x32	N/A	N/A	N/A	No	T2T and MFU only: DH gets the “Sector Select” response status
MFC_Authenticate_REQ	0x40	Sector Address	Key Selector	Key (optional)	No	DH asks NFCC to perform MFC Authenticate command.
MFC_Authenticate_RSP	0x40	N/A	N/A		No	DH gets the MFC Authenticate command status

All these REQs and RSPs are detailed in the next sections.

9.1.7 [PN7160-NCI] extension: raw data exchange REQs and RSPs

Table 53. XCHG_DATA_REQ

REQ_ID	REQ Name	Number of parameter(s)	Presence of data	Description
0x10	XCHG_DATA_REQ	0	Yes	MFC: DH sends Raw data to the NFCC, which encrypts them before sending them to MFC. T1T/T2T: DH sends Raw data to the NFCC, which forwards them in plain to the Tag.

Table 54. XCHG_DATA_RSP

RSP_ID	RSP Name	Presence of Data	Description
0x10	XCHG_ DATA_RSP	Yes	MFC: DH gets Raw data once RF data from MFC are decrypted by the NFCC, if successful. T1T/T2T: DH gets Raw plain data once the NFCC receives RF data from the Tag, if successful. If the response from the MF tag in the field is an ACK or a NACK, the ACK/ NACK is also sent back to the DH inside the Data field. Since ACK and NACK are 4-bit commands, they are transported on the 4 LSBs of the data Byte; the 4MSBs of that Byte are forced to the logical '0' value.

9.1.8 [PN7160-NCI] extension: T2T and MFU REQs and RSPs

All the REQs and RSPs described in this section can be used whatever the tag between:

- T2T
- MIFARE Ultralight (std or C)

Table 55. MF_SectorSel_REQ

REQ_ID	REQ Name	Number of parameter(s)	Presence of data	Description
0x32	MF_SectorSel_ REQ	1	No	DH Sends the address of the Sector to select.

Table 56. MF_SectorSel_REQ parameter

Parameter	Length	Description
Sector Address	1 Byte	Defines the address of the sector which has to be selected. The address can be any block address in this sector.

Table 57. MF_SectorSel_RSP

RSP_ID	RSP Name	Presence of Data	Description
0x32	MF_SectorSel_ RSP	No	DH gets sector select status

9.1.9 [PN7160-NCI] extension: MIFARE Classic REQs and RSPs

Table 58. MFC_Authenticate_REQ

REQ_ID	REQ Name	Number of parameter(s)	Presence of data	Description
0x40	MFC_ Authenticate_REQ	2 or 3	No	DH asks NFCC to perform MFC authenticate.

Table 59. MFC_Authenticate_REQ parameters

<table>
<tr><th colspan="2">Parameter</th><th>Length</th><th>Value/Description</th></tr>
<tr><td>1</td><td>Sector Address</td><td>1 Byte</td><td>Address of the sector to authenticate</td></tr>
<tr><td>2</td><td>Key Selector</td><td>1 Byte</td><td>
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th><th></th></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Key A (‘0’)
or Key B (‘1’)</td></tr>
<tr><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td><td>0 => use pre-loaded key
1 => use Key embedded in the REQ (param Nbr 3)</td></tr>
<tr><td></td><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td>Pre-loaded key number (0 to 15)</td></tr>
<tr><td></td><td>0</td><td>0</td><td></td><td></td><td></td><td></td><td></td><td>RFU</td></tr>
</table>
</td></tr>
<tr><td>3</td><td>Embedded Key (optional)</td><td>6 Bytes</td><td>This parameter is present in the MFC_Authenticate_CMD only if bit b4 is set to logical '1' in Key Selector parameter. If present, this parameter defines the value of the Key used for the Authentication.</td></tr>
</table>

Table 60. MFC_Authenticate_RSP

RSP_ID	RSP Name	Presence of Data	Description
0x40	MFC_ Authenticate_RSP	No	DH gets the “authenticate” cmd status

Table 61. TAG-CMD RF Status code, in the special case of MFC_Authenticate_CMD

Value	Description	Reason
0x00	STATUS_OK	Authentication was successful
0x03	STATUS_FAILED	Authentication failed (wrong key, time-out triggered during authentication…)
0xB0	RF_TRANSMISSION_ERROR	Not used
0xB1	RF_PROTOCOL_ERROR	Not used
0xB2	RF_TIMEOUT_ERROR	Not used
Others	RFU	

Once a sector is authenticated, PN7160 will automatically encrypt any data sent by the DH to be transferred over RF, thanks to the XCHG_DATA_REQ command. The key used is the one used for the sector currently authenticated.

In a symmetrical way, PN7160 will automatically decrypt the data received from RF before it forwards to the DH thanks to the XCHG_DATA_RSP response, again using the key of the sector currently authenticated.

Here is a typical NFC reader for MIFARE Classic sequence, to illustrate the use of the MFC_Authenticate_REQ and XCHG_DATA_REQ.

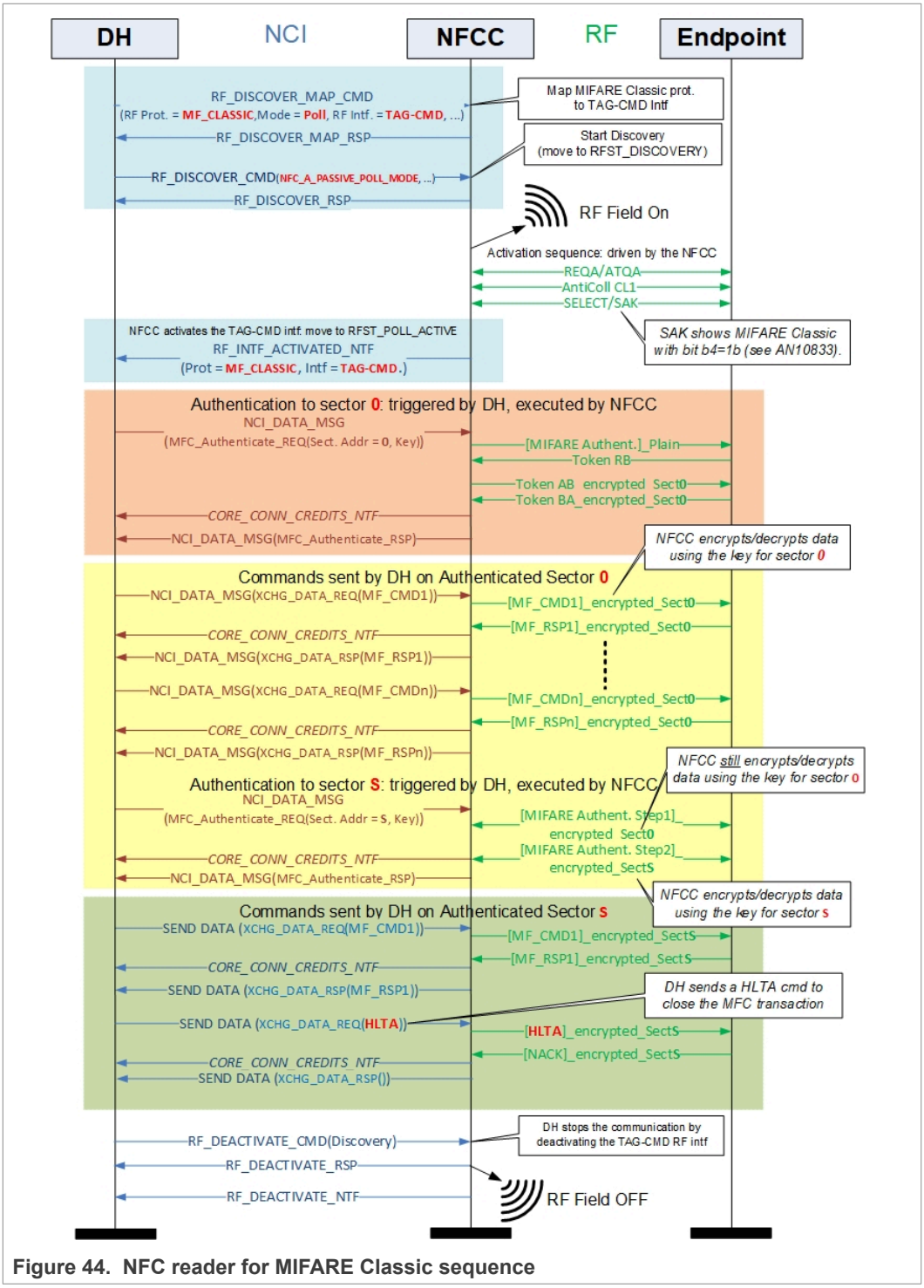

Figure 44. NFC reader for MIFARE Classic sequence

9.1.10 Access through the TAG-CMD RF Interface

The TAG-CMD RF interface allows full access to all the Tags based on NFC-A technology and not supporting the ISO-DEP protocol, leaving up to the PN7160 to manage the low level TAG-CMD:

Table 62. Tag/Cards accessible over the TAG-CMD Interface

Tag/Card	Access through the TAG-CMD Interface
T1T	Supported
T2T	Supported
MIFARE Ultralight, Ultralight C	Supported
MIFARE Classic	Supported
MIFARE Plus for Security levels 1 and 2	Supported

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for T1T, T2T, MIFARE Classic through the TAG-CMD Interface:

Table 63. Configuration sequence for R/W of T1T, T2T and MFC through the TAG-CMD Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol (choose between the 3 possible protocols)	PROTOCOL_T1T
		PROTOCOL_T2T
		PROTOCOL_MIFARE_CLASSIC
	Mode	Poll
	RF Interface	TAG-CMD
CORE_SET_CONFIG_CMD	PA_BAIL_OUT [1]	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_PASSIVE_POLL_MODE

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-A, whatever the value written by the DH to that parameter.

9.2 T3T tag

[NCI] allows the data exchange with a tag T3T by using the Frame RF Interface, so there is no need to add proprietary extensions here.

9.2.1 Access through the Frame RF Interface

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for T3T Tags/Cards through the Frame RF Interface:

Table 64. Configuration sequence for Reader/Writer of T3T through the Frame RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_T3T
	Mode	Poll
	RF Interface	Frame
CORE_SET_CONFIG_CMD	PF_BIT_RATE	

Table 64. Configuration sequence for Reader/Writer of T3T through the Frame RF Interface*...continued*

Command	Main Parameters	Values
	PF_RC_CODE	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_F_PASSIVE_POLL_MODE

9.2.2 Access through the aggregated Frame RF interface

NCI2.0 allows the data exchange with tags T3T using the Frame Aggregated RF Interface Extension.

Pre-requisite: the Frame Aggregated RF interface extension is only available after activation of the Frame RF Interface with protocol T3T. Here is the command with the parameters to activate the Frame Aggregated RF interface mechanism:

Table 65. RF_INTF_EXT_START_CMD

GID	OID	Main Parameters	Values
0001b	0x0A	Extension	Frame Aggregated RF Interface Extension
		Start parameter length	2
		Start parameter	1 byte retry count [1] (0-7) 1 byte Command Timeout

/!\	[1] The aggregation by the NFCC of several responses from the RF End Point is not supported. So if bit 7 of the first bye of the start parameter of the RF_INTF_EXT_START_CMD is set to 1b, it will be ignored by the NFCC.

After the DH has sent an aggregated list of commands via an NCI data packet, the NFCC will proceed the list and execute all command one after the other starting from the first one in the aggregated list of commands.

After processing a T3T command to the RF End point, the NFCC forwards immediately to the DH T3T response received from the RF End Point and processes the next command. If the DH does not fetch T3T response, the processing of T3T command will be stalled until the previous T3T response is successfully read by the DH.

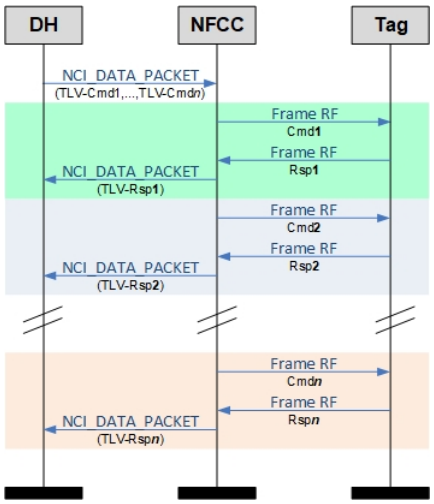

Figure 45. Messaging during Frame Aggregated Rf Interface extension

The NFCC does not support NCI chaining in NCI data packet after the Frame Aggregated RF interface extension has been started.

/!\	If the DH sends an NCI data packet with the PBF bit set to one, the NCI data packet will be rejected by the NFCC.

If the DH needs to send a collection of commands which does not fit a single NCI data packet payload capacity, the commands shall be subdivided per set of commands so that each set fit a single NCI data packet payload capacity.

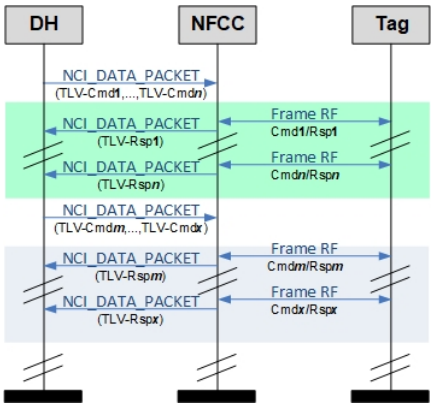

Figure 46. Hudge set of commands during Frame Aggregated Rf Interface extension

Is a T3T response is more than 252 bytes, it cannot be embedded in single NCI data packet because the TLV aggregated response forwarded by NFCC to the DH has 3 bytes of overlay: the TLV-object type, the TLV-object length and the NCI status. As a consequence, the NFCC will forward it to the DH within 2 chained NCI data packet where the PBF bit field of the first NCI data packet is set to 1.

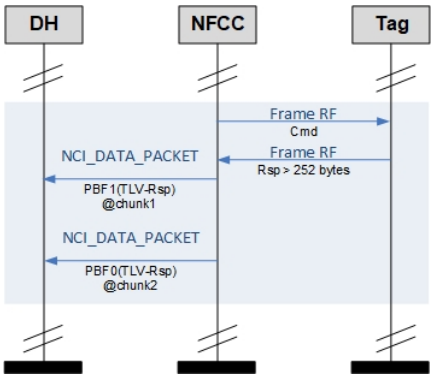

Figure 47. Response chaining during Frame Aggregated Rf Interface extension

9.3 T4T and ISO-DEP Tags/Cards

[NCI] allows the data exchange with a T4T tag or an ISO-DEP tag by using the Frame RF Interface or the ISO-DEP RF Interface, so there is no need to define a proprietary RF interface here.

9.3.1 Access through the Frame RF Interface

The Frame RF interface allows full access to all the Tags based on NFC-A and NFC-B technology and supporting the ISO-DEP protocol, assuming that the ISO-DEP protocol is fully handled by the DH:

Table 66. Tag/Cards accessible over the Frame RF Interface

Tag/Card	Access through the Frame RF Interface
T4T	Supported
MIFARE DESFire	Supported
MIFARE Plus for Security levels 3	Supported
JCOP-based smart cards	Supported

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for ISO-DEP Tags/Cards through the Frame RF Interface for technology NFC-A:

Table 67. Configuration sequence for R/W of NFC-A / ISO-DEP through the Frame RF interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD *	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Poll
	RF Interface	Frame
CORE_SET_CONFIG_CMD	PA_BAIL_OUT [1]	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_PASSIVE_POLL_MODE

* Note: RF_DISCOVER_MAP_CMD is optional since the mapping to Frame RF Interface is done by default.

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-A, whatever the value written by the DH to that parameter.

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for ISO-DEP Tags/Cards through the Frame RF Interface for technology NFC-B:

Table 68. Configuration sequence for R/W of NFC-B / ISO-DEP through the Frame RF interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD *	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Poll
	RF Interface	Frame
CORE_SET_CONFIG_CMD	PB_AFI	
	PB_BAIL_OUT [1]	
	PB_SENSB_REQ_PARAM [2]	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_B_PASSIVE_POLL_MODE

* Note: RF_DISCOVER_MAP_CMD is optional since the mapping to Frame RF Interface is done by default.

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-B, whatever the value written by the DH to that parameter.

/!\	[2] this parameter is not supported in PN7160: STATUS_INVALID_PARAM will be returned to the DH if it attempts to write this parameter.

9.3.2 Access through the ISO-DEP RF Interface

The ISO-DEP RF interface allows full access to all the Tags based on NFC-A and NFC-B technology and supporting the ISO-DEP protocol, leaving up to the PN7160 to manage the ISO-DEP protocol:

Table 69. Tag/Cards accessible over the ISO-DEP RF Interface

Tag/Card	Access through the ISO-DEP RF Interface
T4T	Supported
MIFARE DESFire	Supported
MIFARE Plus for Security levels 3	Supported
JCOP-based smart cards	Supported

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for ISO-DEP through the ISO-DEP Interface for technology NFC-A:

Table 70. Configuration sequence for R/W of NFC-A / ISO-DEP through the ISO-DEP interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Poll
	RF Interface	ISO-DEP
CORE_SET_CONFIG_CMD	PA_BAIL_OUT [1]	
	PI_BIT_RATE	
	PA_ADV_FEAT [3]	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_PASSIVE_POLL_MODE

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-A, whatever the value written by the DH to that parameter.

/!\	[3] this parameter is not supported in PN7160: STATUS_INVALID_PARAM will be returned to the DH if it attempts to write this parameter.

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for ISO-DEP through the ISO-DEP Interface for technology NFC-B:

Table 71. Configuration sequence for R/W of NFC-B / ISO-DEP through the ISO-DEP interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Poll
	RF Interface	ISO-DEP
CORE_SET_CONFIG_CMD	PB_AFI	
	PB_BAIL_OUT [1]	
	PB_H_INFO	
	PI_BIT_RATE	
	PB_SENSB_REQ_PARAM [3]	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_B_PASSIVE_POLL_MODE

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-B, whatever the value written by the DH to that parameter.

/!\	[3] this parameter is not supported in PN7160: STATUS_INVALID_PARAM will be returned to the DH if it attempts to write this parameter.

9.3.3 [PN7160-NCI] extension: Presence check Command/Response

/!\	The Presence check command/response is an extension of NCI1.0, see ISO-DEP R(NAK) Presence Check section for NCI2.0 command/response.

When a Tag/Card has been activated in Poll Mode, the RF State Machine is then in state RFST_POLL_ACTIVE. It is useful for the DH to know if the card is still in the field or not, especially at the end of the transaction. For that purpose, NXP has added a proprietary command to check the Tag/Card presence.

All the rules defined for command/response in [NCI] (section 3.2) apply to the command defined here. Here are two additional rules:

- The DH can use this command ONLY if the RF State Machine is in state RFST_POLL_ACTIVE. PN7160 will respond “STATUS_SEMANTIC_ERROR” in case this command is sent in any other state.
- The DH can use this command ONLY if the active protocol is either ISO-DEP or NFC-DEP.

Table 72. RF_PRES-CHECK_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x11	0	The DH asks to know if the ISO-DEP Tag/Card is in the field or not.

Table 73. RF_PRES-CHECK_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x11	1	The NFCC acknowledges the command received from the DH.

Table 74. RF_PRES-CHECK_RSP parameter

Parameter	Length	Value/Description
Status	1 byte	One of the following Status codes, as defined in [NCI_Table1] 0x00: STATUS_OK 0x01: STATUS_REJECTED 0x06: STATUS_SEMANTIC_ERROR Others: RFU

Table 75. RF_PRES-CHECK_NTF

GID	OID	Numbers of parameter(s)	Description
1111b	0x11	1	NFCC indicates if the ISO-DEP Tag/Card is still in the field or not.

Table 76. RF_PRES-CHECK_NTF parameter

Parameter	Length	Value/Description
Presence	1 byte	0x00: Card no more in the field 0x01: Card still in the field 0x02-0xFF: RFU

9.3.4 [PN7160-NCI] ISO-DEP R(NAK) Presence Check

NCI2.0 defines the ISO-DEP R(NAK) Presence Check to the ISO-DEP RF Interface. When a Tag/Card has been activated in Poll Mode, the RF State Machine is then in state RFST_POLL_ACTIVE. It is useful for the DH to know if the card is still in the field or not, especially at the end of the transaction.

All the rules defined for command/response in [NCI] (section 3.2) apply to the command defined here. Here are two additional rules:

- The DH can use this command ONLY if the RF State Machine is in state RFST_POLL_ACTIVE. PN7160 will respond “STATUS_SEMANTIC_ERROR” in case this command is sent in any other state
- The DH can use this command ONLY if the active protocol is either ISO-DEP or NFC-DEP

Table 77. RF_ISO_DEP_NACK_PRESENCE_CMD

GID	OID	Numbers of parameter(s)	Description
0001b	0x10	0	The DH asks to know if the ISO-DEP Tag/Card is in the field or not.

Table 78. RF_ ISO_DEP_NACK_PRESENCE_RSP

GID	OID	Numbers of parameter(s)	Description
01b	0x10	1	The NFCC acknowledges the command received from the DH.

Table 79. RF_ ISO_DEP_NACK_PRESENCE_RSP parameter

Parameter	Length	Value/Description
Status	1 byte	One of the following Status codes, as defined in [NCI_Table1] 0x00: STATUS_OK 0x01: STATUS_REJECTED 0x06: STATUS_SEMANTIC_ERROR Others: RFU

Table 80. RF_ ISO_DEP_NACK_PRESENCE _NTF

GID	OID	Numbers of parameter(s)	Description
0001b	0x10	1	NFCC indicates if the ISO-DEP Tag/Card is still in the field or not.

Table 81. RF_ ISO_DEP_NACK_PRESENCE _NTF parameter

Parameter	Length	Value/Description
Presence	1 byte	0x00: Card no more in the field 0x01: Card still in the field 0x02-0xFF: RFU

9.3.5 [PN7160-NCI] extension: WTX notification

After data was sent to the card/tag, it can request an additional processing time before sending data response. This is done with WTX (Waiting Time Extension) request. If WTX REQ/RESP exchange phase continues an NCI system notification WTX is sent with a period configurable via READER_FWITOX_NTF_CFG.

Table 82. PH_NCI_OID_SYSTEM_WTX

GID	OID	Numbers of parameter(s)	Description
1111b	0x17	0	Notification indicating that RF communication is in phase of WTX(RTOX) REQ/RESP exchange for longer period of time.

9.3.6 [PN7160-NCI] extension: Higher bit rates in Poll NFC-A and NFC-B

[NCI] does not “officially” support the use of higher bit rates in technology NFC-A and NFC-B.

PN7160 offers 4 different bit rates for these technologies, which can be used either in Poll Mode (to read/write an external Card/Tag) or in Listen Mode (to emulate a card):

1. 106 kb/s (default bit rate, always used during activation)
2. 212 kb/s
3. 424 kb/s
4. 848 kb/s

Everything is prepared (see the RF configuration parameter PI_BIT_RATE), except for the ISO-DEP RF Interface activation.

As currently defined in [NCI], the ISO-DEP RF interface activation for technology NFC-A is incompatible with bit rates higher than 106 kb/s, since this requires to handle the PPS commands exchange, which is not addressed in [NCI].

So the PN7160 implements an ISO-DEP RF Interface activation which is different from the one described in [NCI_Chapt1] (see chapter →3). Here is a copy of this chapter, where the modification as implemented in the PN7160 is highlighted in ***bold italic***:

Copy from [NCI]

8.3.2.2 Discovery and Interface Activation

To enable Poll Mode for ISO-DEP, the DH sends the RF_DISCOVER_CMD to the NFCC containing configurations with RF Technology and Mode values of NFC_A_PASSIVE_POLL_MODE and/or NFC_B_PASSIVE_POLL_MODE.

When the NFCC is ready to exchange data (that is, after receiving a response to the protocol activation command from the Remote NFC Endpoint), it sends the RF_INTF_ACTIVATED_NTF to the DH to indicate that this Interface has been activated to be used with the specified Remote NFC Endpoint.

Detailed ISO-DEP RF Interface activation handling in the NFCC:

For NFC-A:

Following the anti-collision sequence, if the Remote NFC Endpoint supports ISO-DEP Protocol, the NFCC sends the RATS Command to the Remote NFC Endpoint and after receiving the RATS response, ***the NFCC MAY send the PPS command if PI_BIT_RATE was set by the DH to an allowed value higher than 0x00.*** It SHALL then send the RF_INTF_ACTIVATED_NTF to the DH to indicate a Remote NFC Endpoint based on ISO-DEP has been activated. The RF_INTF_ACTIVATED_NTF will inform the Dh on the actual bit rate used on RF.

For NFC-A the RF_INTF_ACTIVATED_NTF SHALL include the Activation Parameters defined in Table 83 (see below).

Table 83. Activation Parameters for NFC-A/ISO-DEP Poll Mode

Parameter	Length	Description
RATS Response Length	1 byte	Length of RATS Response Parameter (n)
RATS Response	n bytes	All Bytes of the RATS Response as defined in [DIGITAL] starting from and including Byte 2

End of Copy from [NCI]

Note: due to internal limitation, the default value set for bLA_RATS_RESP_TC1 parameter is not correct and shall be reset individually with a SET_Configuration

9.4 [PN7160-NCI] extension: 15693 and ICODE tags

The version 1.0 of the NCI standard allows the data exchange with a tag ISO 15693 by using the RF Frame interface. No additional interface is needed for this protocol. However, the data mapping is not yet defined in [NCI], therefore, NXP has defined it for [PN7160-NCI-1.0].

9.4.1 Access through the Frame RF interface

The Frame RF interface allows full access to all the Tags based on NFC-15693 technology. Here is a list of such tags from the NXP portfolio:

Table 84. NFC-15693 compliant Tag/Cards accessible over the Frame RF Interface

Tag/Card	Access through the Frame RF Interface
ICODE SLI	Supported
ICODE SLI-L	Supported
ICODE SLI-S	Supported

Here are the commands and configuration parameters to prepare the Reader/Writer Mode for NFC-15693 Tags/Cards through the Frame RF Interface:

Table 85. Configuration sequence for R/W of NFC-15693 through the Frame RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD *	RF Protocol	PROTOCOL_15693 (NCI1.0) PROTOCOL_T5T (NCI2.0)
	Mode	Poll
	RF Interface	Frame RF
RF_DISCOVER_CMD	RF Technology and Mode	NFC_15693_PASSIVE_POLL_MODE (NCI1.0) NFC_V_PASSIVE_POLL_ MODE (NCI2.0)

* Note: RF_DISCOVER_MAP_CMD is optional since the mapping to Frame RF Interface is done by default.

9.4.2 [PN7160-NCI] extension: Specific parameters for NFC_15693 Poll Mode

Once PN7160 detects and activates a remote NFC Endpoint based on NFC_15693, PN7160 will activate the Frame RF Interface, providing the following activation parameters:

Table 86. Specific parameters for NFC_15693 Poll Mode

Parameter	Length	Description
FLAGS	1 byte	1st Byte of the Inventory Response
DSFID	1 byte	2nd Byte of the Inventory Response
UID	8 bytes	3rd Byte to last Byte of the Inventory Response

9.4.3 [PN7160-NCI] extension: Data Mapping between the DH and RF

9.4.3.1 Data from the DH to RF

The NCI Data Message corresponds to the Request Format defined in [ISO15693-3] Section 7.3.

After receiving a Data Message from the DH, the PN7160 appends the appropriate EoD, SOF and EOF and then sends the result in an RF Frame in NFC-15693 technology to the Remote NFC Endpoint.

The following figure illustrates the mapping between the NCI Data Message Format and the RF frame when sending the RF frame to the Remote NFC Endpoint. This figure shows the case where NCI Segmentation and Reassembly feature is not used.

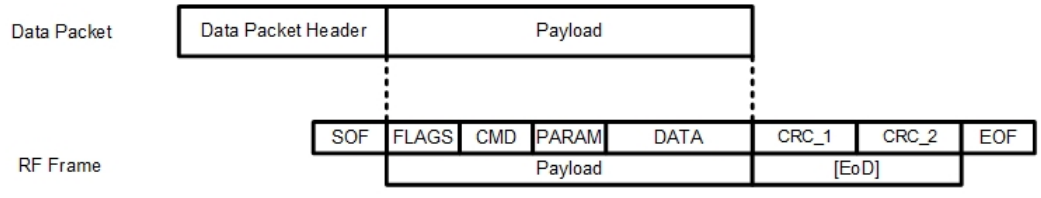

Figure 48. Format for Frame RF Interface (NFC-15693) for Transmission

Although the Frame RF interface is defined to be a transparent interface where the NFCC does not parse/modify the Bytes transmitted by the DH, the following exceptions occur:

/!\	PN7160 is parsing the bit Option_Flag (bit b7 in the request Flags Byte, as defined in ISO15693) to check if this bit is set by the DH or not. If set, this indicates that the tag is from TI, and PN7160 is sending commands over RF using a special mode, as defined for some commands in ISO15693.

9.4.3.2 Data from RF to the DH

The NCI Data Message corresponds to the Payload of the Response Format defined in [ISO15693-3] Section 7.4, followed by a Status field of 1 byte.

After receiving an RF frame, the PN7160 checks and removes the EoD, the SOF and EOF and sends the result in a Data Message to the DH.

In case of an error, the Data Message may consist of only a part of the Payload of the received RF frame but it will always include the trailing Status field. So the PN7160 may send a Data Message consisting of only the Status field if the whole RF frame is corrupted.

If the RF frame was received correctly, the PN7160 sets the Status field of Data Message to a value of STATUS_OK. If the PN7160 detected an error when receiving the RF frame, it sets the Status field of the Data Message to a value of STATUS_RF_FRAME_CORRUPTED.

The following figure illustrates the mapping of the RF frame received from the Remote NFC Endpoint in technology NFC-15693 to the Data Message format to be sent to the DH. This figure shows the case where NCI Segmentation and Reassembly feature is not used.

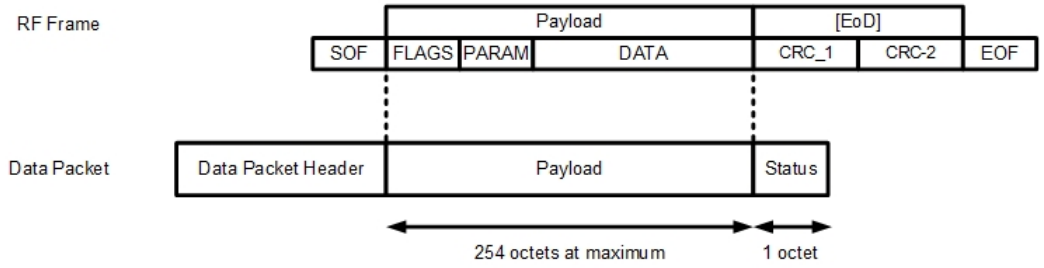

Figure 49. Format for Frame RF Interface (NFC-15693) for Reception

9.4.4 [PN7160] behavior with multiple VICCs

PN7160 supports collision resolution (using the Inventory command), so it can detect multiple VICCs (2 maximum, as defined for CON_DEVICE_LIMIT in →Section 7.3.5).

Here is the behavior when two VICCs are detected and then, one of them is removed from the Field before the DH wants to select it:

- PN7160 is in state RFST_DISCOVERY; it detects 2 VICCs. It sends an RF_DISCOVER_NTF to the DH for VICC1 and moves to RFST_W4_ALL_DISCOVERIES.
- [PN7160_Ref] is in state RFST_W4_ALL_DISCOVERIES, it sends an RF_DISCOVER_NTF to the DH for VICC2 and moves to RFST_W4_HOST_SELECT.

- [PN7160_Ref] is in state RFST_W4_ALL_DISCOVERIES and waits for the DH to select one of the 2 VICCs. Once it receives the RF_DISCOVER_SELECT_CMD from the DH, [PN7160_Ref] immediately activates the Frame RF Interface and does not check if the selected VICC is still in the field. That means that [PN7160_Ref] will not send a CORE_GENERIC_ERROR_NTF (Discovery_Target_Activation_Failed) to the DH if the selected VICC is not in the field anymore. The state is now changed to RFST_POLL_ACTIVE.
- PN7160 is in state RFST_POLL_ACTIVE; it waits for the DH to send some data to transfer over RF. Once it gets this data, [PN7160_Ref] forwards it over RF. If the selected VICC is not in the field anymore, PN7160 will stay mute and will not send any data back to the DH. The DH has to implement a time-out function, to detect that the VICC is not in the field anymore. Once this timeout is triggered, the DH can de-activate the Frame RF Interface by sending the RF_DEACTIVATE_CMD.

9.5 [PN7160-NCI] extension: KOVIO tags

Kovio tags are very particular tags which use a subset of NFC-A technology.

The basic concept is that the tag is powered from RF Field generated by PN7160, and it will spontaneously generate a 16-Byte ID using NFC-A load modulation, although it did not receive any command from PN7160. Once PN7160 has detected a Kovio tag by capturing its ID, PN7160 will send a RF_INTF_ACTIVATED_NTF, transporting the tag ID as RF parameter.

Table 87. Kovio specific RF parameters inside the RF_INTF_ACTIVATED_NTF

Payload Field(s)	Length	Value/Description
…		
Length of RF Technology Specific Parameters	1 byte	16 (0x10) / 32 (0x20)
RF Technology Specific Parameters	16/32 bytes	Kovio ID
…		

It is then up to the DH to decide when to leave the RFST_POLLING_ACTIVE state, and also to decide if it directly comes back to RFST_DISCOVERY, where the same Kovio tag may be discovered again, or if it comes back to RFST_IDLE first, in order to wait without any RF activity or re-configuring the RF Discovery so that PN7160 does not poll for a Kovio tag again.

9.5.1 Access through the [NCI] Frame RF Interface

Due to the very particular behavior of the Kovio tags, it is necessary to configure the RF Discovery specifically for these tags, using the NFC-A_KOVIO_POLL_MODE parameter for the RF_DISCOVER_CMD as highlighted in the table below:

Table 88. Configuration sequence for Reader/Writer of Kovio tags through the Frame RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_KOVIO
	Mode	Poll
	RF Interface	Frame RF Interface
CORE_SET_CONFIG_CMD	PA_BAIL_OUT*	

Table 88. Configuration sequence for Reader/Writer of Kovio tags through the Frame RF Interface*...continued*

Command	Main Parameters	Values
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_KOVIO_POLL_MODE

/!\	[1] this parameter is not active in PN7160: it can be read/written, but PN7160 will always behave with Bail Out in NFC-A, whatever the value written by the DH to that parameter.

10 Listen side: Card Emulation Mode

10.1 ISO-DEP based on NFC-A and NFC-B

For Card Emulation hosted by the DH based on either technology NFC-A or technology NFC-B, the PN7160 only supports the ISO-DEP protocol.

[NCI] defines all the mechanisms necessary to implement this feature. Two options are possible:

1. The DH wants to manage by itself the ISO-DEP protocol, it SHALL then map the ISO-DEP protocol on the Frame RF Interface.
2. The DH leaves the ISO-DEP protocol management to the NFCC: it SHALL then map the ISO-DEP protocol on the ISO-DEP interface.

10.1.1 Access through the Frame RF Interface

/!\	Not supported in PN7160

10.1.2 Access through the ISO-DEP RF Interface

Here are the commands and configuration parameters to prepare the ISO-DEP Card Emulation for technology NFC-A in the DH through the ISO-DEP RF Interface:

Table 89. Configuration sequence for ISO-DEP/NFC-A Card Emulation in the DH over ISO-DEP RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Listen
	RF Interface	ISO-DEP
CORE_SET_CONFIG_CMD	LA_BIT_FRAME_SDD	
	LA_PLATFORM_CONFIG	
	LA_SEL_INFO	
	LA_NFCID1	
	LI_FWI	
	LA_HIST_BY	
	LI_BIT_RATE	
RF_SET_LISTEN_MODE_ ROUTING_CMD	Technology-based routing	NFC_RF_TECHNOLOGY_A routed to DH-NFCEE
	Protocol-based routing	PROTOCOL_ISO-DEP routed to DH-NFCEE (!! Will also route NFC-B to NFCEE-DH !!)
	AID-Based routing	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_PASSIVE_LISTEN_MODE

Here are the commands and configuration parameters to prepare the ISO-DEP Card Emulation for technology NFC-B in the DH through the Frame RF Interface:

Table 90. Configuration sequence for ISO-DEP/NFC-B Card Emulation in the DH over ISO-DEP RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_ISO-DEP
	Mode	Listen
	RF Interface	ISO-DEP
CORE_SET_CONFIG_CMD	LB_SENSB_INFO	
	LB_NFCID0	
	LB_APPLICATION_DATA	
	LB_SFGI	
	LB_ADC_FO	
	LI_FWI	
	LB_H_INFO_RESP	
	LI_BIT_RATE	
RF_SET_LISTEN_MODE_ ROUTING_CMD	Technology-based routing	NFC_RF_TECHNOLOGY_B routed to DH-NFCEE
	Protocol-based routing	PROTOCOL_ISO-DEP routed to DH-NFCEE (!! Will also route NFC-A to NFCEE-DH !!)
	AID-Based routing	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_B_PASSIVE_LISTEN_MODE

10.2 NFC-F Card Emulation

10.2.1 Introduction to the Extended Routing Solution

PN7160 supports card emulation on the DH for type F as part of the so called Extended Routing Solution, which offers:

- Protocol-based routing to an NFC-DEP target
- Protocol-based routing to a T3T target

A simple state machine is used that consists of three different states:

- NEUTRAL
- NFC_DEP SELECTED
- T3T SELECTED

Initially the NEUTRAL state is active while waiting for commands being received from external RF counterparts like readers. Once a target is selected (so one of the two possible targets mentioned above) the SELECTED state is reached.

Depending on the structure of the SENF_REQ used by an external RF counterpart like a reader, a different target emulated by PN7160 will be accessible.

Table 91. Different SENSF_REQ structures supported by PN7160

System Code SC	RC	Response priority
FFFF	0	NFC-DEP, T3T

Table 91. Different SENSF_REQ structures supported by PN7160*...continued*

System Code SC	RC	Response priority
FFFF	1	T3T
FFFF	Other	T3T, NFC-DEP

10.2.2 Configuring the T3T card emulation

As described in the NFC specification, several Listen F parameters exist to set up T3T with NCI commands.

Table 92. Values to configure the T3T card emulation

ID	Length	Values and description
LF_T3T_MAX	1 byte	0 – 16, defines the maximum amount of LF_T3T_IDENTIFIERS supported by the NFCC. PN7160 supports four maximum.
LF_T3T_IDENTIFIERS_1 - 4	10 bytes	byte 0 and 1 define the SC to be used by the T3T on DH. byte 2 – 0 define the NFCID2 the T3T on DH will use.

10.2.3 Access through the Frame RF Interface

The Frame RF interface allows the DH to emulate a T3T, assuming that the DH is able to manage the T3T protocol on its own.

Here are the commands and configuration parameters to prepare the T3T Card Emulation for technology NFC-F through the Frame RF Interface:

Table 93. Configuration sequence for T3T/NFC-F Card Emulation over Frame RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD *	RF Protocol	PROTOCOL_T3T
	Mode	Listen
	RF Interface	Frame
CORE_SET_CONFIG_CMD	LF_T3T_IDENTIFIERS_X	See above, used to set SC, NFCID2
	Protocol-based routing	PROTOCOL_T3T routed to DH-NFCEE
RF_DISCOVER_CMD	RF Technology & Mode	NFC_F_PASSIVE_LISTEN_MODE

* Note: RF_DISCOVER_MAP_CMD is optional since the mapping to Frame RF Interface is done by default.

11 Poll and Listen sides: P2P Initiator and Target Mode

[NCI] defines all the mechanisms necessary to implement this feature. Two options are possible:

1. The DH leaves the NFC-DEP protocol management to the NFCC: it SHALL then map the NFC-DEP protocol on the NFC-DEP interface.
2. The DH wants to manage by itself the NFC-DEP protocol, it SHALL then map the NFC-DEP protocol on the Frame RF Interface.

11.1 NFC-DEP RF interface

11.1.1 P2P Passive mode

The NFC-DEP RF Interface allows the DH to emulate an NFC-DEP Target or Initiator in P2P Passive, leaving up to the PN7160 to manage the NFC-DEP protocol.

Here are the commands and configuration parameters to prepare the NFC-DEP Target in P2P Passive, for technologies NFC-A and NFC-F, through the NFC-DEP RF Interface:

Table 94. Configuration sequence for NFC-DEP/NFC-A&F Passive Target over NFC-DEP RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_NFC-DEP
	Mode	Listen
	RF Interface	NFC-DEP
CORE_SET_CONFIG_CMD	LA_BIT_FRAME_SDD	
	LA_PLATFORM_CONFIG	
	LA_SEL_INFO	
	LA_NFCID1	
	LF_CON_BITR_F	
	LF_PROTOCOL_TYPE	
	LN_WT	
	LF_ADV_FEAT[1]	
	LN_ATR_RES_GEN_BYTES	
	LN_ATR_RES_CONFIG	
	NFCDEP_OP	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_PASSIVE_LISTEN_MODE
	RF Technology & Mode	NFC_F_PASSIVE_LISTEN_MODE

/!\	[1] this parameter is not supported in [PN7160_Ref]

Here are the commands and configuration parameters to prepare the NFC-DEP Initiator in P2P Passive, for technologies NFC-A and NFC-F, through the NFC-DEP RF Interface:

Table 95. Configuration sequence for NFC-DEP/NFC-A&F Passive Initiator over NFC-DEP RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_NFC-DEP
	Mode	Poll
	RF Interface	NFC-DEP
CORE_SET_CONFIG_CMD	PA_BAIL_OUT[1]	
	PF_BIT_RATE	
	PF_RC_CODE	
	PN_NFC_DEP_SPEED	
	PN_ATR_REQ_GEN_BYTES	
	PN_ATR_REQ_CONFIG	
	NFCDEP_OP	
RF_DISCOVER_CMD	RF Technology & Mode	NFC_A_PASSIVE_POLL_MODE
	RF Technology & Mode	NFC_F_PASSIVE_POLL_MODE

11.1.2 P2P Active mode

The NFC-DEP RF interface allows the DH to emulate an NFC-DEP Target or Initiator in P2P Active, leaving up to the the PN7160 to manage the NFC-DEP protocol.

Here are the commands and configuration parameters to prepare the NFC-DEP Target in P2P Active, for technologies NFC-A and NFC-F, through the NFC-DEP RF Interface:

Table 96. Configuration sequence for NFC-DEP/NFC-A&F Active Target over NFC-DEP RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_NFC-DEP
	Mode	Listen
	RF Interface	NFC-DEP
CORE_SET_CONFIG_CMD	LA_BIT_FRAME_SDD	
	LA_PLATFORM_CONFIG	
	LA_SEL_INFO	
	LA_NFCID1	
	LF_CON_BITR_F	
	LF_PROTOCOL_TYPE	
	LN_WT	
	LN_ATR_RES_GEN_BYTES	
	LN_ATR_RES_CONFIG	
RF_DISCOVER_CMD	RF Technology and Mode	NFC_A_ACTIVE_LISTEN_MODE
	RF Technology and Mode	NFC_F_ACTIVE_LISTEN_MODE (NCI1.0 only)

Here are the commands and configuration parameters to prepare the NFC-DEP Initiator for technologies NFC-A and NFC-F through the NFC-DEP RF Interface:

Table 97. Configuration sequence for NFC-DEP/NFC-A&F Active Initiator over NFC-DEP RF Interface

Command	Main Parameters	Values
RF_DISCOVER_MAP_CMD	RF Protocol	PROTOCOL_NFC-DEP
	Mode	Poll
	RF Interface	NFC-DEP
CORE_SET_CONFIG_CMD	PA_BAIL_OUT	
	PF_BIT_RATE	NCI1.0 only
	PN_NFC_DEP_SPEED PN_NFC_DEP_PSL	NCI1.0 only NCI2.0 only
	PN_ATR_REQ_GEN_BYTES	
	PN_ATR_REQ_CONFIG	
	PACM_BIT_RATE	Default is 0x01 for 212 Kbps (NCI2.0 only)
RF_DISCOVER_CMD	RF Technology and Mode	NFC_A_ACTIVE_POLL_MODE
	RF Technology and Mode	NFC_F_ACTIVE_POLL_MODE (NCI1.0 only)

In NCI2.0, RF technology and mode NFC_F_ACTIVATE_POLL_MODE are not supported. It is replaced by the parameter PACM_BIT_RATE which define the activation bit rate. The parameter PF_BIT_RATE is only used in NCI1.0 in order to determine the activation bit rate in technology F. For NCI2.0, it is replaced by the PACM_BIT_RATE parameter.

/!\	In NCI2.0, do not use NFC_F_ACTIVE_POLL_MODE neither PF_BIT_RATE but PACM_BIT_RATE

11.1.3 LLCP Symmetry RF Interface Extension

LLCP enables to set up an asynchronous balance mode communication between two NFC-DEP peer to peer endpoints. Therefore, after a specific timeout, an LLCP symmetric packet is sent by the initiator even if there is no data to send. The target can respond with data frames or with an LLCP symmetric packet if there is no data to be sent. After receiving an LLCP symmetric packet or a data packet, the initiator will itself send a data packet or send an LLCP symmetric packet, and so one.

This mechanism is by default supported by the DH, but since NCI2.0, the LLCP symmetry handling can be managed directly by the NFCC on request of the DH. The advantage consists on removing the exchange of LLCP symmetric packet between the DH and the NFCC.

Pre-requisite: the LLCP Symmetry RF interface extension is only available after activation of the NFC-DEP RF Interface. Here is the command with the parameters to activate the LLCP symmetry mechanism in P2P mode:

Table 98. RF_INTF_EXT_START_CMD

GID	OID	Main Parameters	Values
0001b	0x0A	Extension	LLCP Symmetry RF Interface Extension
		Start parameter length	2
		Start parameter	1 byte LLCP-Low Remote Link Timeout 1 byte LLCP-Low Local Link Timeout

The LLCP symmetry mechanism can be stopped and the DH can hand over the LLCP symmetric packet handling with the RF_INTF_EXT_STOP_CMD command.

Table 99. RF_ INTF_EXT_STOP_CMD

GID	OID	Main Parameters	Values
0001b	0x0B	Extension	LLCP Symmetry RF Interface Extension
		Start parameter length	0

11.1.4 Presence check command

As already described in previous chapter, the PN7160 comes with a proprietary function to allow the DH knowing if the Tag/Card is still present or not. The command description in →Section 9.3.3 also applies in Initiator mode (Active or Passive).

11.1.5 WTX notification

As already described in previous chapter, the PN7160 comes with a proprietary notification WTX which indicates that peers are in phase of exchanging RTOX REQ/RESP (NFC DEP equivalent of WTX in ISO DEP) for the configured period of time. The notification description in →Section 9.3.5 also applies in Initiator mode (Active or Passive).

11.2 RF Frame Interface

/!\	Not supported in PN7160

12 RF Discovery (Polling Loop) Management

12.1 RF Discovery functionalities

This contains the overall RF Discovery concepts applied in PN7160. [NCI] defines the general RF state machine allowing the NFC controller to discover either cards or readers or peers. This RF state machine contains a state called RFST_DISCOVERY where the RF Discovery profile is applied.

In order to ensure standard compliance, the PN7160 supports 2 different RF discovery profiles:

- NFC Forum profile: implementation of the NFC Forum polling activity,
- either limited to the current technologies defined in this standardization body (NFC-A, NFC-B, NFC-F, P2P passive).
- Or extended with the additional technologies supported by PN7160, i.e. P2P Active and 15693. PN7160 also offers the possibility to extend this profile by polling for both NFC-F 424 and NFC-F 212.
- EMVCo profile: mode allowing the PN7160 to be compliant to the EMVCo polling activity.

In addition to these RF profiles, the PN7160 offers a way to limit the power consumption by applying a tag detector concept. The tag detector can be seen as a precondition to enable a dedicated profile. It means that if the tag detector is triggered, the default profile is automatically started.

Note that [NCI] defines the TOTAL_DURATION of the discovery period independently of the reader phases applied. To simplify the implementation, for the PN7160 it has been decided to apply a timer only during the Listen/pause phase. So depending on the polling phase configuration (1 technology or more), the total duration will vary a bit. This is considered as acceptable and agreed by the NCI task Force in the NFC Forum.

12.1.1 RF Discovery State Machine

The following drawing shows the PN7160-NCI RF state machine, which differs from [NCI] only by the additions in red. Here are these additions:

- A loop-back transition on state RFST_POLL_ACTIVE, corresponding to the RF_PRES_CHECK_CMD which can be sent by the DH to know if the Card/PICC is still in the field. See the command description in chapter →Section 9.3.3.
- A new status code used on the CORE_GENERIC_ERROR_NTF loop-back transition on state RFST_DISCOVERY: this new status code is used when PN7160 is configured to behave as an EMVCo PCD, and it detects collision. See →Section 12.5.1 for more details.
- A new transition from RFST_POLL_ACTIVE to RFST_DISCOVERY: this transition is triggered by PN7160, when it is configured to behave as an EMVCo PCD and it detects that the RF communication with the PICC is broken.

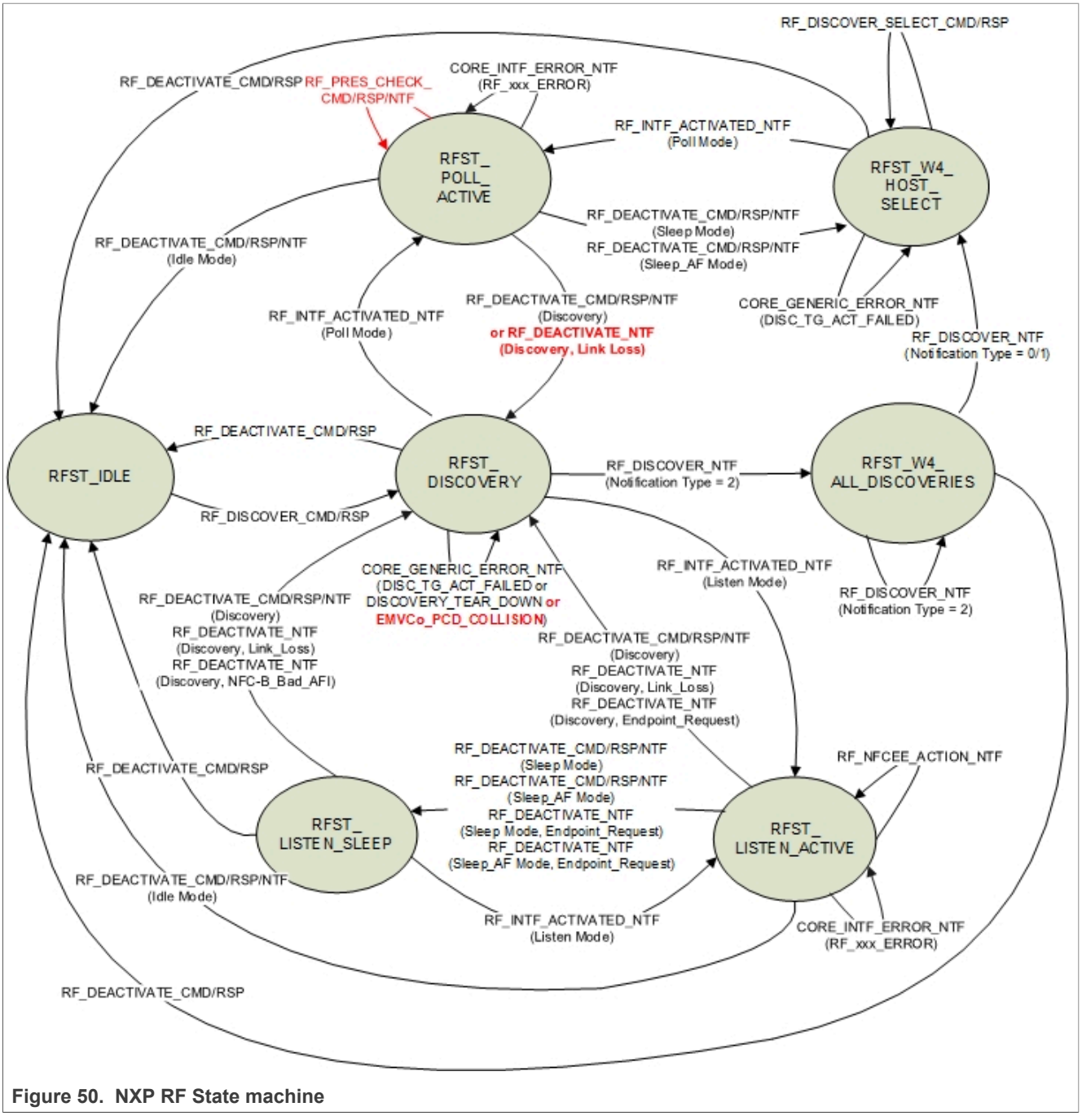

Figure 50. NXP RF State machine

Since the [NCI] RF State Machine is quite complex, it is presented slightly differently in Annex A of the present document: the State Machine is drawn depending on the RF interface to be used. See chapter →Section 17 for further details.

/!\	Since [PN7160_Ref] does not support Listen Mode using the Frame RF Interface, it does not accept the RF_DEACTIVATE_CMD(Sleep Mode) or RF_DEACTIVATE_CMD(Discovery) in RFST_LISTEN_ACTIVE or RFST_LISTEN_SLEEP.

12.2 NFC Forum profile as defined in [NCI]

The NFC Forum profile is the implementation of the RF discovery activity as defined in the NFC Forum (see [ACTIVITY] specification).

[NCI] only covers technologies NFC-A, NFC-B and NFC-F. So the basic NFC Forum profile will poll for these technologies only. Furthermore, for NFC-F, only one-bit rate is used during the polling phase. This is configured thanks to the “Poll F parameter” PF_BIT_RATE as defined in [NCI], section 6.1.4. So the DH configures if NFC-F is polled at 212 kb/s or at 424 kb/s, before it activates the discovery by sending the RF_DISCOVER_CMD command.

The figure bellow represents the profile defined by the NFC Forum, assuming that the DH has enabled the 3 technologies currently supported by the NFC Forum (NFC-A, NFC-B, NFC-F) in Poll mode and Listen mode. To do so, it has to send the following command:

```
RF_DISCOVER_CMD( 6,
[NCI_DISCOVERY_TYPE_POLL_A_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_B_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_F_PASSIVE,1],
[NCI_DISCOVERY_TYPE_LISTEN_A_PASSIVE,1],
[NCI_DISCOVERY_TYPE_LISTEN_B_PASSIVE,1],
[NCI_DISCOVERY_TYPE_LISTEN_F_PASSIVE,1] )
```

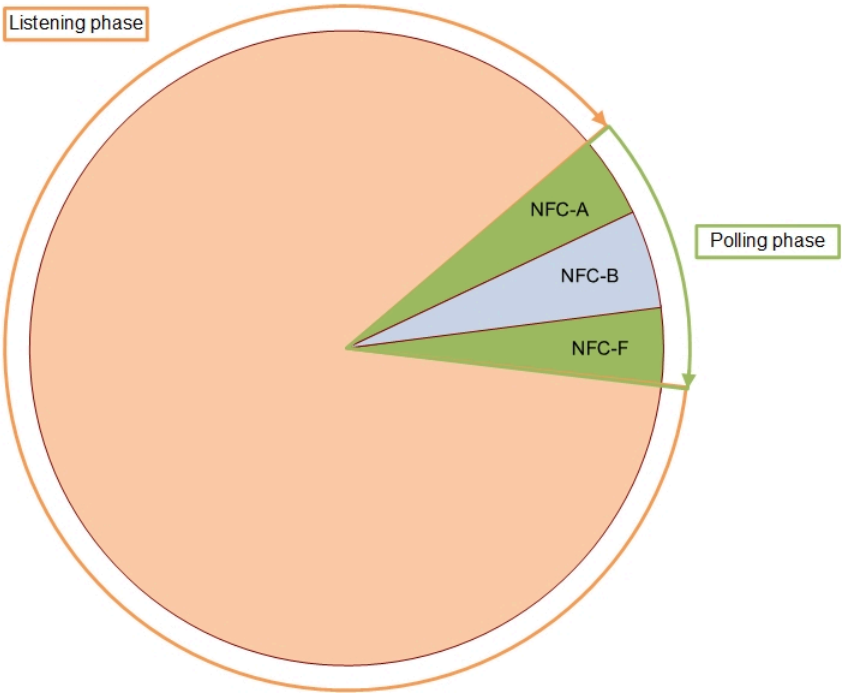

Figure 51. RF Discovery NFC Forum profile (when there is no card/tag in the field)

12.3 [PN7160-NCI] extension: additional technologies not yet supported by the NFC Forum

PN7160 supports more technologies than currently supported by the NFC Forum specifications: P2P Active and 15693.

Furthermore, PN7160 offers an additional proprietary value for the configuration parameter PF_BIT_RATE, which allows configuring that both 212 kb/s and 424 kb/s are polled for NFC-F in Passive Mode.

Thanks to the RF_DISCOVER_CMD and the PF_BIT_RATE, the DH has full flexibility to extend the default RF Discovery profile as currently defined in the NCI 1.0 specification. Here is an example how the DH can enable all technologies available in PN7160, for both Poll & Listen Mode:

1. The DH sets PF_BIT_RATE to 0x08, such that the PN7160 polls for 212 and 424 kb/s in technology F PASSIVE.

```
CORE_SET_CONFIG_CMD( NbrParam = 0x01,
ID = 0x18,
Length = 0x01,
Val = 0x08 )
```

1. The DH enables all technologies and modes available in PN7160:

```
RF_DISCOVER_CMD( 8,
[NCI_DISCOVERY_TYPE_POLL_A_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_B_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_F_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_15693_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_A_ACTIVE*,1],
[NCI_DISCOVERY_TYPE_LISTEN_A_PASSIVE,1],
[NCI_DISCOVERY_TYPE_LISTEN_B_PASSIVE,1],
[NCI_DISCOVERY_TYPE_LISTEN_F_PASSIVE,1] )
```

** NCI_DISCOVERY_TYPE_POLL_F_ACTIVE is not allowed with the [PN7160_Ref], see →Section 7.3.4.*

The resulting RF discovery is drawn below:

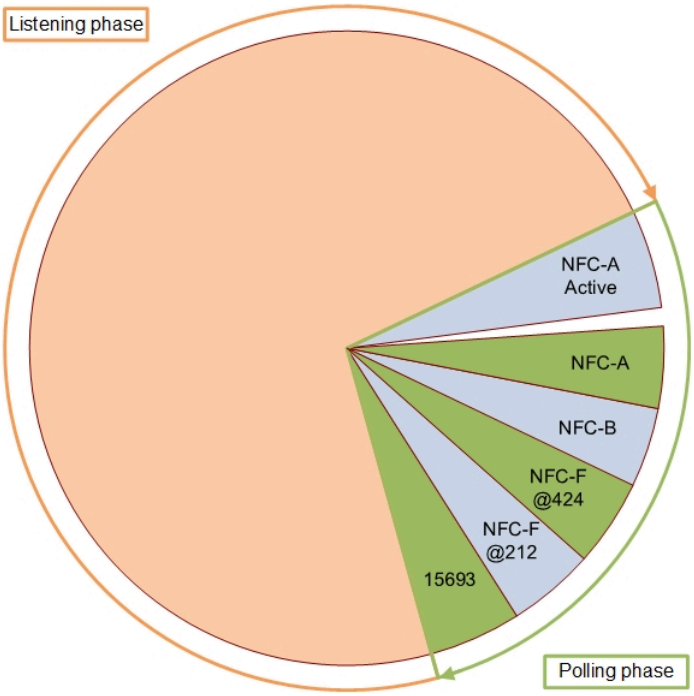

Figure 52. NFC Forum+ profile (when there is no card/tag in the field)

Note: the transition from the Poll NFC-A Active phase to the Poll NFC-A (passive) is done through an RF field OFF/ON sequence.

For more details concerning the different phases duration, guard time, Bail-out, please refer to the configuration section (chapter →Section 13.1) where all these parameters are defined.

12.4 [PN7160-NCI] extension: Low Power Card Detector (LPCD) Mode

12.4.1 Description

The Low-Power Card Detector is an NXP proprietary extension, which is applied in case the DH wants to reduce the power consumption.

The concept is to avoid using the Technology Detection Activity as defined in [ACTIVITY], which implies to generate an RF Field for several tens of milliseconds and to send technology-specific request commands to see if there is a Card/Tag in the field to respond. The more technologies the PN7160 is configured to detect, the longer the RF Field is generated and the higher the current consumption.

The LPCD is based on another concept, which only relies on the antenna characteristics, not on valid responses from a Card/Tag. Indeed, the antenna impedance is influenced by the Card/tag which may enter into its proximity, due to the magnetic coupling between the 2 antennas. The LPCD is therefore monitoring the antenna impedance, to see if there is a significant variation which is interpreted as being caused by a Card/Tag being in proximity.

To achieve that, the LPCD periodically generates very short pulses of RF Field, without any modulation, and measures some antenna characteristics during this pulse. The time between these RF pulses is defined by the TOTAL_DURATION parameter, as specified for the RF Discovery in [NCI].

When a Card/Tag enters the field, there is an antenna impedance variation. If this variation is higher than a pre-defined threshold, the default Technology Detection profile (NFC Forum, or EMVCo, see next sections) is automatically started. The PN7160 is then sending technology-specific request commands, expecting a response since the LPCD detected a change on the antenna impedance.

Note: the LPCD may also be triggered by a metal object, which can influence the Antenna impedance in a similar way as a Card/Tag. The PN7160 will anyhow detect that this object is not a contactless device since it immediately starts sending contactless commands to check if a Card/Tag can respond.

The Low-Power Card Detector is configured and enabled/disabled thanks to a specific configuration parameter (TAG_DETECTOR_CFG) described in →Section 13.2.1. The threshold is also defined by an additional configuration parameter (TAG_DETECTOR_THRESHOLD_CFG) described in the same section.

The figure below describes the RF Discovery when the LPCD is enabled:

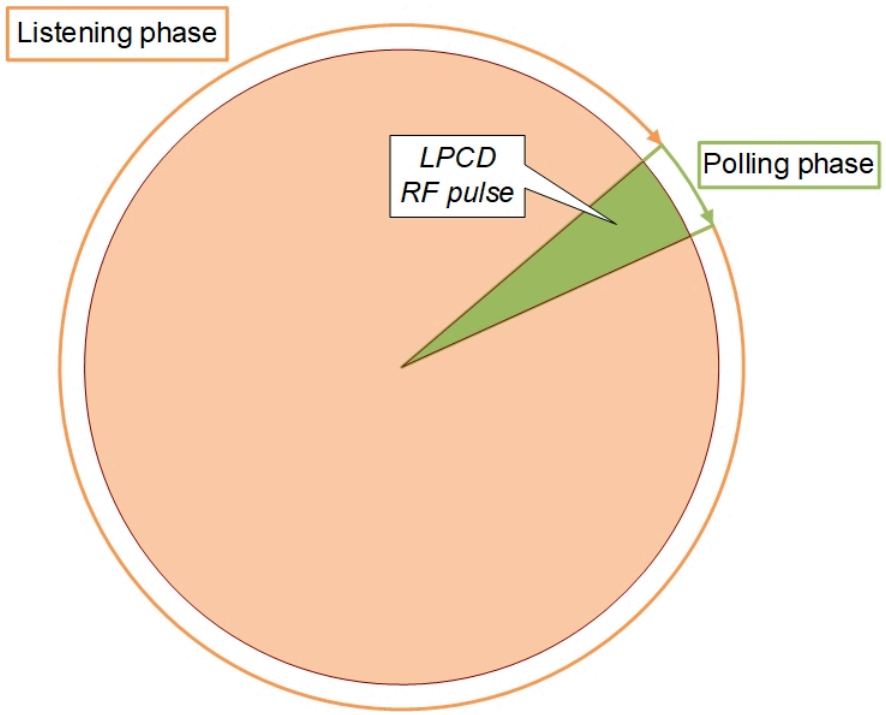

Figure 53. Low-Power Card Detector mode (when there is no card/tag in the field)

The figure below compares the RF Discovery with the LPCD disabled to the RF Discovery with the LPCD enabled and highlights the impact on the average current consumption (the assumption being here that TOTAL_DURATION ~ 300 ms):

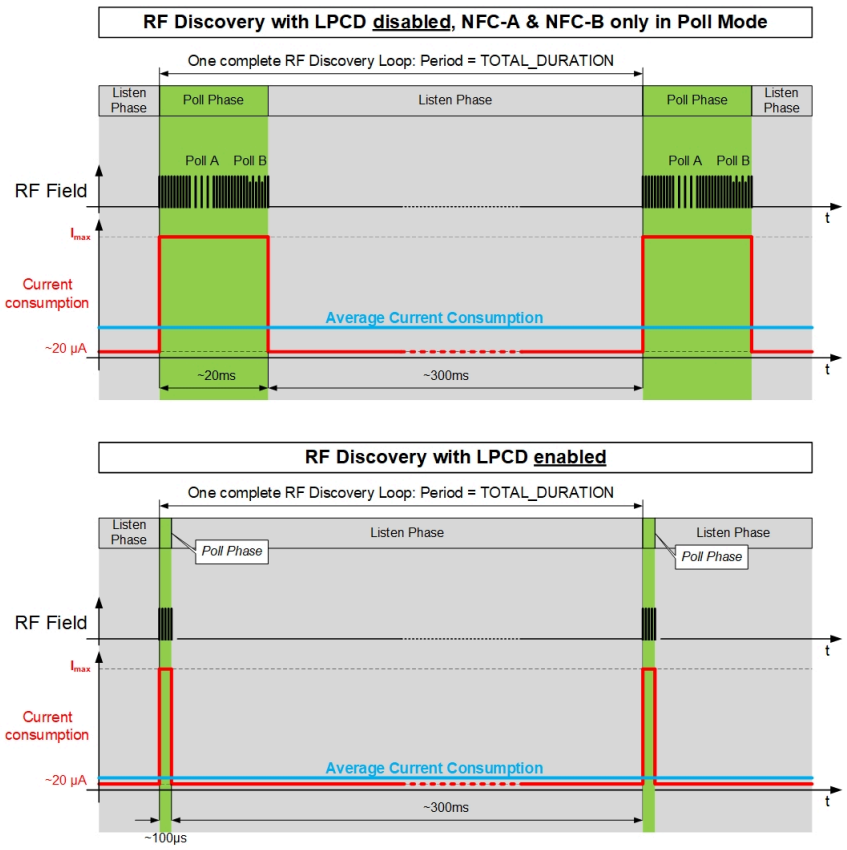

Figure 54. Comparison of the RF Discovery with the LPCD disabled or enabled

12.4.2 Configuration of the Technology Detection Activity when the LPCD has detected an "object"

As described in the previous chapter, once the PN7160 detects a change in the antenna impedance, it performs a Technology Detection as defined in [ACTIVITY] which tries to activate the “object” by sending Request Commands from the different technologies configured for the RF Discovery.

In order to improve the likelihood to catch such a Card/Tag, the PN7160 comes with a retry mechanism which performs several Technology Detection polling cycles before it switches back to LPCD.

During this retry mechanism, a temporary period is used, called **TechDet_PERIOD** which is specified in steps of 10 ms. The number of the retry cycles can also be configured thanks to the **TechDet_NBR_RETRIES** parameter.

Table 100. Parameters used to configure the overall period of the RF Discovery:

LPCD Status	Period between 2 consecutive Technology Detections	Period between 2 consecutive LPCD RF pulses
Enabled	TechDet_PERIOD	TOTAL_DURATION
Disabled	TOTAL_DURATION	Not applicable

The next figure illustrates how these 3 parameters **TOTAL_DURATION**, **TechDet_PERIOD** and **TechDet_NBR_RETRIES** influence the Low-Power Card Detector and the RF Discovery:

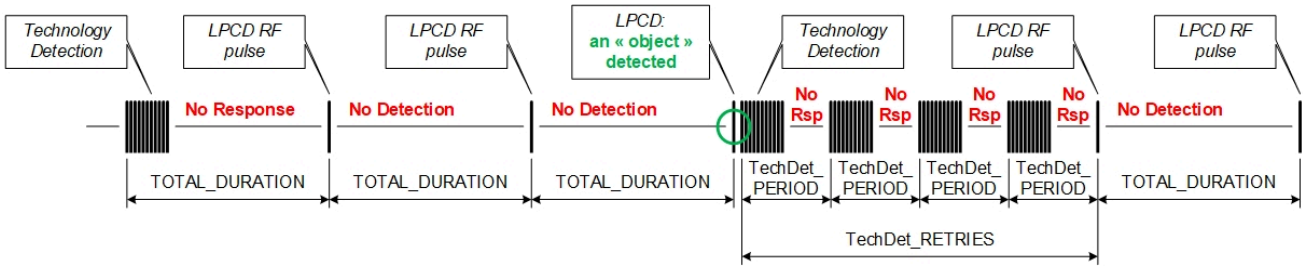

Figure 55. Illustration of the Low-Power Card detector and the subsequent Technology Detection cycles

See →Section 13.2.1 for the description of the configuration parameter called " TechDet_AFTER_LPCD_CFG " which contains the 2 parameters **TechDet_PERIOD** and **TechDet_NBR_RETRIES**.

12.4.3 Notification when the Trace Mode is enabled

The Low-Power Card Detector needs to be tuned in each application; it is therefore useful to get some information from PN7160 so that the Low-Power Card Detector can be appropriately configured.

The Low-Power Card Detector can be configured to enable a Trace Mode, where the following Notification will be sent to the DH by PN7160:

Table 101. RF_LPCD_TRACE_NTF

GID	OID	Numbers of parameter(s)	Description
1111b	0x13	2	PN7160 sends the actual measurement + the threshold

Table 102. RF_LPCD_TRACE_NTF parameters

Payload Field(s)	Length	Value/Description
Reference Value	2 bytes	Reference Value used by Low-Power Card Detector function to compare with the measurement value. Coding is little Endian. Higher bit (bit 15) is RFU, its value shall not be considered.
Measurement Value	2 bytes	Value measured on the AGC. Coding is little Endian.

12.5 [PN7160-NCI] extension: EMVCo Profile in Poll & Listen Modes

The EMVCo profiles are introduced in PN7160 for EMVCo compliancy. Indeed there are incompatibilities between the RF Discovery activity as defined in the NFC Forum and the RF discovery defined in EMVCo standard.

12.5.1 EMVCo profile in Poll Mode

12.5.1.1 Configuring PN7160 to implement the EMVCo polling algorithm

To be compliant to the EMVCo certification tests, the RF Discovery has to be configured so that only NFC-A and NFC-B are supported in Poll phase and so that there is no Listen phase. So the DH has to send the following command:

```
RF_DISCOVER_CMD( 2,
[NCI_DISCOVERY_TYPE_POLL_A_PASSIVE,1],
[NCI_DISCOVERY_TYPE_POLL_B_PASSIVE,1])
```

In addition, PN7160 needs to be aware of the fact that it has to behave according to the EMVCo RF discovery, not according to the NFC Forum RF discovery based on [ACTIVITY].

A specific configuration parameter is defined for that purpose. It allows selecting which profile is active during the RF discovery in Poll Mode. This parameter (POLL_PROFILE_SEL_CFG) is detailed in section Section 13.2.1.

When this parameter is set to 0x01, PN7160 implements a specific discovery algorithm, compliant to the EMVCo standard. The target is to ensure that there is one single card in the field. So PN7160 has to detect any collision inside 1 technology (NFC-A or NFC-B) or to detect if there are multiple cards based on different technologies (i.e. 1 card in NFC-A and 1 card in NFC-B).

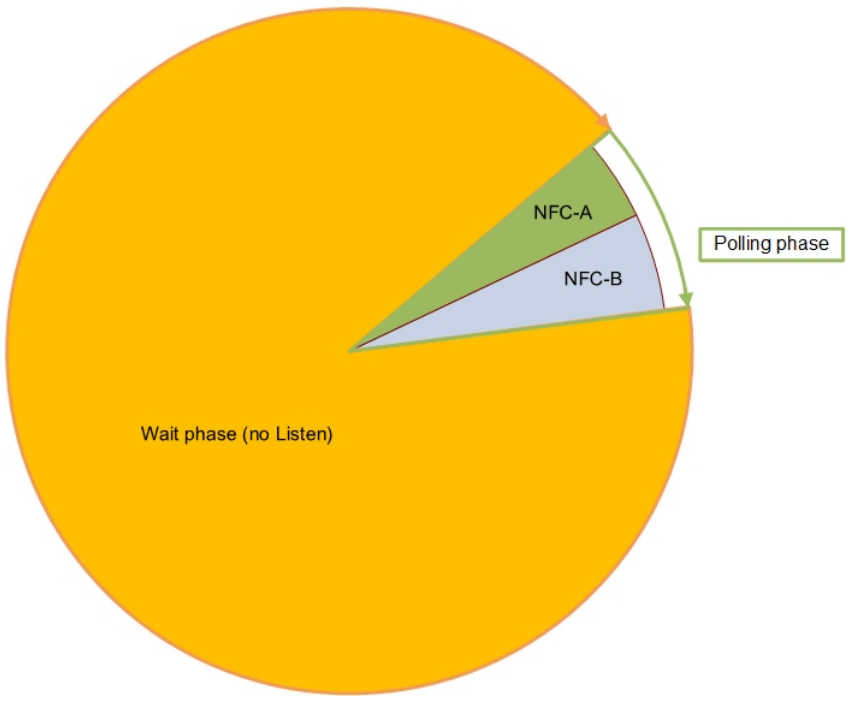

Figure 56. EMVCo profile (when there is no card in the field)

If there is a card detected in the field, then the polling sequence is modified by the PN7160, in order to look for another potential card in the field.

This is illustrated by the 2 figures below:

- On the 1st one, there is no card in the RF Field, so PN7160 keeps polling by alternating WUPA and WUPB commands.

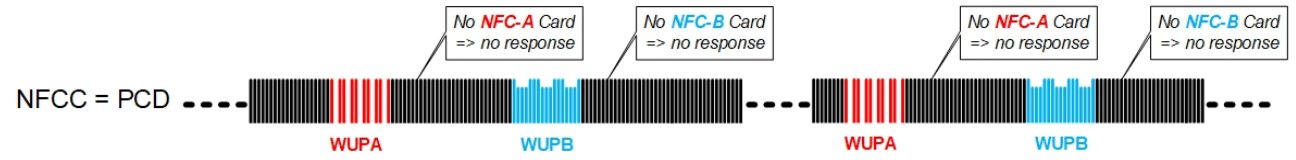

Figure 57. EMVCo polling without a card in the field

- On the 2nd one, an NFC-A card is placed in the RF Field. The PN7160 detects it, activates it and puts it in HALT state and then looks for a potential NFC-B card in the field. Since there is no NFC-B card in the field, the PN7160 activates the NFC-A card again, then the PN7160 activates the ISO-DEP interface and the DH can start to exchange data with the NFC-A card to proceed with the payment application.

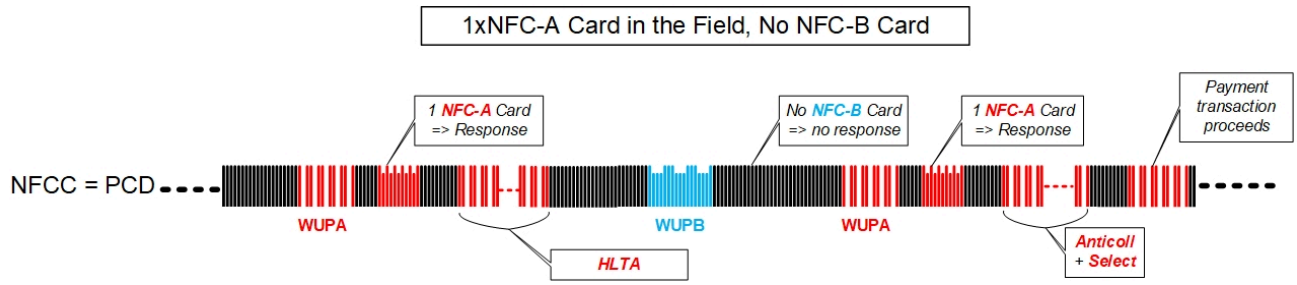

Figure 58. EMVCo polling with an NFC-A card in the field

/!\	In [PN7160_Ref], the Low Power Card Detector is automatically disabled when the EMVCo profile is enabled, since these 2 features are conflicting if simultaneously enabled.

12.5.1.2 Notification for RF technology collision

When the EMVCo profile for Poll Mode is activated, PN7160 will activate the ISO-DEP RF Interface through RF_INTF_ACTIVATED_NTF only when there is 1 single card in the field, whatever the technology (NFC-A or NFC-B).

When the EMVCo profile for Poll Mode is activated and PN7160 detects a collision on RF (either in one technology or between technologies), it will report a special Status in the CORE_GENERIC_ERROR_NTF: STATUS_EMVCo_PCD_COLLISION. The current state will remain RFST_DISCOVERY, as graphically described in →Section 12.1.1. The identifier of this proprietary Status is defined in →Section 7.4.8.Note that if the cards remain in the RF Field, PN7160 will keep sending the CORE_GENERIC_ERROR_NTF(STATUS_EMVCo_PCD_COLLISION) at each polling loop: this can be used as a presence check mechanism.

When the EMVCo profile for Poll Mode is activated and PN7160 has detected a single PICC (ie no collision) but it is unable to properly activate this PICC, then PN7160 will send a CORE_GENERIC_ERROR_NTF(DISCOVERY_TARGET_ACTIVATION_FAILED) as defined in [NCI].

12.5.1.3 Modification of the NCI RF State Machine in case of failure during data exchange

When the EMVCo profile for Poll Mode is activated, the NFCC has to comply with tight timings verified during the EMVCo PCD certification. In case the RF link with the PICC is broken, the regular way to behave according to NCI is that the NFCC will detect a time-out or an unrecoverable protocol error and send then a CORE_INTERFACE_ERROR_NTF with the appropriate status. It is then up to the DH to stop the RF Discovery with RF_DEACTIVATE_CMD(IDLE) and to restart the RF Discovery with RF_DISCOVER_CMD. Unfortunately the time required to execute this sequence is highly dependent on the DH latency and it is often not possible to match the timings expected and checked by the EMVCo PCD certification.

To solve this issue, NXP has decided to add a transition from the RFST_POLL_ACTIVE to RFST_DISCOVERY, triggered by the sending of the RF_DEACTIVATE_NTF(Discovery, Link Loss). In such a way, when PN7160 has detected a timeout or an unrecoverable protocol error during the RF communication with the PICC, it will autonomously come back to RFST_DISCOVERY, switching OFF the RF Field, as requested by EMVCo and then restarting the Polling phase in a timely manner, as requested by EMVCo.

This new transition is graphically described in →Section 12.1.1.

12.5.2 EMVCo profile in Listen Mode

12.5.2.1 Configuring PN7160 to behave as a single EMVCo card in Listen

To be compliant to the EMVCo certification tests emulating an EMVCo PICC, PN7160 has to behave as a single PICC based on either technology NFC-A or NFC-B.

This requirement might be in conflict with the configuration applied by an NFCEE over SWP. Indeed, the NFCEE may configure PN7160 to behave as a Card Emulator for both technologies NFC-A and NFC-B. In that case, the PCD will detect 2 PICCs sequentially (PN7160 will answer to the REQA and later on to the REQB). The EMVCo PCD will then abort the transaction.

In order to solve this issue, PN7160 comes with a specific configuration parameter: LISTEN_PROFILE_SEL_CFG, detailed in section Section 13.2.2.

Thanks to this parameter, a specific EMVCo PICC profile can be activated such that PN7160 will “hide” the non-yet-selected technology to the EMVCo PCD. Once this parameter is activated, the PICC selection sequence is as follows (assuming NFC-A is selected first):

- Once NFC-A has been selected by the PCD through the REQA command, PN7160 disables the NFC-B card emulation so that the REQB command sent later on by the EMVCo PCD gets no answer.
- The payment transaction can then successfully go through based on technology NFC-A.
- PN7160 waits then for an RF Field OFF/Field ON sequence before enabling the non-selected technology (NFC-B) again.

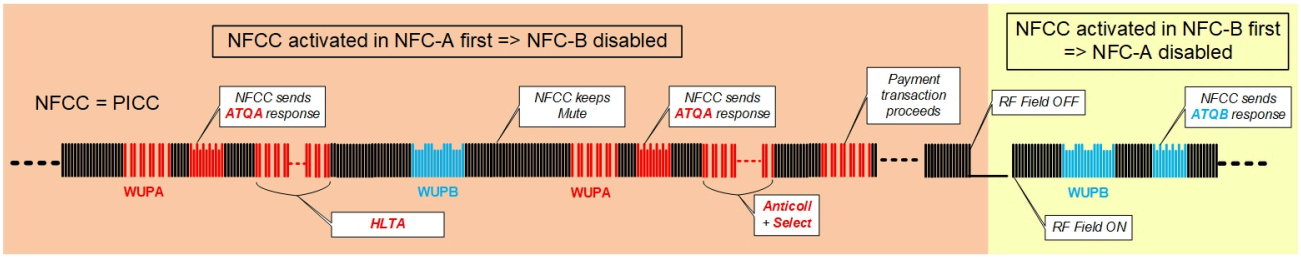

Figure 59. EMVCo Listening with an NFC-A activated first by the PCD and then NFC-B activated first, after Field OFF/ON sequence

12.6 [PN7160-NCI] extension: Power optimization

PN7160 offers a standby mode, which can be activated together with the RF Discovery, such that the overall power consumption is significantly reduced.

One dedicated proprietary function is added to enable/disable this standby mode: CORE_SET_POWER_MODE.

12.6.1 CORE_SET_POWER_MODE Command/Response

/!\	The Standby mode is enabled by default in [PN7160_Ref]. Given the very strong impact on the power consumption, disabling the Standby mode should be restricted to debug sessions.

Table 103. CORE_SET_POWER_MODE_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x00	1	Command to request the PN7160 to enable/disable the Standby mode

Table 104. CORE_SET_POWER_MODE_CMD parameter

Payload Field(s)	Length	Value/Description
Mode	1 byte	0x00: Standby mode disabled 0x01: Standby mode enabled 0x02 - 0xFF: RFU

Table 105. CORE_SET_POWER_MODE_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x00	1	Response to inform the DH of the status of the CORE_SET_POWER_MODE_CMD

Table 106. CORE_SET_POWER_MODE_RSP parameter

Payload Field(s)	Length	Value/Description
Status	1 byte	0x00: STATUS_OK 0x06: STATUS_SEMANTIC_ERROR 0x09: STATUS_INVALID_PARAM Others: RFU

12.6.2 Standby wake-up

The PN7160 wakes-up from standby when one of the following events occurs:

- Regular polling-loop starts. When the DH has served the PN7160 with a NCI_RF_DISCOVER_CMD command, the PN7160 enters into the standby mode and automatically leaves the low-power mode after the period defined by TOTAL_DURATION.
- RF level detector triggered. An external field has been introduced in the NFC volume during the standby period of the polling loop and at least one listen phase has been requested by the NCI_DISCOVER_CMD.
- Host interface activity detected as depicted in →Section 6.2.3.

Remark: One can prevent the PN7160 to ever enter standby by asserting the wake-up pin WUP_REQ. As long as it is high, the PN7160 will not be able to enter standby.

12.7 [PN7160-NCI] extension: Management of the "Screen Off/Locked" modes

It is very common in phones or tablets implementation to change the RF Discovery settings when the screen of the device is turned On, Off or locked.

Indeed, if it makes sense to have both Poll and Listen Modes enabled at the same time when the screen is On, it also makes sense to disable the Poll Mode when the screen is Off, since in that case the Device is trying to optimize the power consumption. A further optimization is also to disable the P2P target and the Card Emulation hosted in the DH, since when the Screen is OFF, the DH is usually in sleep mode and cannot react on time in case an external P2P initiator or a PCD would like to start a contactless transaction.

Apart from the power consumption aspects, it is also commonly required to define the Card Emulation Routing in a specific way for these "screen Off" or "screen Locked" modes (payment applications are typically disabled in such as power modes).

Therefore, [PN7160_Ref] comes with 2 proprietary power modes, called "Screen Off" and "Screen Locked", in addition to the existing power modes defined in [NCI] (Switched On, Switched Off, Battery Off). See →Section 7.4.11 for the definition of the extended value field of the power modes.

The DH can then configure the Listen Routing Table in a specific way for the " Screen off" or "Screen Locked" power modes. It can especially ensure that neither the DH P2P target nor the DH Card Emulation modes are enabled, if required.

A specific proprietary command has been defined for the DH to inform the [PN7160_Ref] when the DH is moving from " Screen On" to " Screen Off" or " Screen Locked" or vice-versa:

Table 107. SCREEN_STATE_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x15	1	The DH informs the NFCC that the Screen Mode has changed: either On or Off

Table 108. SCREEN_STATE_CMD parameter

Payload Field(s)	Length	Value/Description
Screen_State	1 byte	0x00: The Screen is turned On 0x01 The Screen is turned Off 0x02 The Screen is locked 0x03-0xFF: RFU

/!\	SCREEN_STATE_CMD SHALL NOT be sent by the DH if the NFCC is not in RFST_IDLE.

Table 109. SCREEN_STATE_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x15	1	The NFCC acknowledges the command received from the DH and gives a status

Table 110. SCREEN_STATE_RSP parameters

Payload Field(s)		Length	Value/Description
Status		1 byte	One of the following Status codes, as defined in [NCI_Table1]: 0x00: STATUS_OK 0x01: STATUS_REJECTED 0x06: STATUS_SEMANTIC_ERROR Others: RFU

When the DH wants to inform the NFCC on a change of the "Screen Off/Locked to Screen On" status and possibly reconfigure the RF Discovery, it has to implement the following sequences:

Indicating that the Device is entering Screen Off/Locked power mode and stopping any Polling during RF Discovery:

- RF_DEACTIVATE_CMD(Idle)
- SCREEN_STATE_CMD(Screen_State = '1/2')
- RF_DISCOVER_CMD(any technology, in Listen Mode only)

Indicating that the Device is entering Screen On power mode and re-enabling Polling in the RF Discovery:

- RF_DEACTIVATE_CMD(Idle)

- SCREEN _STATE_CMD(Screen_State = '0')
- RF_DISCOVER_CMD(any technology, Poll and Listen Modes)

Here is a description of how the transition from one power mode to another power mode is triggered. It clearly shows that this new power mode "Screen Off/Locked" can only be activated coming from the "Switched On" power mode.

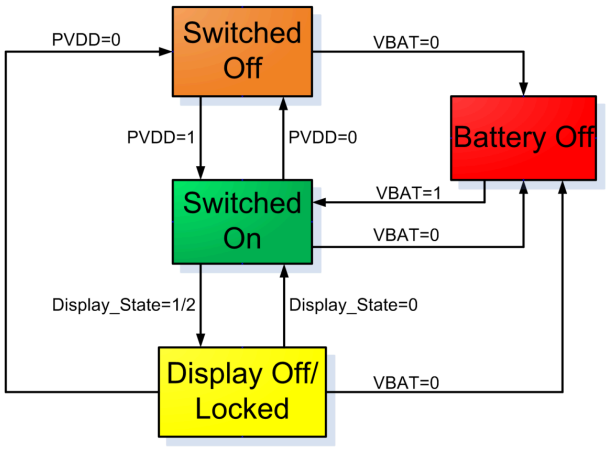

Figure 60. Power Mode transitions

12.8 [PN7160-NCI] extension: Error notifications

PN7160 offers more debug possibilities than previous products, indicating as much as possible by notifications when some RF features cannot be used due to:

- bad configuration
- bad input power
- missing or unstable clock
- bad frames received on RF

Table 111. RF_PLL_UNLOCKED_NTF

GID	OID	Numbers of parameter(s)	Description
0001b	0x21	0	Notification used to indicate that the PLL has been started but could not have been locked. This might be due to a missing or unstable input clock.

Table 112. RF_TXLDO_ERROR_NTF

GID	OID	Numbers of parameter(s)	Description
0001b	0x23	0	Notification used to indicate that TxLdo (RF Transmitter) could not start. This is usually due to a missing or bad power supply on VUP/TVDD or a bad clock/power configuration

13 Configurations

/!\	When the DH needs to update the value of the parameters described hereafter, it shall send a CORE_RESET_CMD/CORE_INIT_CMD sequence after the CORE_SET_CONFIG_CMD, to ensure that the new value is used for the parameters. If numerous parameters are updated thanks to multiple CORE_SET_CONFIG_CMDs, a single CORE_RESET_CMD/ CORE_INIT_CMD sequence is enough after the last CORE_SET_CONFIG_CMD.

/!\	Any CORE_SET_CONFIG_CMD to one of the following parameters or to the [NCI] standard parameters will trigger an EEPROM write cycle. Since the PN7160 EEPROM has a limited number of Erase/Write cycles (300 000), it is highly recommended to only use the CORE_SET_CONFIG_CMD during the NCI initialization sequence.

13.1 [PN7160-NCI] extension: System configurations

PN7160 offers several parameters used to configure the system aspects. System configurations are always written into EEPROM memory as mentioned by the syntax '***RW in E²PROM'*** described for each parameter below. Among these parameters, some them are persistent after a fw download when the syntax ***'Persistent After FwDL'*** is mentioned. The benefit is that they can be written only once, since they will never change later on. The drawback is that their value needs to be carefully checked, since if it is wrong, it will be wrong for ever (a FW upload will not solve an issue due to a wrong value of these parameters). For any other proprietary parameter described in this User Manual, the FW upload will set the parameter back to its default value (see the column "default value'). If this value is not correct for the application, the DH needs to overwrite it at boot.

Table 113. Core configuration parameters

Name and Rights	Description	Ext. Tag	Len.	Default Value
CLOCK_REQUEST_CFG ***RW in E²PROM***	Indicates how the clock is requested to the DH by the PN7160: • 0x00: Clock Request is disabled (Not supported) • 0x01: Hardware-based Clock Request is enabled: CLK_REQ pin set to high when clock requested, otherwise it is set to hi-Z (High Impedance). CLK_REQ pin stays high until the NFCC goes into Standby or is reset. • 0x02-0xFF: RFU	0xA0 0x02	1	0x01

Table 113. Core configuration parameters*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>CLOCK_SEL_ CFG
RW in E²PROM
Persistent After FwDL</td><td>Input Clock selection and configuration for the internal 13.56 MHz CLOCK.
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td>x</td><td></td><td></td><td></td><td>If set to 1b (XTAL mode), A 27.12MHz quartz has to be connected to PN557
If set to 0b (PLL mode), A clean clock signal has to be directly provided on the Clock pad</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td>x</td><td>x</td><td>x</td><td>Any value
(previously used on previous projects to indicate the PLL input frequency: now RFU)</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td></td><td></td><td></td><td>RFU</td></tr>
</table></td><td>0xA0 0x03</td><td>1</td><td>0x08</td></tr>
<tr><td>CLOCK_TO_CFG
RW in E²PROM</td><td>Indicates the timeout value to be used for clock request acknowledgment (from 1.53 ms to 10 ms in steps of 330 µs).
So the actual Time Out value (in µs) is given by the following formula:
TimeOut (µs) = 1200 + (CLOCK_TO_CFG)*330
Value 0x00 SHALL NOT be used, otherwise there is no timeout (no wait time). Minimum value is 01.</td><td>0xA0 0x04</td><td>1</td><td>0x01</td></tr>
<tr><td>IRQ_POLARITY_ CFG
RW in E²PROM
Persistent After FwDL</td><td>Configuration of the IRQ pin polarity
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td>Host Interface wake up with WUP_REQ pin
‘1’ => enabled
‘0’ => disabled</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td><td>IRQ PIN polarity config.</td></tr>
<tr><td>0</td><td>0</td><td></td><td></td><td>0</td><td>0</td><td></td><td>0</td><td>All these bits SHALL be set to logical ‘0’ (RFU)</td></tr>
</table>
b1=’0’ => PN7160 requests to transmit when IRQ pin = ’1’.
b1=’1’ => PN7160 requests to transmit when IRQ pin = ‘0’.</td><td>0xA0 0x05</td><td>1</td><td>0x00</td></tr>
<tr><td>VBAT_MONITOR_ EN_ CFG
RW in E²PROM
Persistent After FwDL</td><td>To Enable/Disable the Battery monitor and configure the Threshold
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td>Vbat Monitor Enable</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td><td>Vbat Monitor Threshold</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td></td><td>RFU</td></tr>
</table>
b0: ‘1’ to Enable, ‘0’ to disable.
b1: ‘1’ to set the threshold to 2.3V and ‘0’ to set it to 2.75V.</td><td>0xA0 0x06</td><td>1</td><td>0x00</td></tr>
</table>

Table 113. Core configuration parameters*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>VEN_CFG
RW in E²PROM
Persistent After FwDL</td><td>Configures the internal VEN signal, in case the VEN pin driver NOT is supplied from PVDD. In such a case, when PVDD is switched OFF, the VEN pin level in unknown, so the internal VEN signal is defined by one bit in an internal register (VEN_Value) while the VEN pin has to be pulled-down (to avoid leakage) thanks to a 2nd bit in the same register (VEN_Pulld) which has then to be set to '1' to activate the Pull Down. These 2 bits can be configured through NCI thanks to VEN_CFG LSbits, according to the following table:
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td>VEN_Value</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td></td><td>VEN_Pulld</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td></td><td>RFU</td></tr>
</table>
Note, in order to force a certain VEN value to be used internally (no matter which state the external VEN pin level is in) the VEN_Pulld value HAS to be set. Only if VEN_Pulld is set and PVDD is switched off the internal VEN state will be forced to what is specified in VEN_Value.</td><td>0xA0
0x07</td><td>1</td><td>0x03</td></tr>
<tr><td>TO_BEFORE_ STDBY_ CFG
RW in E²PROM</td><td>Timeout used to wait after last DH/NFCEE communication before going into standby (from 50ms to 65.536s in steps of 1ms).
Applies only when the discovery is stopped and standby mode is activated by SET_PWR_MODE_CMD.</td><td>0xA0
0x09</td><td>2</td><td>0x03E8
=1000 (1s)</td></tr>
</table>

Table 113. Core configuration parameters*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>PMU_CFG
RW in E²PROM</td><td>Configuration of the Power Management Unit (PMU)
Byte 0: IRQ Enable
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td>1: Enable level IRQ
0: Disable level IRQ</td></tr>
<tr><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td><td>PVDD IRQ</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td>Temp sensor IRQ</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td></td><td>0</td><td>0</td><td>0</td><td></td><td>RFU</td></tr>
</table>
Byte1: RFU
Byte 2 and Byte 3: Power and Clock Configuration per power mode configuration (Byte 2 for device ON, Byte 3 for Device OFF)
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>DC/DC usage in card mode
0b: DCDC bypassed
1b: DCDC is used</td></tr>
<tr><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td>DC/DC usage in reader mode
0b: DCDC bypassed
1b: DCDC is used</td></tr>
<tr><td></td><td></td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td>Clock source:
00b: Default configuration given by CLOCK_SEL_CFG (PLL or XTAL)
11b: DLL RF Clock from external RF reader</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>X</td><td>X</td><td>VUP input voltage:
01b: CFG1
10b: CFG2
Others: RFU</td></tr>
<tr><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td></td><td></td><td>RFU</td></tr>
</table>
Byte 4: RFU
Byte 5: DCDC 0
This register shall be configured when DCDC might be used
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td></td><td>Pulse duration to enable passthrough
100µs per step</td></tr>
<tr><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td></td><td>DC/DC passthrough feature
0b: not supported
1b: supported</td></tr>
<tr><td></td><td></td><td></td><td></td><td>X</td><td></td><td></td><td></td><td>DC/DC usage for LPCD
0b: DCDC is not used
1b: DCDC is used</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>Output pin to drive DC/DC
000b: not connected
010b: TX_PWR_REQ pin
Others: RFU</td></tr>
</table></td><td>0xA0
0x0E</td><td>11</td><td>Byte 0: 0x11
Byte 1: 0x01
Byte 2: 0xC2
Byte 3: 0xB2
Byte 4: 0x00
Byte 5: 0xDA
Byte 6: 0x1E(600µs)
Byte 7: 0x14
Byte8: 0x00
Byte 9: 0xD0
Byte 10: 0x0C</td></tr>
</table>

Table 113. Core configuration parameters*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td></td><td>
Byte 6: DCDC 1
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>Delay to wait for DC/DC reaches its nominal vout in 20us step.</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td></td><td></td><td></td><td></td><td></td><td>RFU</td></tr>
</table>
Byte 7: TXLDO
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td>TxLdo output voltage:</td></tr>
<tr><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td>CFG1 and CFG2 DC/DC bypass
0000b: 3V
0001b: 3.3V
0010b: 3.6V
0011b: 4.5V
0100b: 4.75V
0101b: 5.25V
1000b: 2.7V
1001b: 3.9V
1010b: 4.2V
1011b: 4.7V
1111b: 5.0V
Others: RFU</td></tr>
<tr><td></td><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td>CFG2 except DC/DC bypass
0000b: 3V
0001b: 3.3V
0010b: 3.6V
0011b: 4.5V
0100b: 4.75V
0101b: 5.25V
1000b: 2.7V
1001b: 3.9V
1010b: 4.2V
1011b: 4.7V
1111b: 5.0V
Others: RFU</td></tr>
</table>
NOTE: a transmitter voltage drop around 0,3V (refer to the PN7160 data sheet [PN7160_DS] for accurate value) shall be taken into consideration between external VUP voltage to TxLDO output voltage.
Byte 8: RFU
Byte 9: TXLDO check
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td>VUP voltage check</td></tr>
<tr><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>When VUP expected being set to 5V
0 disabled, 1 enabled</td></tr>
<tr><td></td><td>X</td><td></td><td></td><td></td><td></td><td></td><td></td><td>When VUP expecting being set to 3.6v
0 disabled, 1 enabled</td></tr>
<tr><td></td><td></td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>RFU</td></tr>
</table>
NOTE: FW automatically knows whether 5V or 3,6V is expected on VUP depending on other configuration bytes.
Byte 10: RFU
</td><td></td><td></td><td></td></tr>
</table>

Table 113. Core configuration parameters*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>XTAL_
SETTINGS_CFG
RW in E²PROM
Persistent After FwDL</td><td>Parameter used to configure the XTAL oscillator
If the XTAL is selected in CLOCK_SEL_CFG
<table><tr><th>Byte</th><th>Description</th></tr><tr><td>0</td><td>XTAL kick time in µs</td></tr><tr><td>1</td><td>XTAL start time in µs (LSB)</td></tr><tr><td>2</td><td>XTAL start time in µs (MSB)</td></tr><tr><td>3</td><td>XTAL stop time in µs</td></tr></table></td><td>0xA0
0xA7</td><td>4</td><td>Byte 0:
0x14
Byte 1:
0x00
Byte 2:
0x00
Byte 3:
0x14</td></tr>
<tr><td>PLL_SETTINGS_
CFG
RW in E²PROM
Persistent After FwDL</td><td>Parameter used to configure the PLL lock
If the PLL is selected in CLOCK_SEL_CFG
<table><tr><th>Byte</th><th>Description</th></tr><tr><td>0</td><td>Lock time for PLL1 and PLL2</td></tr><tr><td>1</td><td>Lock time for PLL2 (PLL1 bypassed)</td></tr><tr><td>2</td><td>Lock loop iterations</td></tr><tr><td>3</td><td>Delay between disable and enable</td></tr></table></td><td>0xA0
0x16</td><td>4</td><td>Byte 0:
0xCD
Byte 1:
0x67
Byte 2:
0x22
Byte 3:
0xFF</td></tr>
<tr><td>MW_EEPROM_
AREA
RW in E²PROM
Persistent After FwDL</td><td>32-Byte EEPROM area dedicated to the MW to store/retrieve non-volatile data. The 32 Bytes have to be read (CORE_GET_CONFIG_CMD) or written (CORE_SET_CONFIG_CMD) in a row: it is not possible to access only a subset of these 32 Bytes.</td><td>0xA0
0x0F</td><td>32</td><td>N/A</td></tr>
<tr><td>DH_EEPROM_
AREA
RW in E²PROM
Persistent After FwDL</td><td>32-Byte EEPROM area dedicated to the DH to store/retrieve non-volatile data. The 32 Bytes have to be read (CORE_GET_CONFIG_CMD) or written (CORE_SET_CONFIG_CMD) in a row: it is not possible to access only a subset of these 32 Bytes.</td><td>0xA0
0x14</td><td>32</td><td>N/A</td></tr>
<tr><td>PLL_SETTING
Persistent After FwDL</td><td>PLL setting for external clock supplies, 19.2 MHz by default
See possible values to be used for external clocks below:
<table><tr><th>External clock frequency</th><th>PLL_SETTING value</th></tr><tr><td>13 MHz</td><td>0x02A2520802E10130</td></tr><tr><td>19.2 MHz</td><td>0x02E3518802E121B8</td></tr><tr><td>24 MHz</td><td>0x83A2C22802E11188</td></tr><tr><td>26 MHz</td><td>0x82A2520802E10130</td></tr><tr><td>32 MHz</td><td>0x82A351B802F0F188</td></tr><tr><td>38.4 MHz</td><td>0x82E3518802E12188</td></tr><tr><td>48 MHz</td><td>0x82D3513802E0E158</td></tr></table></td><td>0xA0
0x20</td><td>8</td><td>0x02E3518
802E121B8</td></tr>
</table>

Table 113. Core configuration parameters*...continued*

Name and Rights	Description	Ext. Tag	Len.	Default Value
DPLL_SETTING ***Persistent After FwDL***	DPLL setting for external clock supplies, 19.2 MHz by default See possible values to be used for external clocks below: External clock frequency / DPLL_SETTING value 13 MHz / 0x02A3424002E20188 19.2 MHz / 0x02E2018801A200F0 24 MHz / 0x02D3413802E20188 26 MHz / 0x01A3412002E20188 32 MHz / 0x82A301B002E20188 38.4 MHz / 0x82E2018801A200F0 48 MHz / 0x82D3413802E20188	0xA0 0x26	8	0x02E2018 801A200F0

13.2 [PN7160-NCI] extension: RF Discovery configuration

13.2.1 Poll Mode

Several configuration parameters are required for the Poll Mode in RF discovery:

Table 114. Poll Mode configuration

Name and Rights	Description	Ext. Tag	Len.	Default Value
TAG_ DETECTOR_CFG ***RW in E²PROM***	Tag detector enabling/disabling as follows: Bit Mask (b7 b6 b5 b4 b3 b2 b1 b0) / Description X / Activation of the Trace mode X (b3) / Fake LPCD Wake up X (b0) / Enable Tag Detector (Detection based on the AGC) 0 0 0 0 0 0 / RFU So the valid values for this parameter are: 0x00 - tag detector disabled 0x01 - tag detector enabled (using AGC) 0x09 - tag detector enabled with fake detection reported (CORE_GENERIC_ERROR_NTF with reason "0xA3" will be sent when LPCD is triggered without any tag found, see Section 7.4.9.3) 0x81 - tag detector enabled with trace mode (see Section 12.4.3)	0xA0 0x40	1	0x00
TAG_ DETECTOR_ THRESHOLD_ CFG ***RW in E²PROM***	Sets the detection level (threshold between last Reference AGC value and actual measured AGC value)	0xA0 0x41	1	0x04
TAG_ DETECTOR_ PERIOD_CFG ***RW in E²PROM***	Time in steps of 8us to wait before sampling the AGC value.	0xA0 0x42	1	0x0F

Table 114. Poll Mode configuration*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>TAG_ DETECTOR_ FALLBACK_CNT_ CFG
RW in E²PROM</td><td>Parameter used to configure the "Hybrid" mode to insert a regular Polling cycle every N pulses generated by the LPCD
<table><tr><td>0x00</td><td>Hybrid mode disabled: LPCD only, no regular Polling cycle unless an "object" is detected by the LPCD.</td></tr><tr><td>0x02- 0xFF</td><td>Hybrid mode enabled, inserting a regular Polling cycle every 'N' pulses of LPDC. 'N' is coded by the value assigned to TAG_DETECTOR_FALLBACK_CNT _CFG in decimal. Default N = 0x50 = 80.</td></tr></table></td><td>0xA0 0x43</td><td>1</td><td>0x50</td></tr>
<tr><td>TechDet_AFTER_ LPCD_CFG
RW in E²PROM</td><td>Parameter used to configure the RF Discovery taking place right after the Low-Power Card Detector has triggered a detection:
<table><tr><th colspan="8">Bit Mask</th><th>Description</th></tr><tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr><tr><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td>TechDet_PERIOD
In steps of 10ms</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>TechDet_NBR_RETRIES</td></tr></table>See →Section 12.4.2 for more details on the use of this parameter.</td><td>0xA0 0x61</td><td>1</td><td>0x00</td></tr>
<tr><td>POLL_PROFILE_ SEL_ CFG
RW in E²PROM</td><td>Discovery profile selection in Poll Mode as follows:
<table><tr><td>0x00</td><td>NFC FORUM profile according [ACTIVITY] All static configurations (Bail-out) will be set to the [NCI] default value (disabled).</td></tr><tr><td>0x01</td><td>EMVCo PCD profile</td></tr><tr><td>0x02- 0xFF</td><td>RFU</td></tr></table></td><td>0xA0 0x44</td><td>1</td><td>0x00</td></tr>
<tr><td>STANDARD_ SEL_CFG
RW in E²PROM</td><td>Default standard supported by NFCC
<table><tr><th colspan="8">Bit Mask</th><th>Description</th></tr><tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td>x</td><td></td><td>If set to 1, NFC Forum Activity 1.1 is selected by default
If set to 0, NFC Forum Activity 1.0 is selected by default</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td>1</td><td>RFU</td></tr></table></td><td>0xA0 0x3F</td><td>1</td><td>0x03</td></tr>
<tr><td>GT_NFC-AA_CFG
RW in E²PROM</td><td>Guard time (in steps of 0.59 µs) used between the start of unmodulated RF field and 1st command for Poll NFC-A Active (min='0001', max='FFFF')</td><td>0xA0 0x46</td><td>2</td><td>0x21C4 (5.1ms)</td></tr>
<tr><td>GT_NFC-AP_CFG
RW in E²PROM</td><td>Guard time (in ms) used between the start of unmodulated RF field and 1st command for Poll NFC-A Passive (min='0001', max='FFFF')</td><td>0xA0 0x47</td><td>2</td><td>0x21C4 (5.1ms)</td></tr>
<tr><td>GT_NFC-B_CFG
RW in E²PROM</td><td>Guard time (in ms) used between the start of unmodulated RF field and 1st command for Poll NFC-B Passive (min='0001', max='FFFF')</td><td>0xA0 0x48</td><td>2</td><td>0x21C4 (5.1ms)</td></tr>
<tr><td>GT_NFC-F_CFG
RW in E²PROM</td><td>Guard time (in ms) used between the start of unmodulated RF field and 1st command for Poll NFC-F Passive (min='0001', max='FFFF')
Note: If previous phase on polling loop is a FeliCa Poll that fail on Timeout, you will see an additional 5 ms delay due to the FeliCa timeout itself</td><td>0xA0 0x49</td><td>2</td><td>0x84E2 (20.07ms)</td></tr>
</table>

Table 114. Poll Mode configuration*...continued*

Name and Rights	Description	Ext. Tag	Len.	Default Value
GT_15693_CFG ***RW in E²PROM***	Guard time (in ms) used between the start of unmodulated RF field and 1st command for Poll 15693 Passive (min='0001', max='FFFF')	0xA0 0x4A	2	0x211B (5ms)
PF_SYS_CODE_ CFG ***RW in E²PROM***	Discovery configuration parameters for Poll F: system code	0xA0 0x4C	2	0xFFFF
MFC_KEY-0_CFG ***WO[1] in E²PROM***	Key 0, used in MIFARE Classic Authentication command.	0xA0 0x4D	6	0xA0A1A2A3 A4A5
MFC_KEY-1_CFG ***WO[1] in E²PROM***	Key 1, used in MIFARE Classic Authentication command.	0xA0 0x4E	6	0xD3F7D 3F7D3F7
MFC_KEY-2_CFG ***WO[1] in E²PROM***	Key 2, used in MIFARE Classic Authentication command.	0xA0 0x4F	6	0xFFFFF FFFFFFF
MFC_KEY-3_CFG ***WO[1] in E²PROM***	Key 3, used in MIFARE Classic Authentication command.	0xA0 0x50	6	0xFFFFF FFFFFFF
MFC_KEY-4_CFG ***WO[1] in E²PROM***	Key 4, used in MIFARE Classic Authentication command.	0xA0 0x51	6	0xFFFFF FFFFFFF
MFC_KEY-5_CFG ***WO[1] in E²PROM***	Key 5, used in MIFARE Classic Authentication command.	0xA0 0x52	6	0xFFFFF FFFFFFF
MFC_KEY-6_CFG ***WO[1] in E²PROM***	Key 6, used in MIFARE Classic Authentication command.	0xA0 0x53	6	0xFFFFF FFFFFFF
MFC_KEY-7_CFG ***WO[1] in E²PROM***	Key 7, used in MIFARE Classic Authentication command.	0xA0 0x54	6	0xFFFFF FFFFFFF
MFC_KEY-8_CFG ***WO[1] in E²PROM***	Key 8, used in MIFARE Classic Authentication command.	0xA0 0x55	6	0xFFFFF FFFFFFF
MFC_KEY-9_CFG ***WO[1] in E²PROM***	Key 9, used in MIFARE Classic Authentication command.	0xA0 0x56	6	0xFFFFF FFFFFFF
MFC_KEY-10_ CFG ***WO[1] in E²PROM***	Key 10, used in MIFARE Classic Authentication command.	0xA0 0x57	6	0xFFFFF FFFFFFF
MFC_KEY-11_ CFG ***WO[1] in E²PROM***	Key 11, used in MIFARE Classic Authentication command.	0xA0 0x58	6	0xFFFFF FFFFFFF
MFC_KEY-12_ CFG ***WO[1] in E²PROM***	Key 12, used in MIFARE Classic Authentication command.	0xA0 0x59	6	0xFFFFF FFFFFFF
MFC_KEY-13_ CFG ***WO[1] in E²PROM***	Key 13, used in MIFARE Classic Authentication command.	0xA0 0x5A	6	0xFFFFF FFFFFFF
MFC_KEY-14_ CFG ***WO[1] in E²PROM***	Key 14, used in MIFARE Classic Authentication command.	0xA0 0x5B	6	0xFFFFF FFFFFFF
MFC_KEY-15_ CFG ***WO[1] in E²PROM***	Key 15, used in MIFARE Classic Authentication command.	0xA0 0x5C	6	0xFFFFF FFFFFFF

Table 114. Poll Mode configuration*...continued*

<table>
<tr><th>Name and Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>FSDI_CFG
RW in E²PROM</td><td>Frame Size value for the NFCC in reader mode presented in RATS or ATTRIB</td><td>0xA0 0x5D</td><td>1</td><td>0x08</td></tr>
<tr><td>JEWEL_RID_CFG
RW in E²PROM</td><td>Parameter used to configure if the RID is sent on RF to the T1T by PN7160 during the RF activation or not (for NCI 1.0 behavior only):
0x01 => The RID is sent on RF to the T1T 0x00 => The RID is NOT sent on RF to the T1T.
In both cases, the RF_INTF_ACTIVATED_NTF will NOT embed the RID response from the T1T, as defined in [NCI 1.0] This is useless in NCI 2.0 mode since the RID is sent on RF anyway and the response is transported inside the RF_INTF_ACTIVATED_NTF.</td><td>0xA0 0x5E</td><td>1</td><td>0x00</td></tr>
<tr><td>FELICA_TSN_CFG
RW in E²PROM</td><td>TSN value transported by the NFCC in the SENSF_REQ command: the DH defines the number of time slots for collision resolution
<table>
<tr><th colspan="8">Bit Mask</th><th>Description</th></tr>
<tr><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td><td></td></tr>
<tr><td>X</td><td>X</td><td>X</td><td>X</td><td></td><td></td><td></td><td></td><td>TSN for collision resolution activity</td></tr>
<tr><td></td><td></td><td></td><td></td><td>X</td><td>X</td><td>X</td><td>X</td><td>TSN for detection activity</td></tr>
</table></td><td>0xA0 0x5F</td><td>1</td><td>0x30</td></tr>
<tr><td>READER_FWITOX_NTF_CFG
RW in E²PROM</td><td>Delay before sending TOX notification to Host. Unit is 4096/fc (~302us)</td><td>0xA0 0x65</td><td>2</td><td>0x1000</td></tr>
</table>

[1] WO (Write Only) parameters can only be written, using CORE_SET_CONFIG_CMD. PN7160 will always return CORE_GET_CONFIG_RSP(STATUS_INVALID_PARAM) to any attempt to read the value of the WO parameter.

13.2.2 Listen Mode

Table 115. Listen Mode Configuration

<table>
<tr><th>Name & Rights</th><th>Description</th><th>Ext. Tag</th><th>Len.</th><th>Default Value</th></tr>
<tr><td>TO_RF_OFF_CFG
RW in E²PROM</td><td>Specifies the time-out (in ms) applied by PN7160 before it restarts a Polling sequence, after it has detected a Field-OFF in Listen Mode</td><td>0xA0 0x80</td><td>2</td><td>0x012C (300ms)</td></tr>
<tr><td>LISTEN_ISODEP_FSCI_ CFG
RW in E²PROM</td><td>Parameter to define the FSC parameter (RF Frame Size for the PICC), as defined in [DIGITAL]:
<table>
<tr><td>0x00</td><td>FSC = 16</td></tr>
<tr><td>0x01</td><td>FSC = 24</td></tr>
<tr><td>0x02</td><td>FSC = 32</td></tr>
<tr><td>0x03</td><td>FSC = 40</td></tr>
<tr><td>0x04</td><td>FSC = 48</td></tr>
<tr><td>0x05</td><td>FSC = 64</td></tr>
<tr><td>0x06</td><td>FSC = 96</td></tr>
<tr><td>0x07</td><td>FSC = 128</td></tr>
<tr><td>0x08</td><td>FSC = 256</td></tr>
<tr><td>0x09- 0xFF</td><td>RFU</td></tr>
</table></td><td>0xA0 0x83</td><td>1</td><td>0x08</td></tr>
</table>

Table 115. Listen Mode Configuration*...continued*

Name & Rights	Description	Ext. Tag	Len.	Default Value
NDEF_INTF_CFG	Enable or disable NDEF emulation. See →Section 7.6 for more details on this parameter.	0xA0 0x95	1	0x00

13.3 [PN7160-NCI] extension: Contactless Interface configurations

PN7160 offers multiple configuration options for the Contactless Interface, to allow an optimum match between the antenna characteristics and the transmitter and receiver in PN7160.

A generic TLV mechanism has been defined to write the Contactless Interface settings. It relies on the [NCI] CORE_SET_CONFIG_CMD and is described hereafter:

Table 116. Mechanism to configure the RF settings or RF transitions

Name & Rights	Description	Ext. Tag	Len.	Default Value
DLMA_CTRL ***RW in E²PROM***	Parameter enabing/disabling DLMA feature and defining options, see chapter →Section 8.7.	0xA0 0xAF	12	N/A
DLMA_RSSI ***RW in E²PROM***	Parameter defining RSSI threshold values, see chapter →Section 8.7.	0xA0 0x34	148	N/A
DLMA_TX ***RW in E²PROM***	Parameter defining TX settings lookup table, see chapter →Section 8.7.	0xA0 0xA9	160	N/A
REXT_RSSI_CFG ***RW in E²PROM***	Parameter defining Rext and associated correction gain, see [AN13223] referenced in →Section 3 for setting guidance. Bits[32-16]: wRextAGCCor (in Omhs) Bits[15-0]: wRextGainCor	0xA0 0xAA	4	0x02300400
RFLD_CFG ***RW in E²PROM***	bits[6:0] = Enhanced RFLD sleep time bits[11:7] = Enhanced RFLD wait time bits[15:12] = Enhanced RFLD clock recovery time bits[17:16] = Enhanced RFLD pre-amplifier gain (00b → 9db, 01b → 10db, 10b → 11db, 11b → 12db) bit[18] = Enhanced RFLD disable (0 → enabled, 1 → disabled) bit[19] = Bypass digital frequency check bits[24:20] = IBias wait time bits[31:25] = RFU Bits[47:32]: 16 bits to start the RFLD	0xA0 0x1F	6	0x006313292FAB
NFCLD ***RW in E²PROM***	Byte 0: CLIF_ANA_NFCLD_REG value for external RF ON detection Byte 1: value for external RF OFF detection (step is 2.5 mV) Byte2: CLIF_ANA_NFCLD_REG for Target P+A Byte3: Bit[0] = 0: Apply RSSI Interpolation algo, 1: RSSI is either raw ADC or AGC (depends on bit 12 of APC entry)	0xA0 0x38	4	Byte 0: 0x14 Byte 1: 0x0B Byte 2: 0x0B Byte 3: 0x00
DPC_CONFIG ***RW in E²PROM***	Settings to enable or disable the DPC (please refer to [AN13224], as referenced in →Section 3 for more information about this register)	0xA0 0x0B	87	N/A

Table 116. Mechanism to configure the RF settings or RF transitions*...continued*

Name & Rights	Description	Ext. Tag	Len.	Default Value
RF_ TRANSITION_ CFG ***RW in E²PROM***	Parameter to configure one RF transition The list of transition IDs and the appropriate values for the register offset as well as its value is available in the [AN13218], as referenced in →Section 3	0xA0 0x0D	3, 4 or 6	N/A

/!\	PN7160 only supports RF_TRANSITION_CFG with the CORE_SET_CONFIG_ CMD; CORE_GET_CONFIG_CMD is not supported.

[PN7160] comes with the possibility to read out the values of the RF transitions. This mechanism to read out the values is not based on the [NCI] CORE_GET_CONFIG_CMD, it rather uses a specific command: RF_GET_TRANSITION_CMD:

Table 117. RF_GET_TRANSITION_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x14	2	The DH asks to read out the value of an RF Transition

Table 118. RF_GET_TRANSITION_CMD parameters

Parameter		Length	Value/Description
1	RF Transition ID	1 Byte	RF Transition Identifier
2	CLIF Register Offset	1 Byte	Offset of the register to read out from the CLIF

Table 119. RF_GET_TRANSITION_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x14	2	The NFCC acknowledges the command received from the DH and sends the RF Transition value to the DH

Table 120. RF_GET_TRANSITION_RSP parameters

Parameter		Length	Value/Description
1	Status	1 Byte	One of the following Status codes, as defined in [NCI_ Table1]: 0x00: STATUS_OK 0x01: STATUS_REJECTED 0x06: STATUS_SEMANTIC_ERROR Others RFU

Table 120. RF_GET_TRANSITION_RSP parameters*...continued*

Parameter		Length	Value/Description
2	RF Transition Length	1 Byte	Length of the following parameter (RF Transition Value): 0x01: 1 byte to follow 0x02: 2 bytes to follow 0x04: 4 bytes to follow Others RFU
3	RF Transition Value	1, 2 or 4 bytes	RF Transition Value **!!! Value coded in little endian !!!**

14 Test Modes

/!\	The PN7160 had previously a specific test command to switch the RF Field On or Off (TEST_RF_FIELD_CMD). This specific command does not exist anymore and has been replaced by an extension of the TEST_ANTENNA_CMD.

14.1 Test Session

Whatever the test command used by the DH, it is necessary to implement a "test session", which isolates the test mode from a regular "NCI session" of PN7160. This test session is defined thanks to the following sequence:

- Reset/Init the PN7160 using CORE_RESET_CMD/CORE_INIT_CMD
- Launch the selected test function.
- Get the response (optionally the notifications) transporting the status of the Test executed
- Reset/Init the PN7160 using CORE_RESET_CMD/CORE_INIT_CMD (except for TEST_PRBS_CMD/RSP, which requires a HW reset first to stop the pattern generation on RF).

14.2 TEST_ANTENNA_CMD/RSP

This command is used to execute the antenna self-test measurements, which allow to check that all the discrete components connected between PN7160 and the contactless antenna are properly soldered on the PCB.

Four different measurements are necessary to check the correct connection of all the discrete components, therefore a complete Antenna Self-Test requires to execute the TEST_ANTENNA_CMD 4 consecutive times, with a different set of parameters for each execution.

Table 121. TEST_ANTENNA_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x3D	2-4	Command to execute antenna self-test measurements.

Table 122. TEST_ANTENNA_CMD parameters

Parameter		Length	Value/Description
1	Test ID	1 Byte	0x01: TxLDO current measurement 0x02: AGC value reading 0x04: AGC value reading with fixed NFCLD level 0x20: Switch RF Field On/Off Others: RFU
2	Parameters of individual test measurement	1-3 bytes	For individual test parameters, refer to →Table 123

Table 123. Parameters to include in TEST_ANTENNA_CMD depending on the measurement to perform

Test ID	Measurement Description	Param. number	Parameter name	Length	Description	Typ. value
0x01	TxLDO current measurement	1	Wait_Time	1 byte	Time to wait (in µs) before capturing the TX-LDO current	0x80
0x02	AGC value reading	1	Wait_Time	1 byte	Time to wait (in µs) before capturing the AGC value	0xC8
		2	CLIF AGC input [7:0]	1 byte	Value to write in CLIF AGC input register, bits [7:0]	0x60
		3	CLIF AGC input [9:8]	1 byte	The 2 LSbits of parameter 3 are mapped on bits [9:8] of CLIF AGC input register. The 6 MSbits of parameter 3 have to be set to '0'.	0x03
0x04	AGC value reading with fixed NFCLD level	1	Wait_Time	1 byte	Time to wait (in µs) before capturing the AGC value	0x20
		2	CLIF ANA NFCLD value [3:0]	1 byte	The LSbits of parameter 2 are mapped on bits [5:0] of CLIF ANA NFCLD input register.	0x3F
		3	TxLDO control voltage	1 byte	parameter 3 is mapped to PMU TxLDO cntrl register and corresponds to TxLdo voltage of 2.7 V	0x08
0x20[1]	Switch RF Field On/Off	1	RF Field Generation	1 byte	'1' => RF Field is generated '0' => RF Field is not generated	

/!\ [1] Option 0x20 (Switch RF Field On/Off) absolutely requires to first disable the Standby mode, thanks to the CORE_SET_POWER_MODE_CMD (see →Section 12.6.1).

Table 124. TEST_ANTENNA_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x3D	5	PN7160 returns individual measurement status code and the result of the measurement.

Table 125. TEST_ANTENNA_RSP parameters

Parameter		Length	Value/Description
1	Status	1 Byte	0x00: STATUS_OK 0x01: Test execution rejected (PN7160 in wrong state) 0x04: STATUS_TEST_EXEC_FAILED 0x09: STATUS_INVALID_PARAM Others: RFU

Table 125. TEST_ANTENNA_RSP parameters*...continued*

Parameter		Length	Value/Description
2	Result_Parameter_1	1 Byte	Value depending on the measurement performed: see →Table 126
3	Result_Parameter_2	1 Byte	
4	Result_Parameter_3	1 Byte	
5	Result_Parameter_4	1 Byte	

Table 126. Parameters provided in TEST_ANTENNA_RSP as a result of the measurement performed

Test ID	Measurement Description	Param. number	Parameter name	Length	Description
0x01	TxLDO current measurement	1	TxLDO output value	1 byte	Raw value (RawVal) of TxLDO measurement (0x00-0x7F) in mA (real absolute value to be calculated in addition to the offset below) Raw value is coded on 7 bits and can take values from 0 to 127.
		2	Measured range	1 byte	Selection of offset for detection range: 0x00: Absolute value = RawVal + 20 [mA] 0x01: Absolute value = RawVal + 60 [mA] 0x02: Absolute value = RawVal + 120 [mA] 0x03: Absolute value = RawVal + 160 [mA]
		3	RFU	1 byte	
		4	RFU	1 byte	
0x02	AGC value reading	1	AGC Value LSB	1 byte	Measured AGC Value (LSB)
		2	AGC Value MSB	1 byte	Measured AGC Value (MSB)
		3	RFU	1 byte	
		4	RFU	1 byte	
0x04	AGC value reading with fixed NFCLD level	1	AGC Value LSB	1 byte	Measured AGC Value that triggers NFCLD (LSB)
		2	AGC Value MSB	1 byte	Measured AGC Value that triggers NFCLD (MSB)
		3	RFU	1 byte	
		4	RFU	1 byte	
0x20	Switch RF Field On/Off	1	RFU	1 byte	
		2	RFU	1 byte	
		3	RFU	1 byte	
		4	RFU	1 byte	

/!\	RFU Bytes in TEST_ANTENNA_RSP can have any value from 0x00 to 0xFF.

14.3 TEST_PRBS_CMD/RSP

This command is used to start PRBS infinite stream generation. Both PRBS generation done by the FW or by a specific HW block can be used, depending on what fits the needs the best. A 511-bit pseudo-random test sequence (PRBS9) or a 32 767-bit pseudo-random test sequence (PRBS15) can be selected.

Table 127. TEST_PRBS_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x30	6	Command to start PRBS generation

Table 128. TEST_PRBS_CMD parameters

Parameter		Length	Value/Description
1	PRBS Mode	1 Byte	0x00: Firmware generated PRBS 0x01: Hardware generated PRBS Others: RFU
2	PRBS type	1 Byte	0x00: PRBS9 when Hardware generated PRBS selected 0x01: PRBS15 when Hardware generated PRBS selected Others: Ignored when Firmware generated PRBS is selected
3	Technology to stream	1 Byte	0x00: Type A 0x01: Type B 0x02: Type F Others: RFU
4	Bitrate	1 Byte	0x00: 106 kbit/s (Type A,B) 0x01: 212 kbit/s (Type A,B and F) 0x02: 424 kbit/s (Type A,B and F) 0x03: 848 kbit/s (Type A,B) Others: RFU
5	PRBS series length	2 bytes	A value between 0x0001 – 0x01FF when Firmware generated PRBS is selected

Table 129. TEST_PRBS_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x30	1	PN7160 reports if the TEST_PRBS_CMD is successful.

In case the NCI package is malformed (wrong length for example) STATUS_SYNTAX_ERROR is returned. If parameters are out of bounds (RFU values as stated above), STATUS_INVALID_PARAM will be sent. In case the PRBS tests are started in any state other than RFST_IDLE, STATUS_SEMANTIC_ERROR will be returned.

Table 130. TEST_PRBS_RSP parameter

Parameter	Length	Value/Description
Status	1 byte	0x00: STATUS_OK 0x05: STATUS_SYNTAX_ERROR 0x06: STATUS_SEMANTIC_ERROR 0x09: STATUS_INVALID_PARAM Others: RFU

/!\	The only way to stop the on-going PRBS pattern generation is to apply a HW reset (through the VEN pin).

14.4 TEST_GET_REGISTER_CMD/RSP

This command is used to retrieve the current Value of the AGC_VALUE_REGISTER.

Table 131. TEST_GET_REGISTER_CMD

GID	OID	Numbers of parameter(s)	Description
1111b	0x32	1	0x01: Command to retrieve the Value of the AGC_VALUE_REGISTER Others: RFU

Table 132. TEST_GET_REGISTER_RSP

GID	OID	Numbers of parameter(s)	Description
1111b	0x32	1	8 bytes corresponding to: Status (1 byte) Current Configuration, Raw RSSI or not (1 byte) Trim source (1 byte) Current Offset of APC table (1 byte) Raw RSSI (2 bytes) Interpolated RSSI (2 bytes)

15 PN7160 download mode

15.1 Introduction

The main part of the PN7160 firmware is permanently stored in an embedded ROM and a small part in an embedded FLASH. This last part of firmware can be updated using the secured firmware update mechanism, most commonly called “sFWu”.

User data is stored in EEPROM and is protected by anti-tearing mechanisms that ensure the integrity and availability of the data. Confidentiality is also ensured for critical data in a dedicated protected area that cannot be updated.

In order to provide NXP’s customers with features that are compliant with the last standards (for instance NFC Forum specifications), both the code in FLASH and the user data in EEPROM can be updated. NXP is in charge of delivering new firmware updates, together with new User data.

The aim of this section is to detail the PN7160 firmware update concepts.

15.2 DH Interface

As already mentioned in →Section 6.1, the PN7160 has two main modes of operation to communicate with the DH:

1. NCI-based communications
2. HDLL-Based communications, only used when the PN7160 is triggered to enter the download mode

Most of the information given in chapter →Section 6 still applies here; the only difference is that the frames exchanged do not carry NCI packets anymore, they carry HDLL packets.

15.2.1 Accessing download mode

In order to force the PN7160 enter the download mode, the DH has to follow the sequence described below:

1. Force DWL_REQ pin to logical ‘1’
2. Generate a reset, either:
   - A hardware reset by pulsing VEN pin low for at least 10 µs
   - A software reset by sending the NCI CORE_RESET_CMD command

PN7160 is now ready to receive commands in download mode, using the HDLL framing.

To get out of the download mode and come back to the normal mode (NCI framing), the DH has to follow the sequence described below:

1. Force DWL_REQ pin to logical ‘0’
2. Generate a reset, either
   - A hardware reset by pulsing VEN pin low for at least 10 µs
   - A software reset by sending the DL_RESET command

PN7160 is now ready to receive NCI commands, assuming that the previous sFWu sequence was successful.

15.2.2 Description of HDLL

HDLL is the link layer developed by NXP to ensure a reliable upload mode.

An HDLL message is made of a 2-Bytes header, followed by a frame, comprising the Opcode and the Payload of the command. Each message ends with a 16-bits CRC, as described on the picture below:

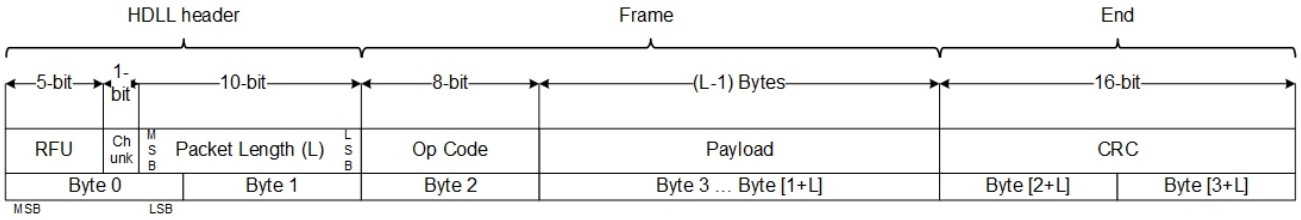

Figure 61. HDLL frame

The HDLL header contains:

- A chunk bit, which indicates if this is the only or last chunk of a message (chunk = 0) or if at least one other chunk will follow (chunk = 1)
- the length of the Payload coded on 10 bits. So the HDLL Frame Payload can go up to 1023 Bytes.

The byte order has been defined as Big Endian, meaning MSByte first.

The CRC16 is compliant to X.25 (CRC-CCITT, ISO/IEC13239) standard with polynome x^16 + x^12 + x^5 +1 and preload value 0xFFFF.

It is calculated over the whole HDLL frame, i.e Header + Frame.

Sample C-Code implementation:

```
static uint16_t phHal_Host_CalcCrc16(uint8_t* p, uint32_t dwLength)
{
   uint32_t i;
   uint16_t crc_new;
   uint16_t crc = 0xffffU;
   for (i = 0; i < dwLength; i++)
   {
      crc_new = (uint8_t)(crc >> 8) | (crc << 8);
      crc_new ^= p[i];
      crc_new ^= (uint8_t)(crc_new & 0xff) >> 4;
      crc_new ^= crc_new << 12;
      crc_new ^= (crc_new & 0xff) << 5;
      crc = crc_new;
   }
   return crc;
}
```

15.2.3 Transport mapping over I²C

15.2.3.1 Write Sequence from the DH

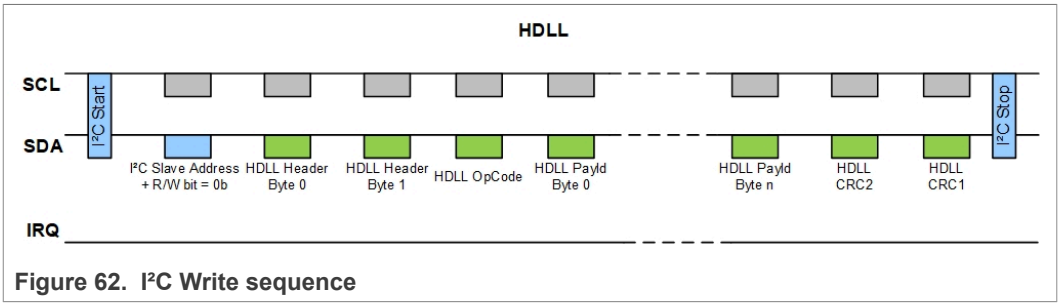

Figure 62. I²C Write sequence

PN7160 may send an I²C NACK (negative acknowledge) when none of the 3 buffers used in the PN7160 is free, which may happen in case the DH does not wait for a response before issuing a new Write command. If one single byte of a complete NCI frame is NACKed by the PN7160, the DH has to re-send the complete HDLL frame and not only this single byte.

15.2.3.2 Read Sequence from the DH

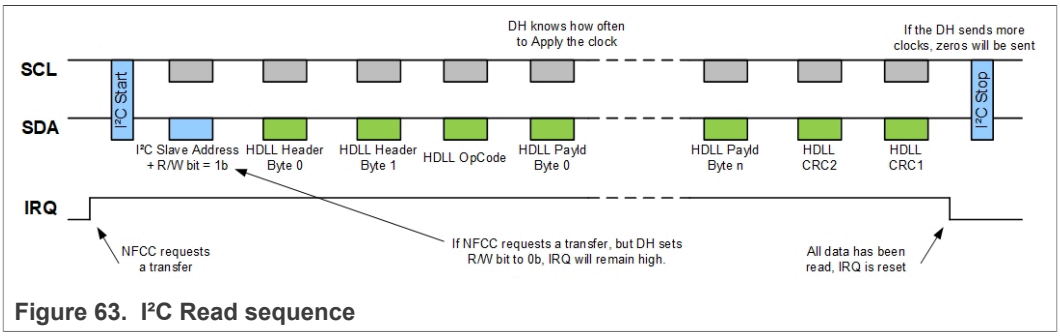

Figure 63. I²C Read sequence

15.2.3.3 Split mode

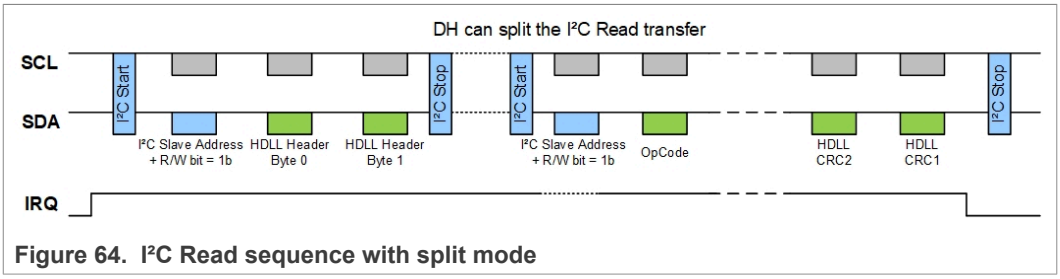

Figure 64. I²C Read sequence with split mode

15.2.4 Transport mapping over SPI

15.2.4.1 Write Sequence from the DH

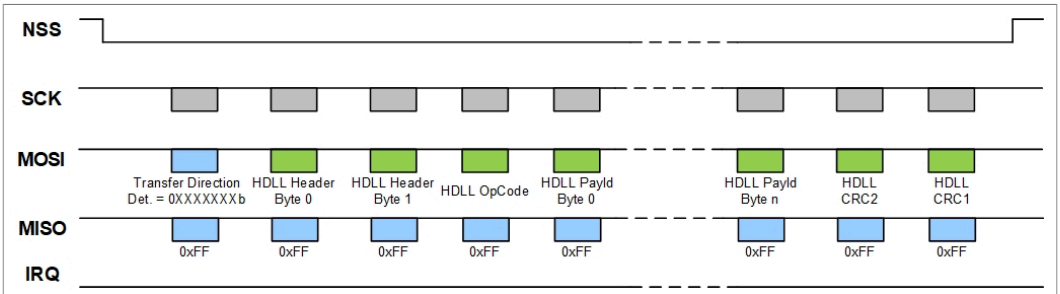

Figure 65. Example of a SPI Write access

15.2.4.2 Read Sequence from the DH

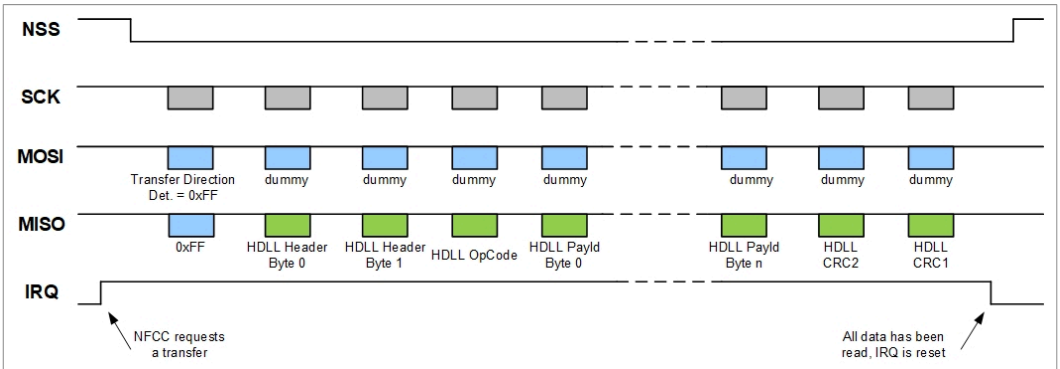

Figure 66. Example of a SPI Read access

15.2.4.3 Split mode

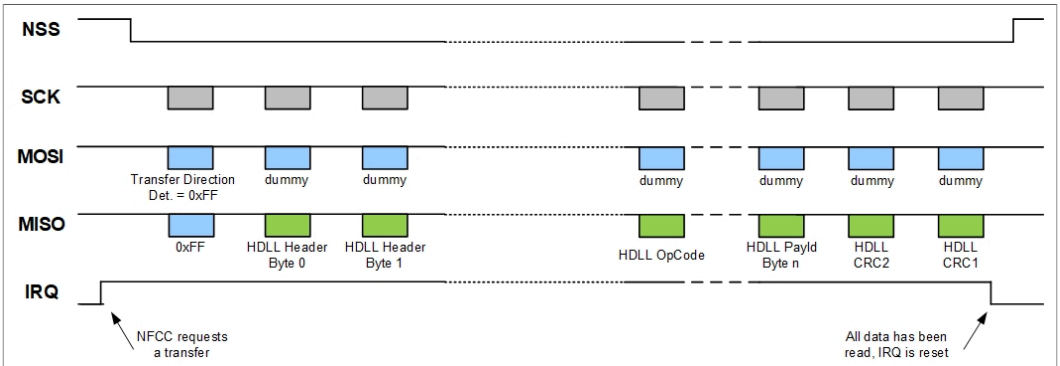

Figure 67. SPI Read sequence with split mode

15.2.4.4 Invalid Sequence from the DH

Any SPI data transfer starting by a Transfer Direction Detector Byte different from either 0XXXXXXXb or 11111111b is discarded by PN7160, as this is an invalid frame.

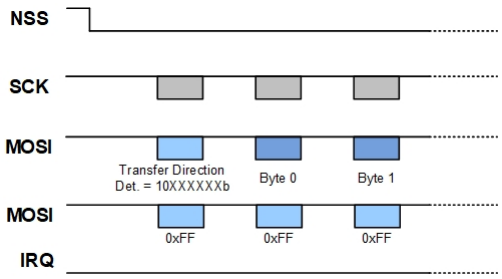

Figure 68. Example of an invalid SPI

15.3 Download mode command set

Commands and responses have the following frame format:

```
DH <--> PN7160 : [{HEAD1 HEAD2} {CMD/STAT} {DATA[0..N-1]} {CRC1 CRC2}]
                  |             |           |              |
                  |             |           |              +-> CRC16
                  |             |           +-> payload
                  |             +-> command opCode or status code
                  +-> chunk flag & length of the payload
```

Example with the *Reset* command:

```
DH ---> PN7160 : [0x00 0x04 0xF0 0x00 0x00 0x00 0x18 0x5B]
```

Table 133. Command set

OpCode	Command alias	Comment
F0h	DL_RESET	Perform software reset
F1h	DL_GETVERSION	Retrieve FW version currently loaded
C0h	DL_SECWRITE	Perform secure write operation

The table below lists the possible response to the download commands.

Table 134. Status codes

Status	Status alias	Comment
00h	DL_OK	command passed
01h	DL_INVALID_ADDR	address not allowed
02h	DL_GENERIC_ERROR	No Version Access
0Bh	DL_UNKNOW_CMD	unknown command
0Ch	DL_ABORTED_CMD	chunk seq. is too big
0Dh	DL_PLL_ERROR	flash not activated
1Eh	DL_ADDR_RANGE_OFL_ERROR	address out of range
1Fh	DL_BUFFER_OFL_ERROR	the buffer is too small
20h	DL_MEM_BSY	no key access
21h	DL_SIGNATURE_ERROR	signature mismatch
24h	DL_FIRMWARE_VERSION_ERROR	already up to date
28h	DL_PROTOCOL_ERROR	protocol error
2Ah	DL_SFWU_DEGRADED	degraded state reached

Table 134. Status codes...*continued*

Status	Status alias	Comment
2Dh	PH_STATUS_DL_FIRST_CHUNK	first chunk received
2Eh	PH_STATUS_DL_NEXT_CHUNK	subsequent chunk received
C5h	PH_STATUS_INTERNAL_ERROR	length mismatch

15.3.1 Reset command

This command resets the PN7160 when is download mode.

Table 135. Reset command

OpCode	Length	Parameter
F0h	0	-

This command trigger software reset action when it is correct and provide a response only in case of error.

```
DH ---> PN7160 : [0x00 0x04 0xF0 0x00 0x00 0x00 0x18 0x5B]
DH <--- PN7160 : [0x00 0x04 STAT 0x00 0x00 0x00 CRC1 CRC2]
```

`STAT` is the return status in case of error.

15.3.2 Get version command

This command allows retrieving current PN7160 HW and FW version.

Table 136. Get version command

OpCode	Length	Parameter
F1h	0	-

```
DH ---> PN7160 : [0x00 0x04 0xF1 0x00 0x00 0x00 0x6E 0xEF]
DH <--- PN7160 : [0x00 0x0A STAT HW_V ROMV RFU1 RFU2 RFU3 RFU4 RFU5 MAJV MINV CRC1 CRC2]
```

`STAT` is the return status.

`HW_V` is the HW version.

`ROMV` is the ROM code version.

`MAJV` is the major FW version.

`MINV` is the minor FW version.

The PN7160 FW version is usually defined as `ROMV.MAJV.MINV`, for instance 12.50.05.

15.3.3 Secure write command

This command is used in the sFWu procedure to write Flash and EEPROM internal memory.

Table 137. Secure write command

OpCode	Length	Parameter
C0h	Parameter length	See Figure 69

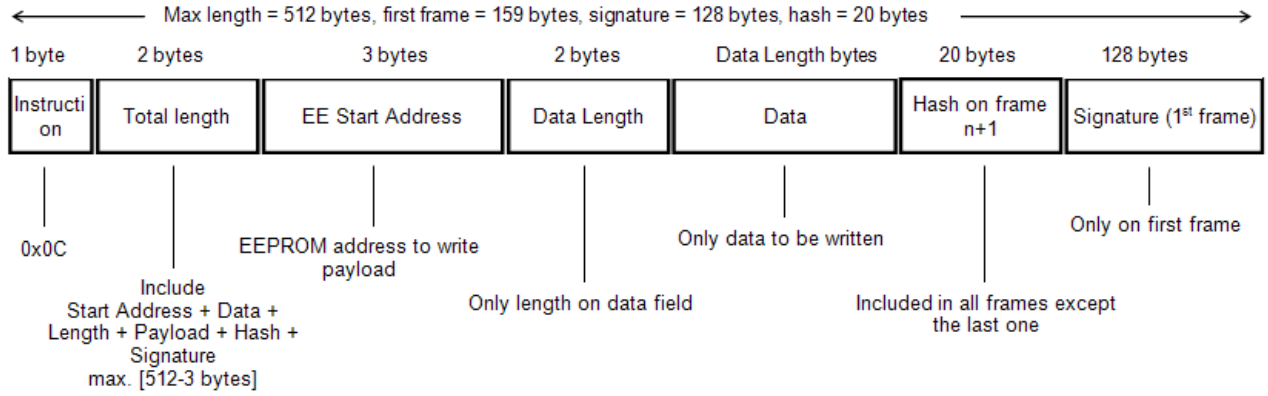

Figure 69. Secure write frame description

3 Types of frames will be sent during a sFWu sequence:

- First frame: first command of the firmware download process must contain the firmware version number, a hash value, and a signature (firmware version's upper byte is the major version and lower byte is the minor version). They are not written directly as for other frames but handled internally.

```
DH ---> PN7160 : [0x00 0xE4 0xC0 0x00 {VERS} {HASH} {SIGN} CRC1 CRC2]
DH <--- PN7160 : [0x00 0x04 STAT 0x00 0x00 0x00 CRC1 CRC2]
```

  VERS is the 2 bytes version of the firmware to be uploaded (major and minor FW version).
  HASH is the 32 bytes digest of the 2nd frame.
  SIGN is the 192 bytes digital signature of the first frame.
  STAT is the return status.
- Middle frames: commands following the first frame command type shall contain firmware data and a hash value.

```
DH ---> PN7160 : [{LEN} 0xC0 {ADDR} {SIZE} {DATA} {HASH} CRC1 CRC2]
DH <--- PN7160 : [0x00 0x04 STAT 0x00 0x00 0x00 CRC1 CRC2]
```

  LEN is the 2 bytes length of the play load.
  ADDR is the 3 bytes address of the data to write to flash or EEPROM memory.
  DATA is the data to write.
  HASH is the 32 bytes digest of the next frame.
  STAT is the return status.
- Last frame: last command of a firmware download process shall only contain firmware data. Additionally to data written in EEPROM some extra processing is handled internally.

```
DH ---> PN7160 : [{LEN} 0xC0 {ADDR} {SIZE} {DATA} CRC1 CRC2]
DH <--- PN7160 : [0x00 0x04 STAT 0x00 0x00 0x00 CRC1 CRC2]
```

  LEN is the 2 bytes length of the play load.
  ADDR is the 3 bytes address of the data to write to flash or EEPROM memory.
  DATA is the data to write.
  STAT is the return status.

15.4 Download command fragmentation

Whenever the platform puts restriction on the low-level frame size transmission, fragmentation can be used to split secure write command into several frames. This is the purpose of the chunk bit of the download mode protocol header.

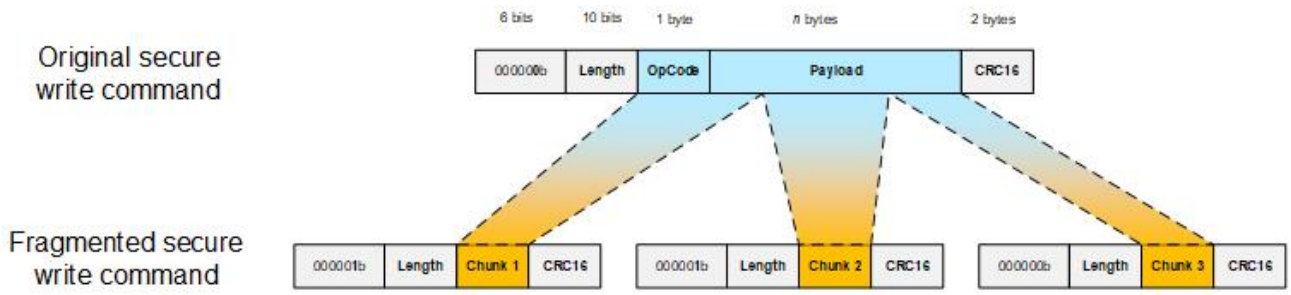

Figure 70. PN7160 download mode protocol

Each split command is then responded with dedicated status in case of success:

- PH_STATUS_DL_FIRST_CHUNK as response to the fist command of the fragmented sequence
- PH_STATUS_DL_NEXT_CHUNK as response to the next commands of the fragmented sequence
- DL_OK as response to the last command of the fragmented sequence if operation succeeds

15.5 Firmware signature and version control

In the PN7160 sFWu procedure, a mechanism ensures that only a firmware signed and delivered by NXP will be accepted. In case of signature mismatch an error DL_SIGNATURE_ERROR is responded to the first secure write command of the sequence.

Furthermore, the firmware update is only possible if the new firmware major version number is bigger or equal to the current one. If not, the first secure write command is responded with error code DL_FIRMWARE_VERSION_ERROR.

15.6 Degraded mode

During sFWu procedure, after the first secure write command passed, if any of the following events occurs the sFWu procedure will be incomplete:

- Reset (hardware or software)
- Hash chain broken during the download
- Protocol error in framing
- EEPROM critical failure

If previous sFWu procedure did not complete successfully, PN7160 will enter in degraded mode (except if DWL_REQ pin is set to 1, in that case PN7160 will enter in download mode). In this mode, all the commands (sFWu or NCI) will be responded with error code DL_SFWU_DEGRADED indicating a new sFWu procedure is required to recover from this state.

16 PN7160 Practical approach

16.1 Basic examples for Reader/Writer (R/W) Mode

16.1.1 R/W Mode (1 NFC Endpoint)

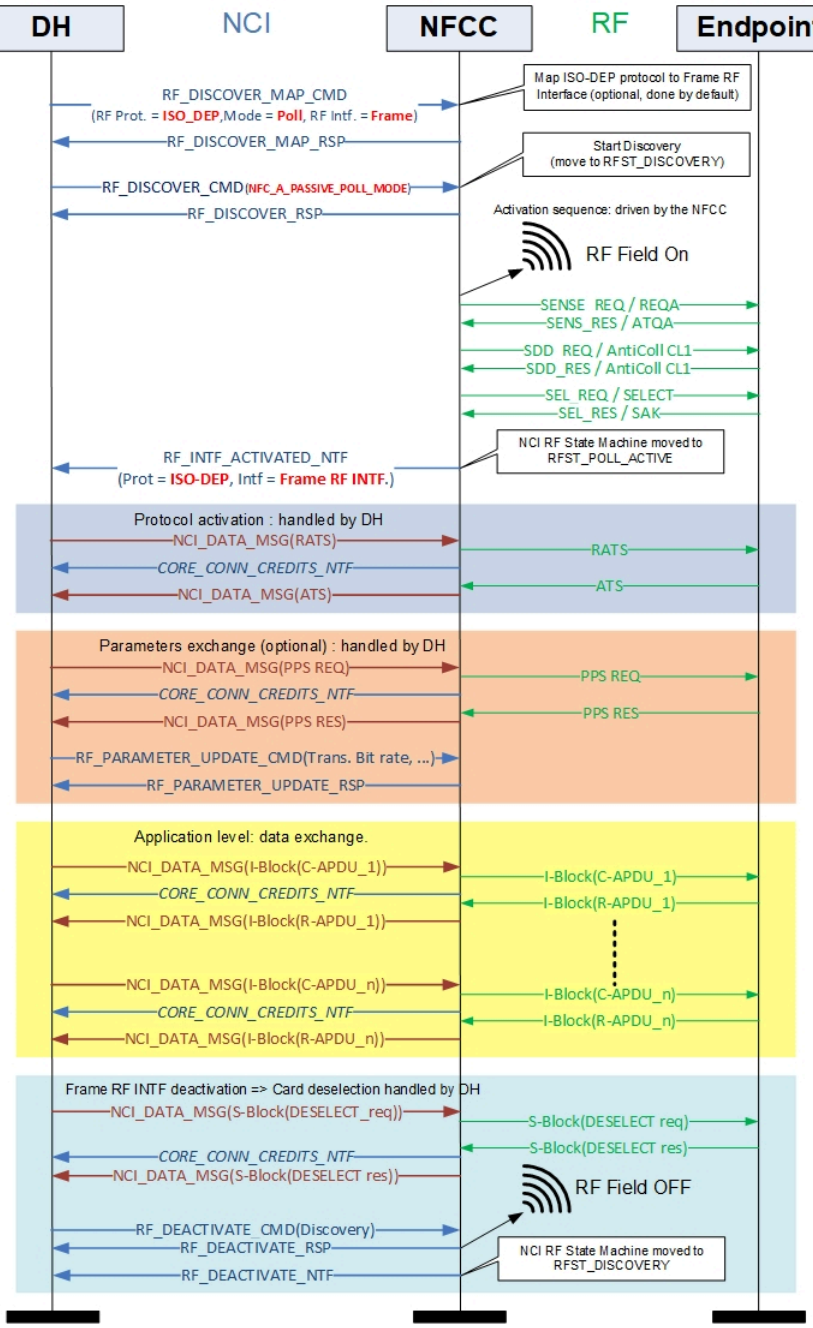

Figure 71. Poll Mode NFC-A, ISO-DEP protocol / RF Frame Interface

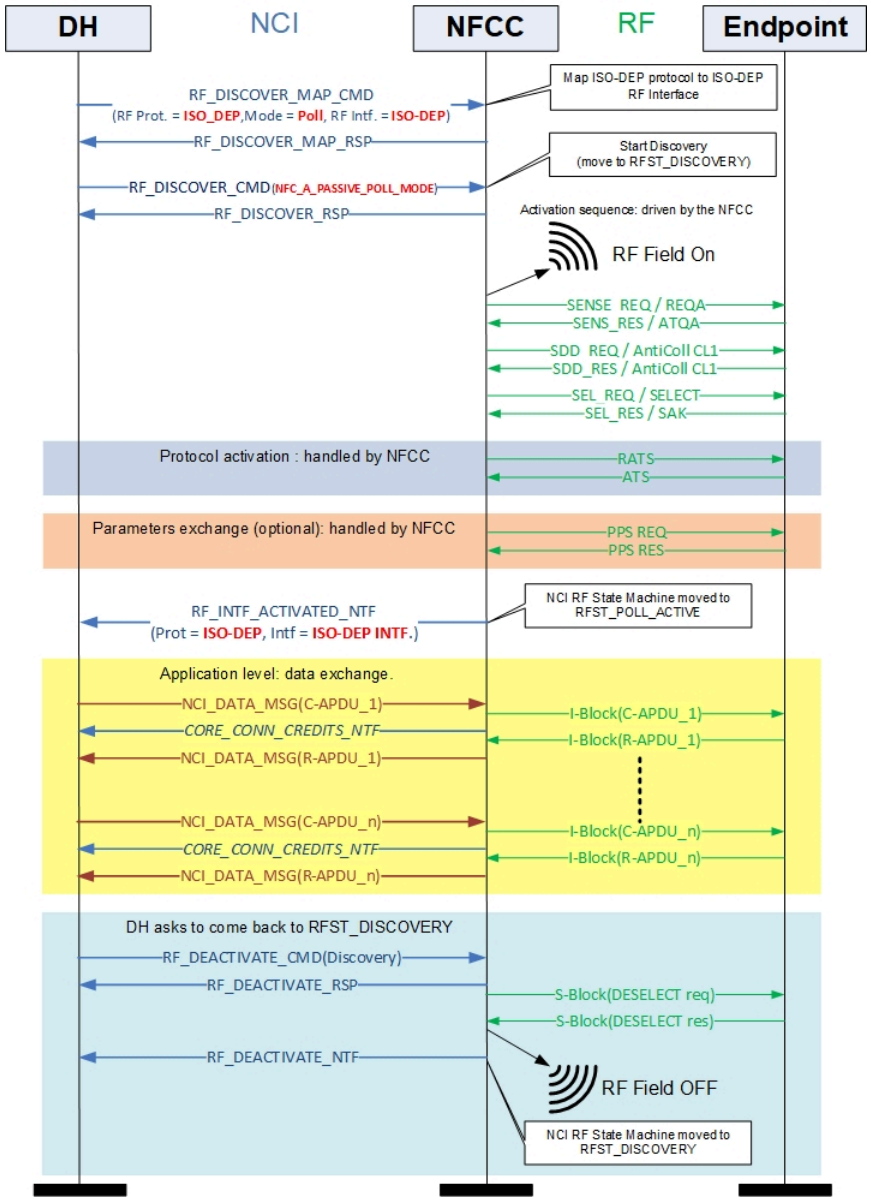

Figure 72. Poll Mode NFC-A, ISO-DEP protocol / ISO-DEP Interface

16.1.2 R/W Mode (2 NFC Endpoints)

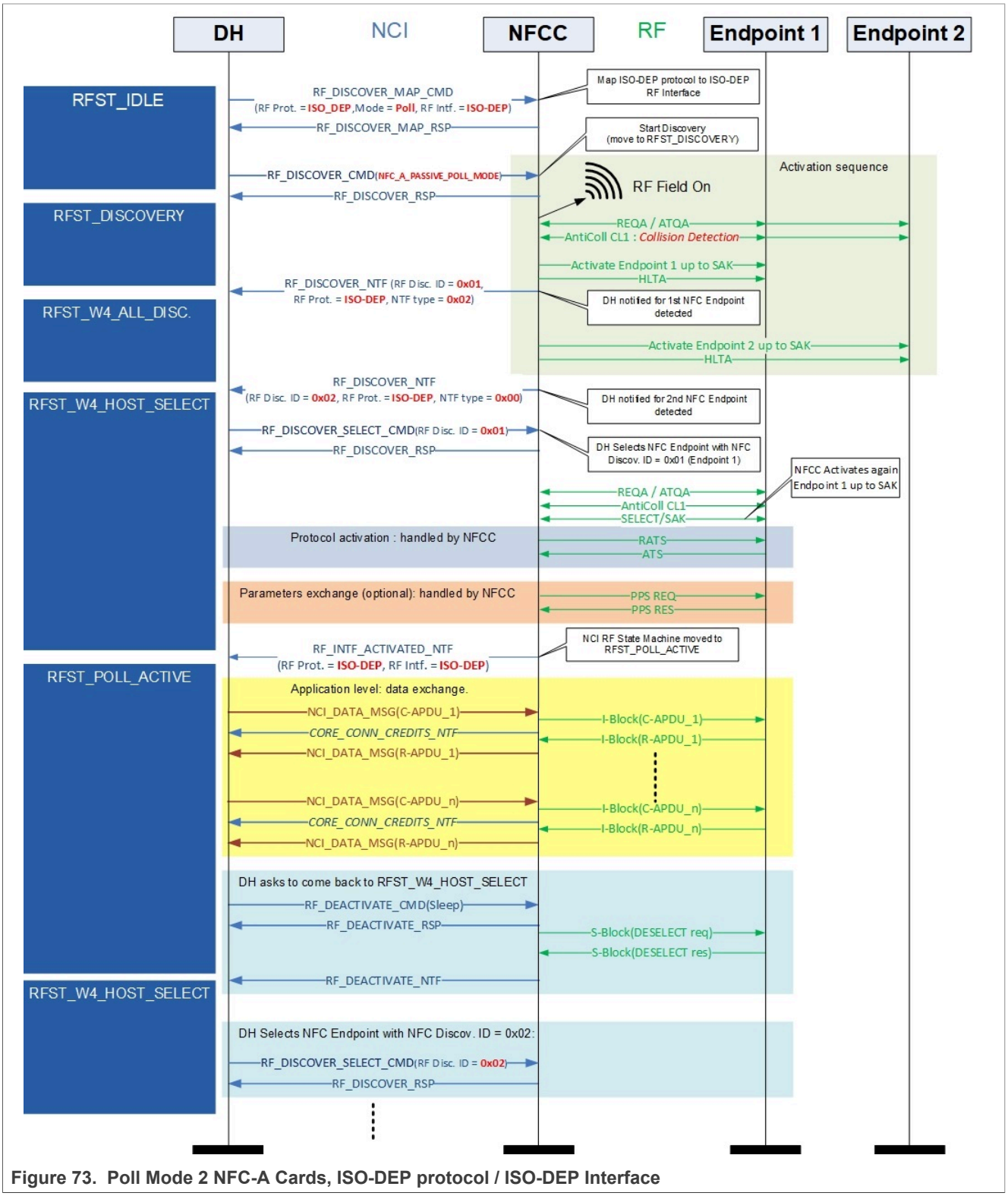

Figure 73. Poll Mode 2 NFC-A Cards, ISO-DEP protocol / ISO-DEP Interface

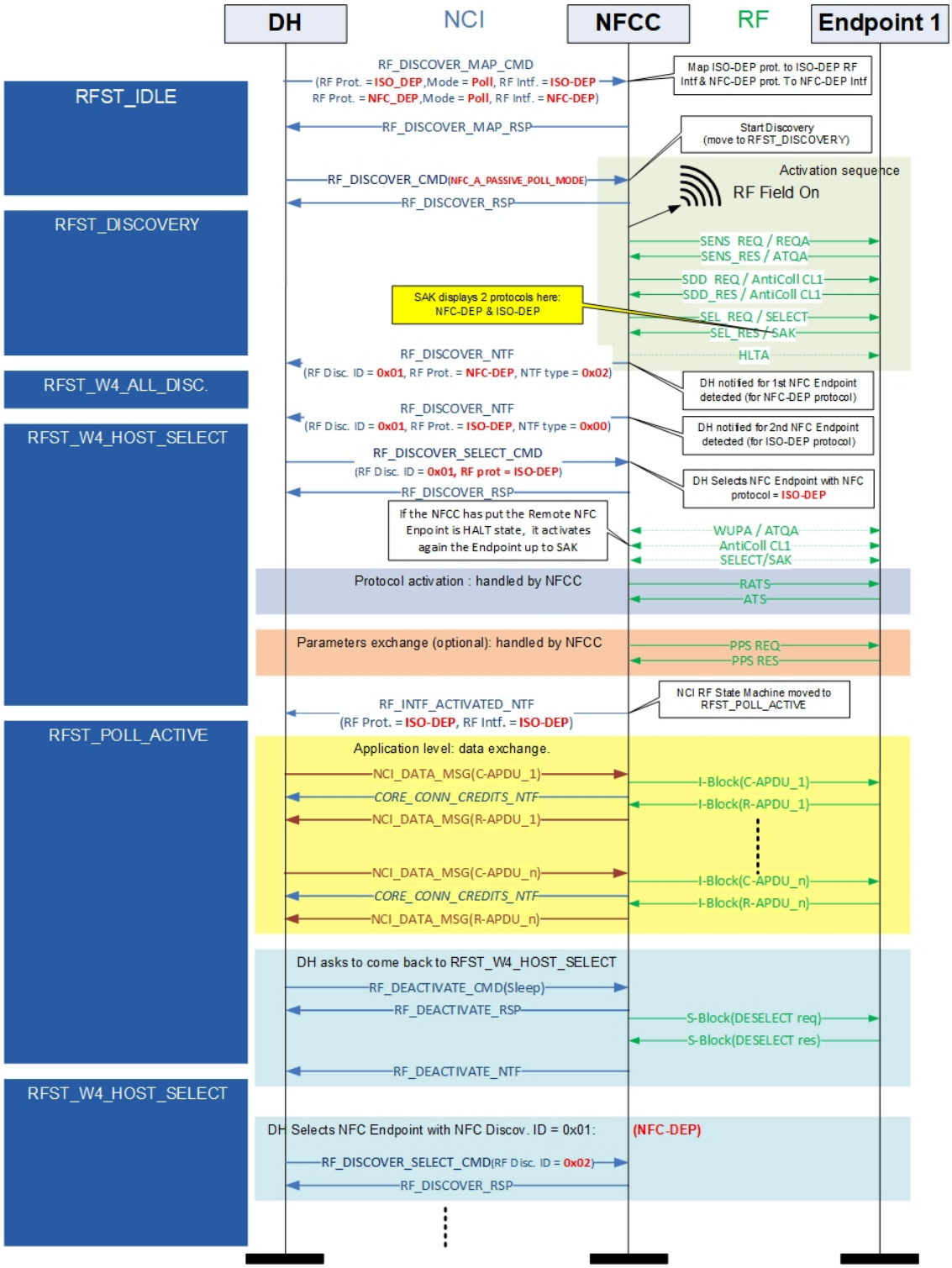

Figure 74. Poll Mode 1 NFC-A Device, 2 RF protocols (merged SAK)

16.2 Basic examples for Card Emulation (CE) Mode

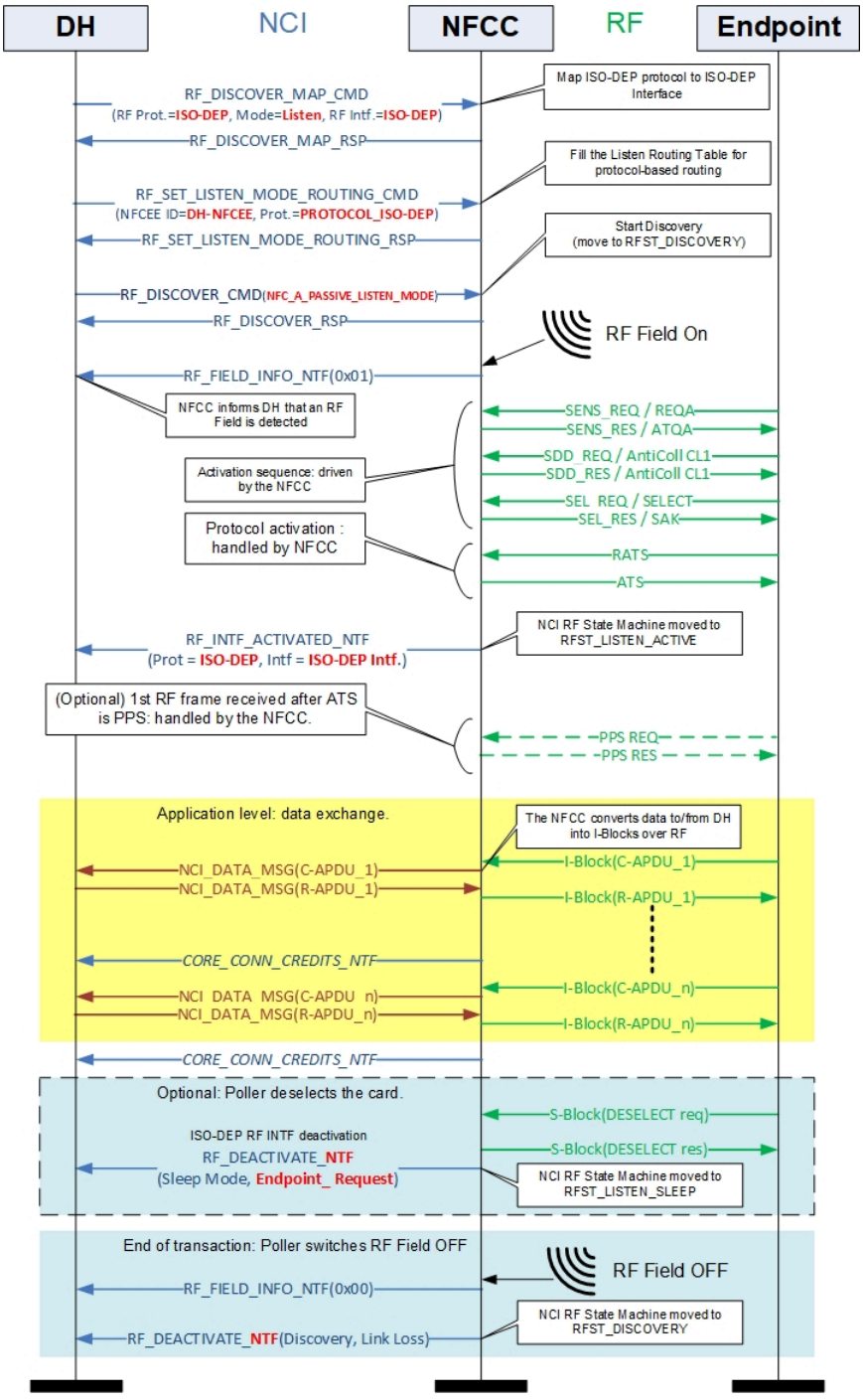

Figure 75. Card Emulation NFC-A, ISO-DEP protocol / ISO-DEP Interface, optional PPS

Note: this sequence is not according to NCI on the ISO-DEP RF interface activation versus the PPS reception. This sequence describes the NXP NFCC behavior, not NCI expectation: RF_INTF_ACTIVATED_NTF will always indicate 106kb/s.

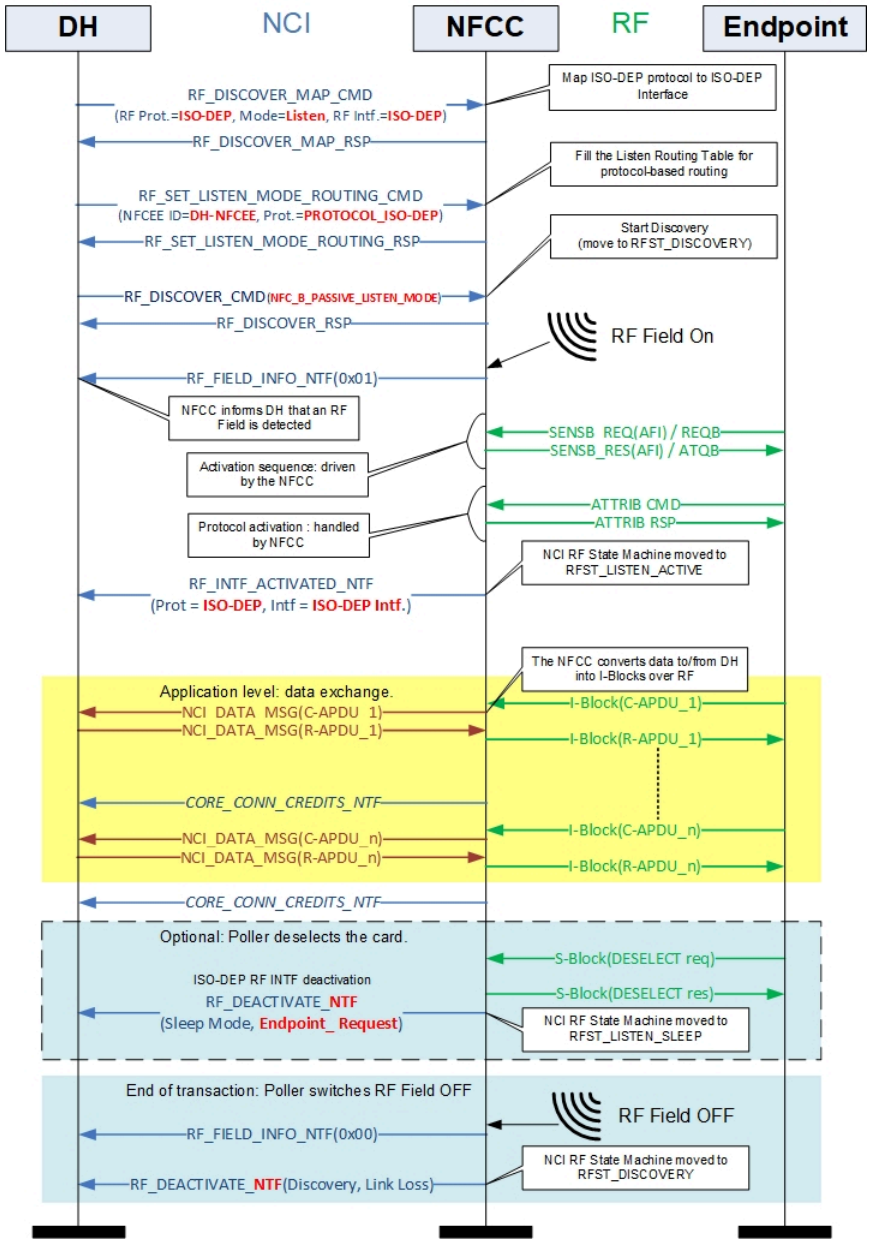

Figure 76. Card Emulation NFC-B, ISO-DEP protocol / ISO-DEP Interface

16.3 Basic examples for Peer-to-Peer (P2P) Passive Mode

16.3.1 Target in P2P Passive Mode

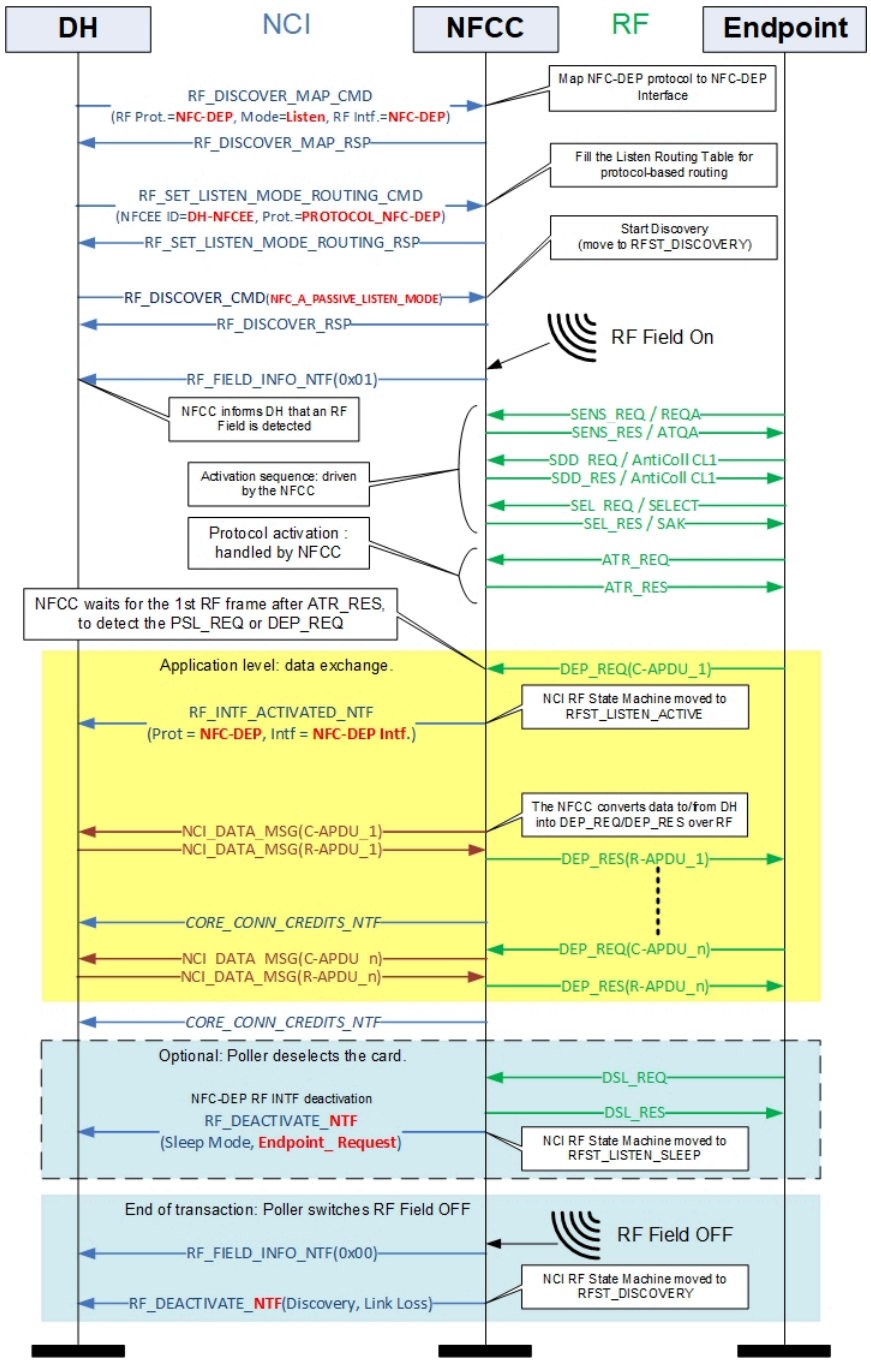

Figure 77. P2P/Target NFC-A Passive, NFC-DEP protocol / NFC-DEP Interface, NO PSL

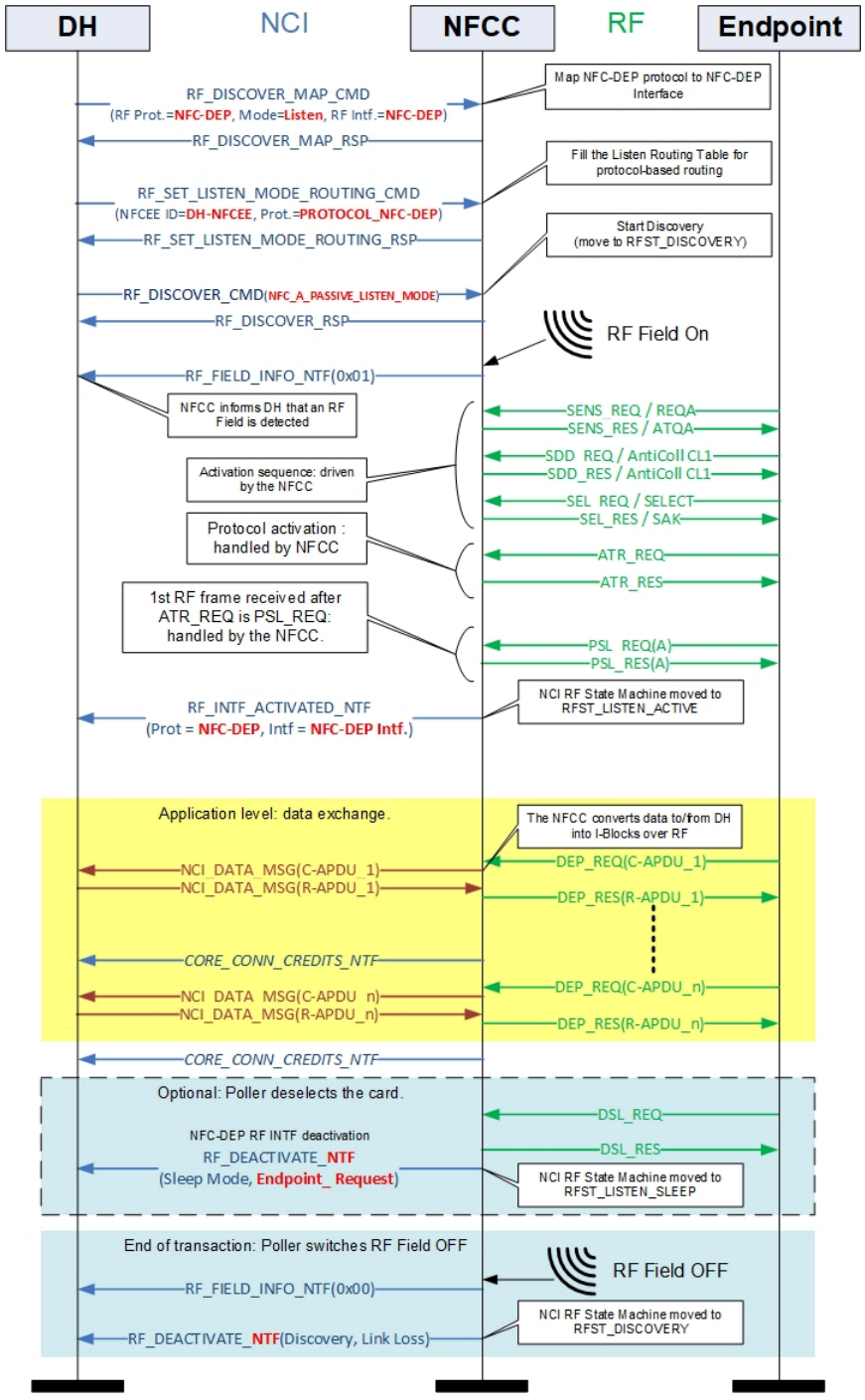

Figure 78. P2P/Target NFC-A Passive, NFC-DEP protocol / NFC-DEP Interface + PSL

16.3.2 Initiator in P2P Passive Mode

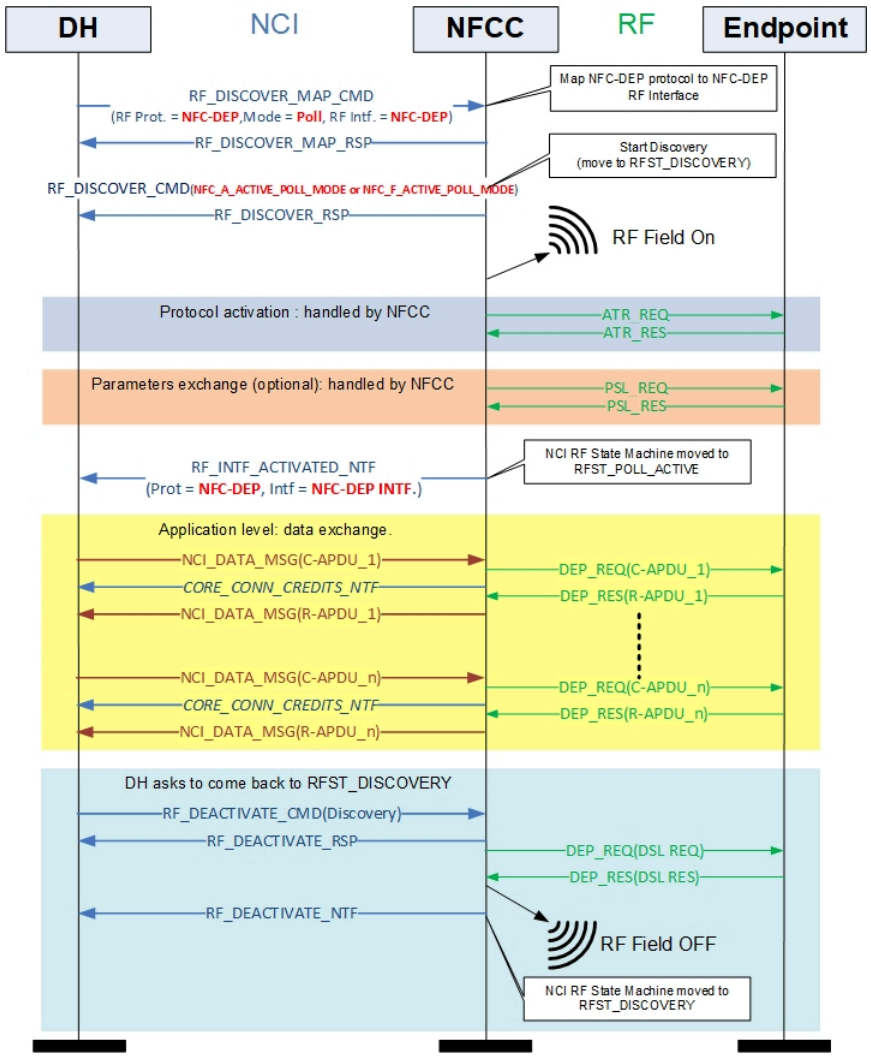

Figure 79. P2P/Initiator NFC-A Passive, NFC-DEP protocol / NFC-DEP RF Interface

16.4 Basic examples for Peer-to-Peer (P2P) Active Mode

16.4.1 Target in P2P Passive Mode

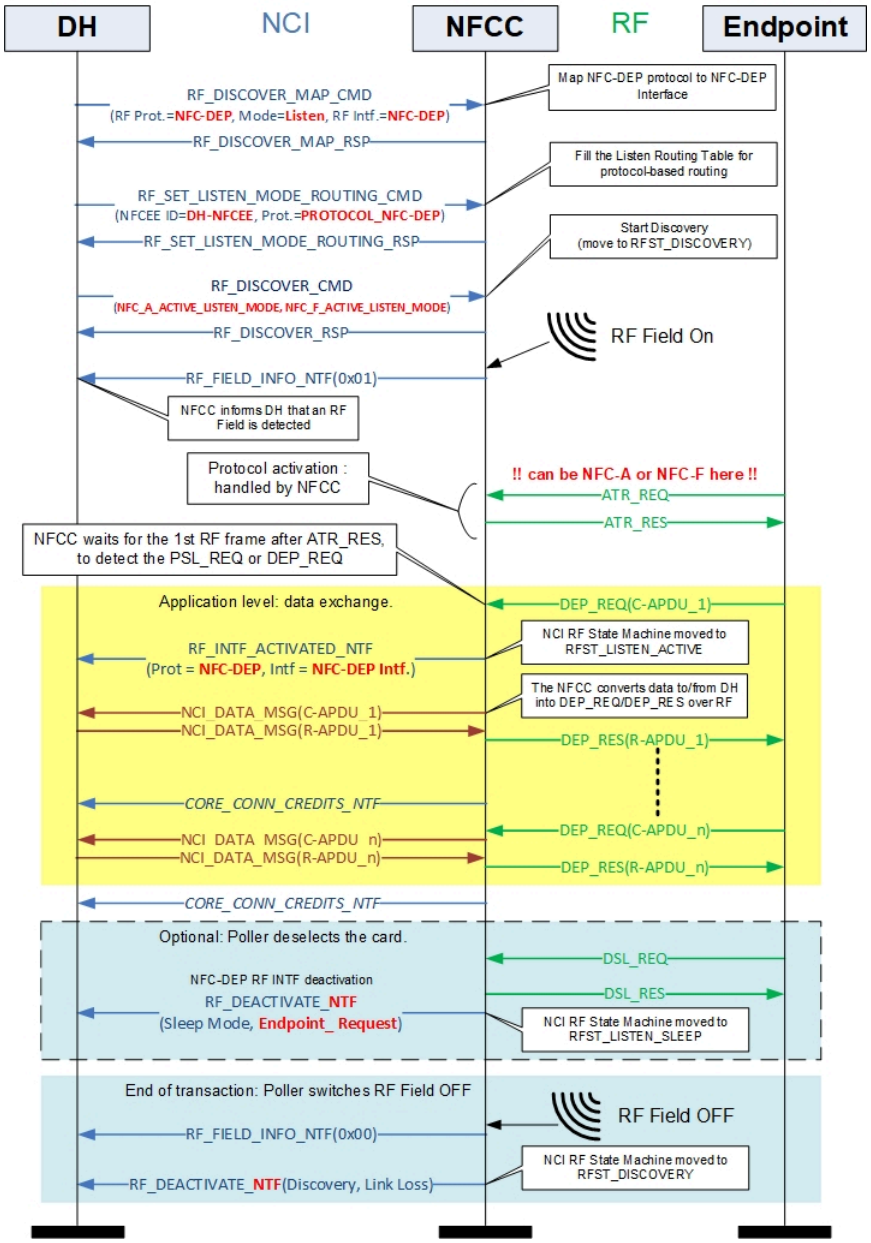

Figure 80. P2P/Target NFC- A or NFC-F Active, NFC-DEP protocol / NFC-DEP Interface, NO PSL

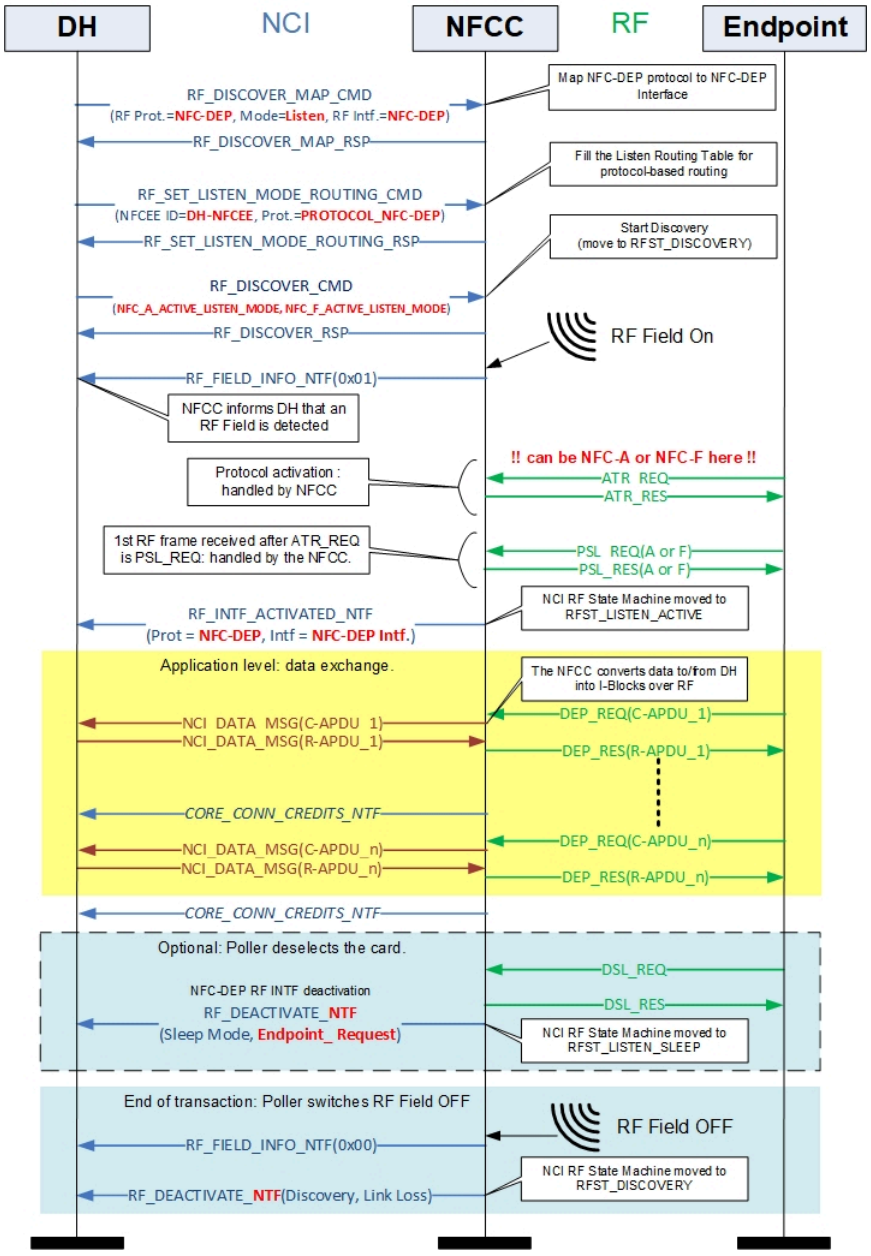

Figure 81. P2P/Target NFC- A or NFC-F Active, NFC-DEP protocol / NFC-DEP Interface + PSL

16.4.2 Initiator in P2P Active Mode

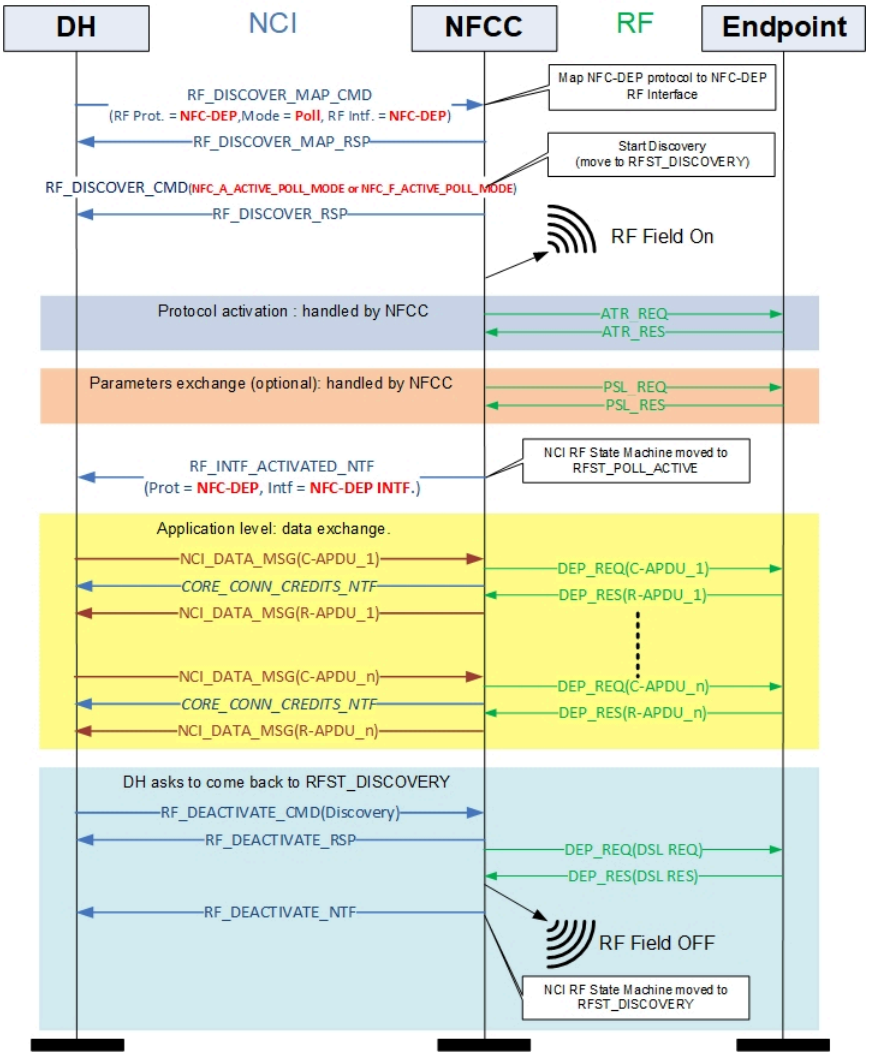

Figure 82. P2P/Initiator NFC-A Active, NFC-DEP protocol / NFC-DEP RF Interface

17 Annex A: details on RF state machine

The [NCI] RF State Machine is quite complex and the drawing proposed in the NCI technical Specification is combining all the different modes of operation in a single drawing.

For debug purposes, it is convenient to draw this State Machine in a simplified way, depending on the Protocol to RF Interface mapping applied by the DH. This is why the following figures are proposed here:

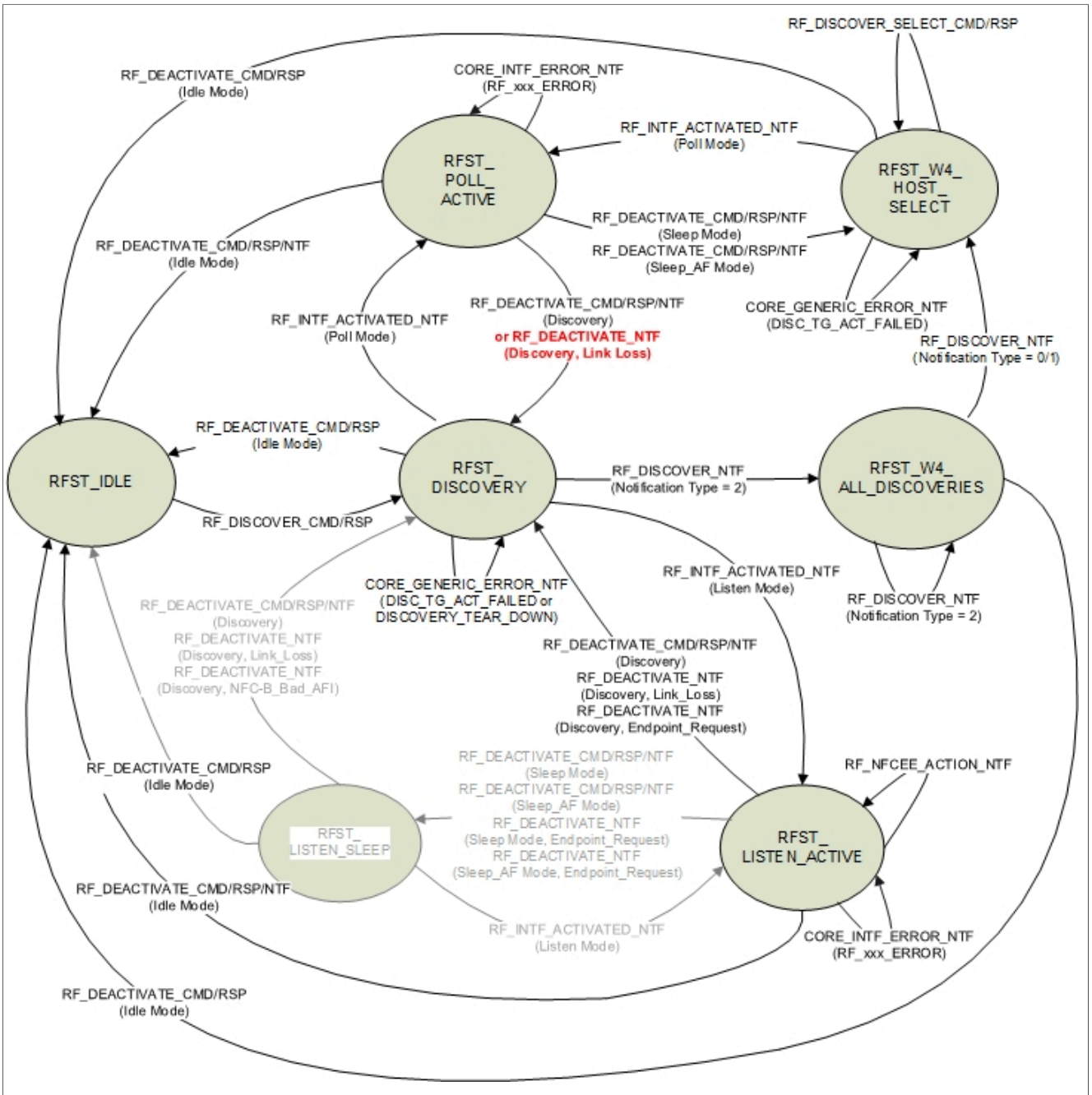

Figure 83. RF State Machine when the Frame RF interface is used

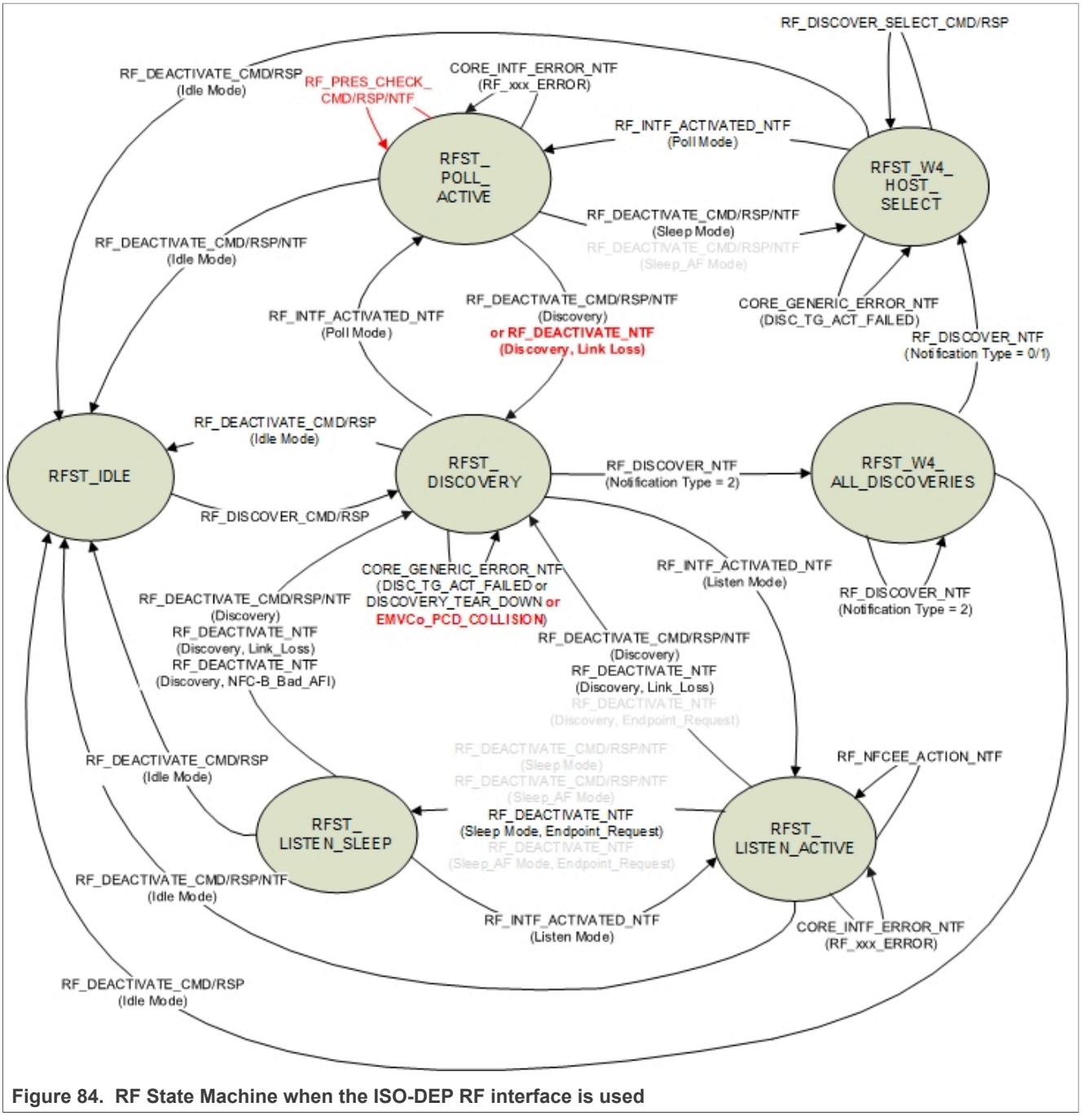

Figure 84. RF State Machine when the ISO-DEP RF interface is used

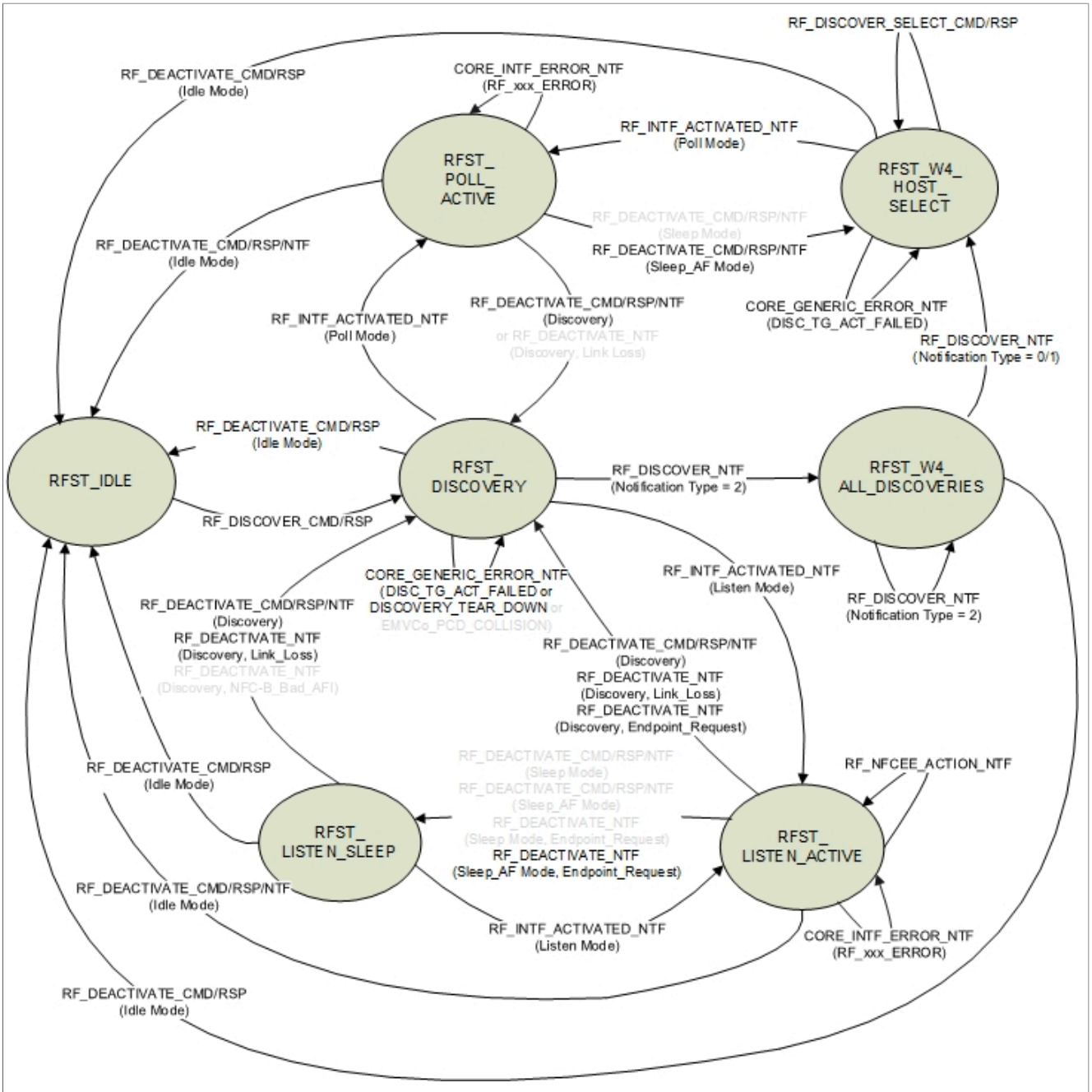

Figure 85. RF State Machine when the NFC-DEP RF interface is used

18 Annex B: List of [PN7160-NCI] extensions to Control Messages + List of TAG-CMD REQs and RSPs

Table 138. [PN7160-NCI] extensions to Control Messages

Chapter	GID	OID	[PN7160-NCI] Control message
→Section 12.6.1	1111b	0x00	CORE_SET_POWER_MODE_CMD CORE_SET_POWER_MODE_RSP
→Section 8.4.1	1111b	0x02	NCI_PROPRIETARY_ACT_CMD NCI_PROPRIETARY_ACT_RSP NCI_PROPRIETARY_ACT_NTF
→Section 9.3.4	1111b	0x11	RF_PRES-CHECK_CMD RF_PRES-CHECK_RSP RF_PRES-CHECK_NTF
→Section 12.4.3	1111b	0x13	RF_LPCD_TRACE_NTF
→Section 13.3	1111b	0x14	RF_GET_TRANSITION_CMD RF_GET_TRANSITION_RSP
→Section 12.7	1111b	0x15	SCREEN_STATE_CMD SCREEN_STATE _RSP
→Section 14.3	1111b	0x30	TEST_PRBS_CMD TEST_PRBS_RSP
→ Section 14.2	1111b	0x3D	TEST_ANTENNA_CMD TEST_ANTENNA_RSP

Table 139. List of REQs and RSPs

Chapter	ID	TAG-CMD REQ and RSP
→Section 9.1.7	0x10	XCHG_DATA_REQ XCHG_DATA_RSP
→Section 9.1.8	0x32	MF_SectorSel_REQ MF_SectorSel_RSP
→Section 9.1.9	0x40	MFC_Authenticate_REQ MFC_Authenticate_RSP

19 Legal information

19.1 Definitions

Draft — A draft status on a document indicates that 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 in a draft version of a document and shall have no liability for the consequences of use of such information.

19.2 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. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. 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 and its suppliers accept 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.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

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.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

19.3 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.

19.4 Trademarks

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

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

MIFARE — is a trademark of NXP B.V.

DESFire — is a trademark of NXP B.V.

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

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

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

JCOP — is a trademark of NXP B.V.

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

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

FeliCa — is a trademark of Sony Corporation.

Tables

Figures

Contents

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

Date of release: 13 September 2021
Document identifier: UM11495