AN108711

Tips and Tricks for PN51x Integration

Rev. 01.10 — 11. May 2006

Application note
108711

Document information

Info	Content
Keywords	Tips and Tricks for PN51x Integration
Abstract	This document provides some hints on how to design in the PN51x from a hardware, software and RF point of view. The main topics are: - How to customize the host controller interface - How to tailor and optimize the targeted NFC system

Revision history

Rev	Date	Description
1.0	20050411	First release along with BFL v3.1, Document in first version is named: Programmers Manual
1.1	May 2006	Added description about event / interrupt handling, added remarks for Target Mode. More consistent naming. Naming changed from PN511 to PN51x. The masking of the last number indicates that this Tips and Tricks for PN51x Integration Application Note is true for the whole PN51x family of NFC RF-Front-end devices except where stated otherwise.

Contact information

For additional information, please visit: http://www.semiconductors.philips.com

For sales office addresses, please send an email to: sales.addresses@www.semiconductors.philips.com

1. Introduction

1.1 Scope

This document provides on top of datasheet information about the PN51x[1]. The interdependency between hardware, software, RF-behavior and NFCIP1-protocol are the discussed topics in this application note.

This information is targeted for developers who want to Design-In the PN51x in an effective way.

It is not intended to describe all details of NFC (this is already covered in the data sheets and specifications) but it should give additional details, which are related to the PN51x Integrated Circuit. Access to the following documents is required.

- ISO/IEC 18092 see Bibliography
- ISO/IEC 14443-3 Standard ("Initialization and anti-collision") see Bibliography
- PN51x Data Sheet
- Philips NFC PN51x Design-In Kit

  In order to test the examples, which are covered in this document, at least the following items shall be accessible

  - Two PN51x demo boards including power supply and serial cable
  - A variety out of tags like Mifare® Standard 1k or 4k, Mifare® UltraLight, JCOP[2] – configured with factory settings

1. The naming PN51x indicates the family of NFC front-end devices available: Current products are PN511 and PN512.
2. JCOP stands for Java Card Open Platform which is the Philips OS for Smart-Cards

1.2 PN51x

This chapter provides a very short summary about the PN51x ICs capabilities. For more detailed information the data sheet should be consulted.

The PN51x is an integrated transmission module for contactless communication with the carrier frequency of f_c=13.56 MHz. The PN51x should be seen as a RF front-end, hence most of the non-critical protocol related items have to be done outside of the hardware, which is usually a device driver or the firmware running on a host controller.

The PN511 transmission module supports three different operating modes

- NFCIP-1 [ISO/IEC 18092] Initiator and Target Mode: The NFC standard offers different communication modes (active and passive) and the bit rates 106kBit/s, 212kBit/s and 424kbit/s. The digital part of the PN511 handles the NFC framing, mode detection and some very time critical parts of the anti-collision. The remaining part of the NFCIP-1 protocol has to be implemented in software.
- Reader/writer[3] mode for FeliCa^{TM}, MIFARE$^{®}$ and ISO/IEC 14443-4 cards: The PN511 supports all the communication capabilities, which are used in the FeliCa^{TM} and ISO/IEC 14443-3A and MIFARE$^{®}$ world. This includes higher data rates up to 424kBit/s and built in reader hardware encryption for the Mifare® Standard protocol. CRC and framing is handled by hardware, too.
- Card interface mode for ISO/IEC 14443-4 based cards: The PN511 is also able to act like an ISO/IEC 14443-4 smart card. Like in the reader/writer mode, the card mode supports higher data rates and several anti-collision mechanisms. Note that opposed to the reader/writer mode the card interface mode cannot emulate Mifare® products.

The PN512 transmission module supports all modes mentioned in the PN511 section plus

- ISO/IEC 14443 PCD Type B functionality. **Philips is giving support in Type B issues on RF side only – In any case the proprietary protocol has to be implemented by our customers by themselves.** All advice given in this application note deals with all NFCIP-1 modes and how to implement them correctly. ISO/IEC 14443 PCD Type B functionality and its implementation is not covered at all.

The NFCIP-1 protocol is defined by the ISO/IEC 18092 standard. The NFC protocol has been designed to be compatible with existing standards ISO/IEC 14443-3A, Mifare® and FeliCa^{TM} protocol. Figure 1 shows the overlapping of the NFCIP-1 protocol with the infrastructure which already existed before the availability of the ISO/IEC 18092.

3. The specifications use different terminologies, which is discussed later on. At this point it is sufficient to know that the term “reader/writer” (FeliCa), the term “PCD” (ISO/IEC 14443) and the term “Initiator” (ISO/IEC 18092) are the same.

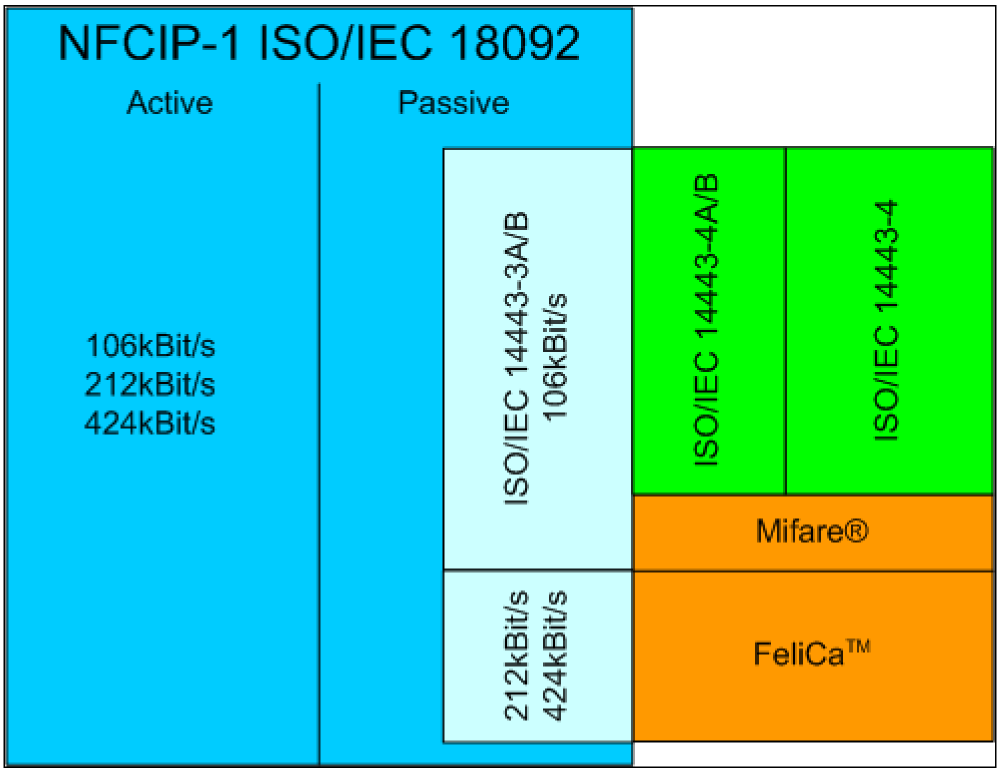

Figure 1 ISO/IEC 18092 Standard's Coverage

The figure also shows that there is no overlapping box in the active part. The active mode of the NFCIP-1 protocol is completely new, whereas the passive mode has many similarities with existing standards like ISO/IEC 14443. This document will especially cover those parts, which reside in the box on the right side (the passive mode with the backward compatibility to "older" technologies). Some parts of this document also refer to ISO/IEC 14443-4. Mifare® and FeliCa™ parts will not be covered in this document.

1.3 ISO/IEC 14443-3A

This standard describes the initialization and anti-collision procedure of contactless communication between a reader/writer (ISO terminology: PCD) and a card (ISO terminology: PICC). The physical layer is covered by the ISO/IEC 14443-2 standard, whereas the transmission protocol is defined in the ISO/IEC 14443-4 standard. The ISO/IEC 14443-3 covers the activation, both for the types A and B, but here only type A will be discussed since the type B is not supported by the NFC protocol[4].

The ISO/IEC 14443-3 standard defines that the card is powered by the RF field. This RF field is generated by the reader/writer. The PCD always acts as master, which initiates and controls the complete protocol and data flow.

1.4 ISO/IEC 18092

This protocol is split up into two major parts. In passive mode the Initiator is generating the RF field and the Target is using load modulation. This approach is compatible with existing smart card technologies like ISO/IEC 14443-4, Mifare or FeliCa. The second

4 Although type B hasn't been considered in the NFC protocol, it does not mean that Type B cards may confuse existing communication.

part, namely the active mode, shares the power between both Initiator and Taget. The device which is sending data is also the device which is generating the RF field.

Figure 2 shows the several paths, which are covered by the NFC specification. The diagram is seen from the Initiator point of view. The Initiator can freely decide on the operating mode. This is very similar to the role of the classic reader/writer. The left part covers passive communication, where even non – NFCIP-1 protocols are possible. In active mode the one and only reachable protocol to date is NFCIP-1.

In order to be fully NFC compliant all bit rates, all modes (Passive and Active) and all NFCIP-1 commands have to be supported.

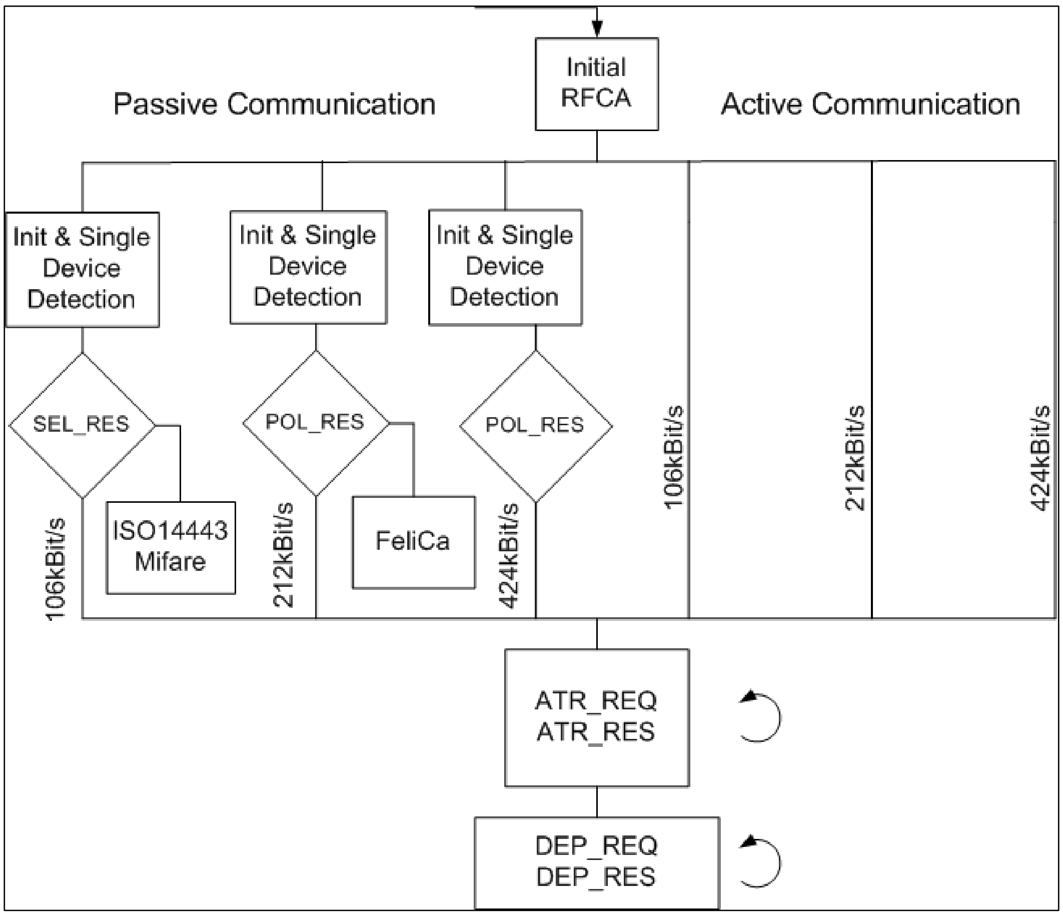

Figure 2 Reaching the NFC Protocol

Figure 2 shows that the Active Mode is the most straightforward (and fastest) way to reach the NFC protocol. The protocol flow is described in a more detailed way in the next chapter.

1.4.1 Passive Communication mode

In passive mode the Initiator first has to check if there is already an existing RF field. If yes, it is not allowed to act as Initiator. If no RF field is detected, the Initiator starts the communication by generating the RF field and sending the data according to the appropriate bit representation at the selected bit rate. The target then answers to an initiator command in a load modulation scheme, where the type of modulation also depends on the bit rate. The Initiator always generates the RF field for the target. As long as the RF field is on (and detected by the Target) the session context[5] is valid. If the RF field has been switched off (or the Target lost the RF field), a new session (including the initialization sequence) has to be started again.

5. The term “session context” is not specified in NFCIP-1, however, it will be used in this document to specify the actions between the beginning and the end of a communication

Three different transfer speeds are defined in the passive communication:

- 106 kBit/s,
  is used for NFCIP-1, Mifare® and ISO/IEC 14443-4 smart cards. The decision, whether NFCIP-1 or others are taken is done by checking the SEL_RES (ISO/IEC 14443-3: SAK) byte. If the appropriate bit is set the standard defines the counterpart to be NFCIP-1 compliant.

- 212 kBit/s,
  is used for the NFC protocol and for the FeliCa™ protocol. The decision, whether NFCIP-1 or FeliCa™ is taken, is done by checking certain bytes in the first response of the counterpart.

- 424 kBit/s,
  is reserved for future FeliCa™ protocol and for NFCIP-1. As currently (May 2006) no smart card (FeliCa™) is supporting this speed the counterpart can only be a NFCIP-1 compliant device - nevertheless the bit check according to ISO/IEC 18092 should be done to avoid problems in future.

When the NFC communication has been set up (ATR_REQ and ATR_RES successful), the bit rate can be negotiated by the optional parameter selection (PSL_REQ and PSL_RES) command. It is possible to perform the communication in an asymmetric manner (different data rates for transmitting and receiving).

1.4.2 Active Communication mode

The active communication mode is exclusively for the NFCIP-1 protocol. Since this scheme doesn't have to consider other protocols, the setup mechanism is more straightforward. The Initiator can define the transfer speed without any preceding negotiation process. If there is a response, it can be assumed that the counterpart is also an NFC device. This device, acting as Target simply detects the communication speed by hardware and responds with the same data rate.

Like in passive mode the precondition is that no RF field shall be detected before the RF is switched on. Therefore RF detection has to be done before any command or any response is issued, because the RF is only switched on when data will be transmitted. However, the PN51x provides means to detect an RF field by itself.

Three different transfer speeds are defined for the active communication mode:

- 106 kBit/s
- 212 kBit/s
- 424 kBit/s

The speed can be adjusted during communication by using the PSL command. Like in passive mode, asymmetric communication is possible after initialization.

1.4.2.1 Initial RF Collision Avoidance / Detection

The RF Collision Avoidance (RFCA)[6] in active mode is based on two techniques:

- RF field sensing
  In discrete time frames the Initiator / Target checks if there is already an RF field - or during communication - if there is a violation due to an invalid CRC.
- Time Jitter
  The initiation of certain commands depends on a randomly chosen time offset. In the NFCIP-1 protocol there are four possible discrete time frames which are selected on a random basis. This prevents having deadlock situations.

The RFCA is performed as follows:

The Initiator first checks if already an RF field can be detected. If there is no RF field for a time greater than T_{IDT} , plus a random time frame (n * T_{RFW}), the initiator activates the RF field. This time frame gives other initiators the chance to detect that another Initiator has already started the RFCA (RF Collision Avoidance) sequence. Now the initiator can send an initial ATR_REQ. After the command has been issued, the initiator switches off the RF field and waits for an ATR_RES from at least one target. The initiation of the ATR_RES follows the same strategy as for the ATR_REQ. This means that the target also starts at a random time frame and also checks if there is already a target, which has already switched on the RF field. If the Target has already detected an RF field (i.e. there is another Target in the RF field) it won’t respond by an ATR_RES.

Although the RFCA is sufficient in most cases, it sometimes can happen that at least two initiators or targets are modulating at the same time. In order to detect these conflicts, the PN51x checks the hardware calculated CRC values from the ATR_REQ and ATR_RES. If there was an error the selection process has to be repeated.

After the ATR_REQ/ATR_RES pair has been established, collisions shouldn’t occur any more. Therefore the time-jitter, which is generated by hardware, is switched off. However, according to NFCIP-1 CRC checks shall be performed throughout the whole communication.

1.4.3 Bit Representation

The bit representation depends on the mode and on the bit rate. Table 1 shows the several combinations, which are covered by NFCIP-1. The modulation scheme has to be set manually, but this functionality is already provided by the BFL.

Table 1: Modulation Schemes

Operating Modes	Speed in kBit/s	Initiator	Target
Passive Mode	106	Miller	Manchester
	212, 424	Manchester	Manchester
Active Mode	106	Miller	Miller
	212, 424	Manchester	Manchester
ISO/IEC 14443-4	106, 212, 424, 848	Miller	Manchester

6. The RF collision avoidance has the same purpose as the anti-collision mechanism in the ISO/IEC 14443-3 standard. However, the way how to select the counterpart is different.

2. PN51x Operating Modes

This chapter shows how the specifications in the previous chapters can be connected together and how they are applied on the PN51x. It is assumed that the BFL[7] is used to build up the system or at least to have it as a reference. Besides several boundary conditions (used microcontroller, interface ...) several decisions have to be made how the BFL must be customized. Hence this chapter will also show which adoptions have to be taken to have a sound NFC solution.

The PN51x can obtain one out of two roles. One is the Initiator, which controls the communication, the other is the Target which answers to the requests coming from the Initiator. If more than two devices are involved still one device is the Initiator and all the others are Targets.

Following facts distinguish these roles:

- Initiator
    - Can decide if passive or active mode shall be used
    - Can decide which bit rate shall be taken
    - Can decide when a request is issued (there is no timeout between a response and a request)
    - Has to consider certain parameters which are conveyed by the Target (like timeout and buffer size)
    - The Initiator is a sort of a master, which actively searches and communicates with a Target.
- Target
    - Has to accept and respond both in active and passive mode
    - Has to accept and respond in all three different bit rates
    - Can specify a timeout between a request and a response
    - Has to respond to a command within a certain timeout
    - Has to consider certain parameters which are conveyed by the Initiator (like buffer size)
    - Has to come back into the Receive mode after it has sent a response.
    - The Target is a sort of a slave, which waits until an Initiator issues a request. A slave is not allowed to act without any request from an Initiator.

The PN51x supports both roles. The device can simply be reconfigured by setting the appropriate registers.

7. BFL stands for Basic Function Library, this software is delivered within the NFC PN51x SDK package

2.1.1 PN51x acting as Initiator

This section shows a protocol flow for the Initiator role. As discussed previously the selection process in the passive section is not that straight forward compared to the active one due to backward compatibility. The bit checks how to identify other than NFCIP-1 technology are described in the corresponding standards.

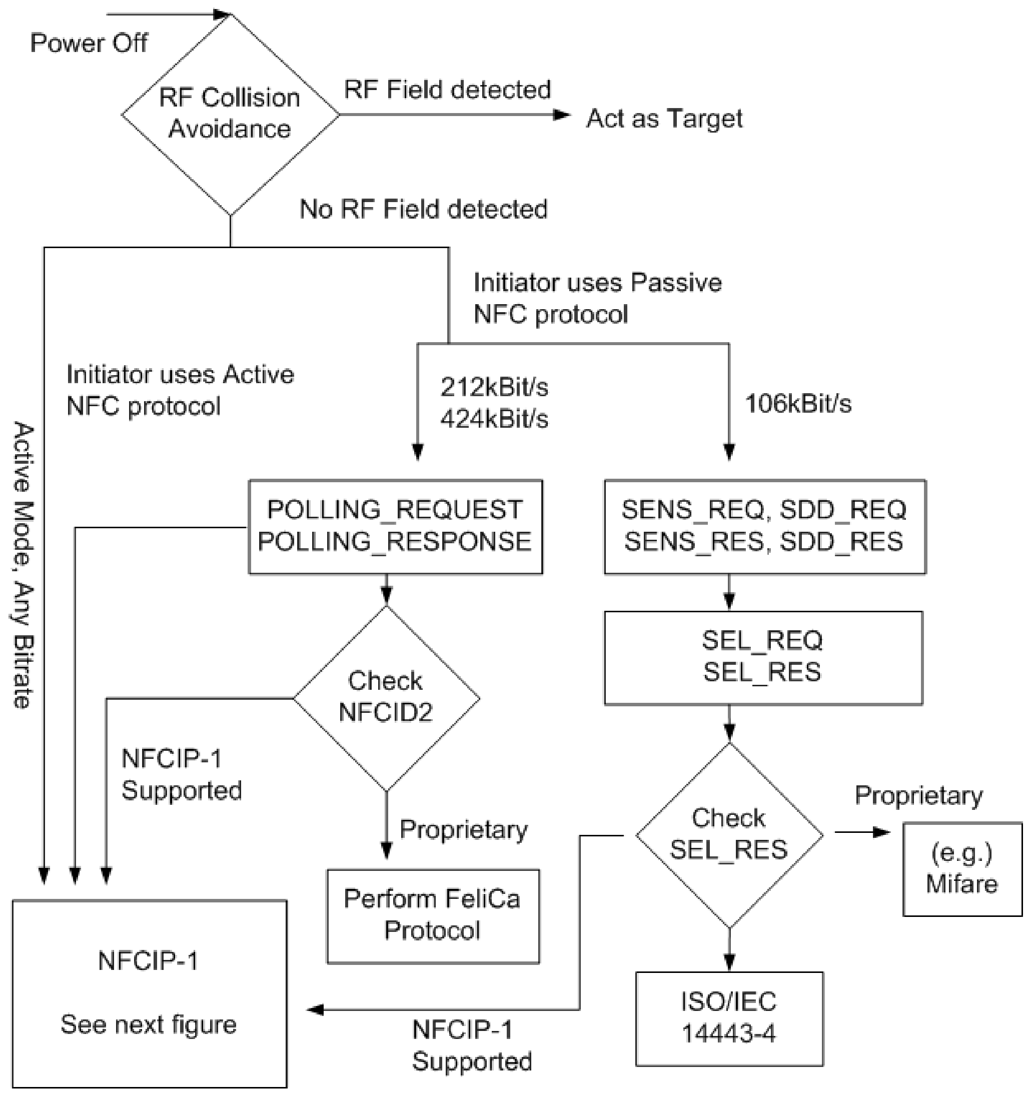

Figure 3 Initiator NFCIP-1 Protocol Flow

<u>How to find the references given in the picture:</u>

“Act as Target” is followed up in section 2.1.3

“NFCIP-1 See next figure” references to section 2.1.2

When acting as NFC initiator the flow could look as follows:

- In the first step the host controller decides which mode (active or passive) will be used.
    - If active mode is chosen, the left branch has to be followed. The host controller can freely select the desired bit rate and can start with an ATR_REQ command.
        - If a collision occurs (detected by an invalid CRC), resend an ATR_REQ and evaluate the ATR_RES as long as the CRC is not valid.
    - If passive mode is chosen, the initiator's host has to decide which bit rate will be used.
        - If 106kBit/s is used, the rightmost branch has to be followed. The branch until SAK is fully compliant to the ISO/IEC 14443-3A specification. After selection of a target the SEL_RES byte has to be checked.
            - If bit 6 of SEL_RES is set to 1, the counterpart is a device capable of handling the NFCIP-1 protocol.
            - If bit 5 of SEL_RES is set to 1, the device can handle the ISO/IEC 14443-4 protocol.
            - If none of these is set the counterpart is a proprietary device. This is then likely to be a Mifare card.
        - If either 212kBit/s or 424kBit/s is selected, the protocol starts with the FeliCa compatible POLLING_REQUEST. The response, namely POLLING_RESPONSE can be used to determine if the counterpart is capable of handling the NFCIP-1 protocol.
        - If the first two bytes of the NFCID2 is equal to 01h FEh the device supports NFCIP-1.
        - Otherwise it does not support NFCIP-1 and is likely to be a FeliCa card.

2.1.2 NFCIP-1 Device in Initiator Mode

This chapter shows the typical flow of an NFCIP-1 based communication. It is a refinement of the NFCIP-1 box showed in Figure 3.

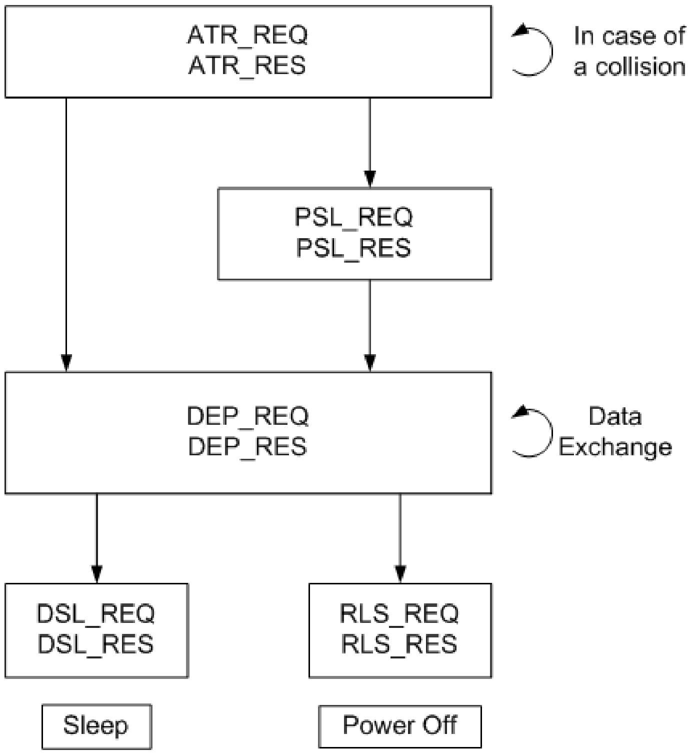

Figure 4 Initiator NFCIP-1 Protocol Flow (continued)

- The first common command (in terms of the several operating modes) is the ATR_REQ. If this command is recognized by the Target is shall send back an ATR_RES. If a collision occurs, the ATR_REQ/ATR_RES pair has to be repeated as long as the CRC is not okay.
- If the Initiator then wants to change a parameter (Frame Size, Bit Rate) it will issue a PSL_REQ. This has to be acknowledged by a PSL_RES. This command pair is optional.
- The DEP_REQ/DEP_RES block is the part where the payload exchange is performed. This pair can be exchanged as long as data is pending.

- After exchanging data the following commands can be issued to deactivate the connection between Initiator and Target:
    - Deactivating the connection by sending a DSL_REQ
      is especially useful if Active Mode is used. A De-Selection causes the Target to set itself into an intermediate state, which enables the Initiator to wake up the Target very fast, because it bypasses the initialization phase.
    - Deactivating the connection by sending a RLS_REQ
      causes the Target to fall back to the origin state. This means that if the connection wants to be established again, the process has to be started from the very initial point. A new NFC ID shell be generated for each new session.

2.1.3 PN51x acting as Target

This section shows the typical flow of a target. The part with non-white (see Figure 5) background is done by the IC's hardware; the other part has to be provided in software.

After having initialized the PN51x for Target Mode, the device waits for incoming commands. Commands which are related to initialization and/or are too complex or very time critical (default timeout < 1ms) are automatically handled by hardware. All the other commands have to be handled by the software / firmware.

The sequence below shows the typical negotiation in Target Mode:

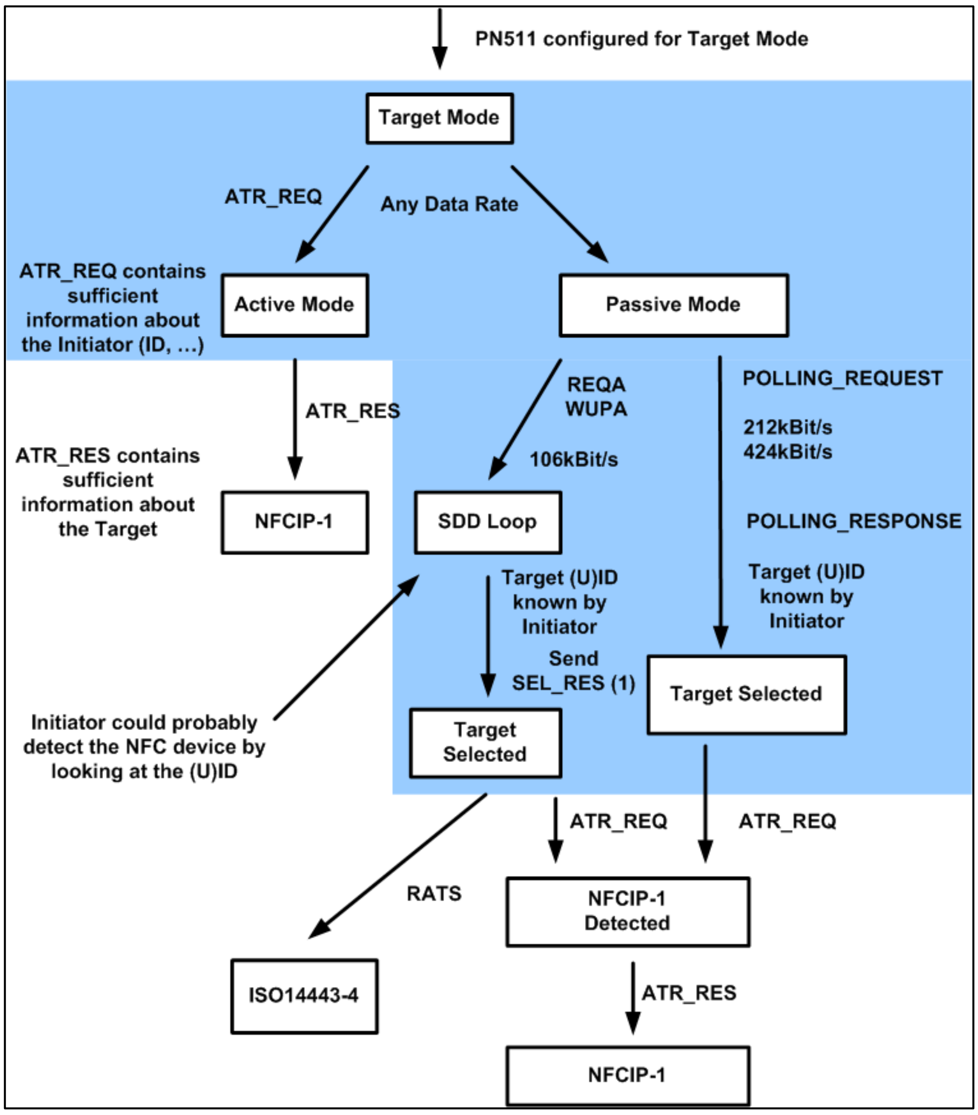
PN511 configured for Target Mode
Target Mode
ATR_REQ
Any Data Rate
ATR_REQ contains sufficient information about the Initiator (ID, ...)
Active Mode
Passive Mode
ATR_RES
REQA
WUPA
106kBit/s
POLLING_REQUEST
212kBit/s
424kBit/s
ATR_RES contains sufficient information about the Target
NFCIP-1
SDD Loop
POLLING_RESPONSE
Target (U)ID known by Initiator
Target (U)ID known by Initiator
Send SEL_RES (1)
Target Selected
Initiator could probably detect the NFC device by looking at the (U)ID
Target Selected
ATR_REQ
ATR_REQ
RATS
NFCIP-1 Detected
ISO14443-4
ATR_RES
NFCIP-1

Figure 5 Target NFCIP-1 Protocol Flow

- Configure the PN51x as Target

  The PN51x is now in “AutoColl Mode” (in PN51x Data Sheet, see Bibliography) and waits for incoming commands. Time critical commands which can be interpreted by the PN51x are now handled by hardware. If a command has been received but is not in the set of the time critical commands the PN51x writes the data into the FIFO and sets the RX interrupt. In case of an unknown command it is up to the software to decide how to proceed.

  Following commands can be interpreted by hardware:

  - 106kBit/s, Passive Mode: SENS_REQ, ALL_REQ, SDD_REQ, SEL_REQ according to ISO/IEC 14443

  - 212, 424kBit/s, Passive Mode : POLLING_REQUEST

- If the RX interrupt has been set by the PN51x following scenarios shall be considered:

  - The command is an ATR_REQ

    If the command is a valid ATR_REQ, the mode and the bit rate shall be checked and an ATR_RES shall be prepared and sent within the specified default timeout.

  - Any other command

    The PN51x does not interpret the frames sent by the initiator. Hence the target has to filter out not NFCIP-1 related commands (e.g. RATS out of ISO/IEC 14443-4) as well as not to this target dedicated (different DiD) but NFCIP-1 compliant frames.

    In case of an unrelated command PN51x has to be reconfigured because the received data may have altered some relevant registers.

3. System Constraints

3.1 General

This chapter describes the software and non PN51x hardware requirements to meet the RF-timings specified from the **host controller link** point of view. Further – in order to be compatible with “old” infrastructures - the ISO/IEC 14443-4A and Mifare® specification will be taken under consideration. The BFL tries to follow all these recommendations as the sample piece of software which Philips delivers. Due to the RS232 interface which is the default configuration problems can occur due to the circumstances described below.

The chapter is split into three different topics:

- Host Interface Related Parameters
  which may have effect on the software for the Initiator mode and the Target mode
- Initiator Mode Constraints and Requirements
  which are relevant for the Initiator
- Target Mode Constraints and Requirements
  which are relevant for the Target

3.2 Host Interface Parameters

The host controller interface has a strong impact on the performance of the host software.

The two most important performance metrics which have to be considered are the **data rate** and the **host response time**. The table below provides **approximate** register operation duration times.

The columns for ReadRegister and WriteRegister comprise a complete interaction (on registers level) between the host controller and the PN51x. The MReadRegister is applicable for certain interfaces, which support consecutive read operations on the same register address.

Table 2: Data Rates and Timings on the Host Interface

Interface	Baud rate	Read Register	MRead Register	Write Register	MWrite Register
UART	9600[8]	2000µs	N.A.	3100µs	N.A.
UART	115.200[9]	160µs	N.A.	260µs	N.A.
I2C Fast Mode	400.000[10]	80µs	20µs	60µs	???
SPI	≤5M	≥3.2µs	≥1.6µs	≥4µs	???
SPI[11]	1M	16µs	8µs	20µs	???

For an example how to use these numbers in some rough calculation please refer to 3.6.2.1 below.

8. This is the default baud rate of the PN51x
9. This is the maximum baud rate supported by a typical PC UART
10 The I2C numbers do not consider selecting the device itself
11 This is an example. The clock (respectively the baud rate) can be configured by the host controller (the master)

Consecutive read / write operations (MRead) are especially useful for the following situations:

- Reading and writing from / to the FIFO buffer
- Consecutive reading of an interrupt register

3.3 Initiator

The following paragraphs highlight the constraints which have to be considered for the Initiator mode.

3.3.1 General

The initiator is subjected to timing constraints which can be avoided through proper NFC-system configuration. The following operations especially do need a close look and investigation on the whole system.

3.3.2 Big Data Packages

On PN51x products we have the 64bytes hardware FIFO which is filled and emptied by the host controller. Big data packages consist of more than these 64bytes (e.g. Short APDU in the T=CL protocol, ISO14443-4 with a length of 256bytes)

If the payload (including command bytes) exceeds the buffer size of the HW-FIFO (64 bytes), the host controller has to write to or read from the register while transmitting the command over the RF field. To enable this PN51x provides configurable water - levels for the FIFO to signal the host controller when the FIFO reaches a lower or higher border. This prevents handling at absolute timings, but still it has to be guaranteed that the FIFO is filled / emptied in time. Remember from one point above: The host controller interface has to be fast enough – it is a precondition to this software constraint!

How to handle it:

The ISO/IEC 18092 provides means to define maximum buffer sizes. This maximum buffer size value is interchanged between Initiator and Target in the initialization phase (by conveying the appropriate values in ATR_REQ and ATR_RES).

Restricting the buffer size to 64 bytes (which is anyway the default) this constraint can be eliminated easily. Of course, it should be considered that this could lead to a performance drop due to necessary chaining - more protocol overhead is transceiver for synchronization purpose.

3.4 Target

The following paragraphs highlight the constraints, which have to be considered for the Target mode.

3.4.1 General

The implementation of the Target is narrowed by a number of timing constraints. Furthermore there are some other non-timing constraints, which also shall be taken into consideration.

3.4.2 Big Data Packages

The same applies as it is described on the Initiator side (see chapter 3.3.2)

3.5 Card Interface Mode

The following restrictions apply, when the PN51x is used in card emulation

- No real “UID”[12]
  The ID can be arbitrarily configured by the host controller. Although the probability is very low ($1/2^{24}$), it could occur that two targets choose the same ID. The Initiator will not detect any collision in the anti-collision phase. Hence a new ID shall be generated after every session in order to prevent collisions over several sessions. For this purpose the random number generator was included in the PN51x design.

- The first byte of the ID is fixed and has the value 08h
  which is according to ISO/IEC 14443-3A and prevents UID clashes with existing smart cards.

- No Cascade Level 2 and 3
  The PN51x only supports IDs for Cascade Level 1. Some tags like Mifare® UL use a double UID (7bytes UID). A PCD which receives the first part with the cascade bits set (indicating CL2) will ask for the second part of the UID which PN51x is not able to deliver.

3.5.1 Mifare Crypto

The Mifare encryption is only implemented for reader mode. There is no way to emulate the card–side cryptography as the algorithm is not disclosed.

3.6 Timings

3.6.1 ISO/IEC 14443-3A

The initialization phase is completely performed by the PN51x. The needed information (ID, SEL_RES…) can be configured beforehand. The selection of the card is signaled by an interrupt (if so configured). After that the host controller has to interact with the PN51x in order to follow the ISO/IEC 14443-3A protocol flow.

3.6.2 T=CL (ISO/IEC 14443-4) Card (106kBit/s)

Data exchange is covered by the ISO/IEC14443-4 (“Transmission Protocol”). This protocol is entered beforehand by performing the anti-collision procedure according to the ISO/IEC 14443-3A protocol. By specifying the appropriate bit in SAK (SEL_RES) the PCD is informed that the PICC is capable handling the ISO/IEC 14443-4 protocol. This protocol is then entered by the command RATS. The PICC has to compose the ATS. This response has to be sent back within 4.8ms. The ATS is the first response, which has to be assembled (correct framing due to RATS parameters) by the PN51x’s host controller. The ATS can also specify new timeout values, which are valid for the subsequent responses. This timeout value is specified as FWT (Frame Waiting Time). (Please refer to chapter 5.2.5 in the ISO/IEC 14443-4 standard.) The only exception is that the DESELECT command has to be responded within 4.8ms even if the timeout value has specified a different value.

12. ID = Identifier, UID = Unique Identifier

Table 3: Timeout Values for ISO/IEC 14443-4 (T=CL) Communication

Command	Default FWT [ms]	Adjustable	Remarks
ATS/ RATS	4.8	No	
Data Exchange	4.8	Yes	Can be specified between 302µs and 4949ms
Deselect	4.8	No	
WTX	4.8	Yes	Requests for a higher timeout value (only for the current command)

3.6.2.1 Example (Acting as PICC emulation)

Assumptions:

- FWT = 4 (Defined by the PICC through ATS in TB(1) Bit8..Bit5).
- Interface type is SPI with 400kBit/s
- I – Block which has received a payload (INF bytes) of 10 bytes
- I – Block with 20 INF Bytes will be sent back (as response)
- # PN51x Read Operations: 20 (Configuration Offset) + 10 (INF bytes)
- # PN51x Write Operations: 20 (Configuration Offset) + 20 (INF bytes)
- Typical Dispatch Time (time between receiving, interpreting the command, and reacting according to the content): 1 ms

Calculations:

- Interface Speed Calculations: 400kBit/s = 50kByte/s = 20µs/Byte
- Typical Read = Typical Write = 2 typical R/W operations = 40µs/Read Register = 40µs/Write Register
- # Register Accesses: (20 + 10 + 20 + 20) * 40µs = 2.8ms
- Overall Time: #Register Accesses + Typical Dispatch Time = 3.8ms

The 3.8 ms is smaller than the defined timeout value of 4.8ms. Thus this Command/ Response pair will work.

3.6.3 NFCIP-1 Target (Active communication, all three speeds)

When the Initiator sends an initial ATR_REQ, the target has to react within a certain time frame. More precisely the Target has to switch on its RF field to notify the Initiator that a response to the request will follow. There is no timing constraint (according to ECMA340) between switching on the RF field and sending the response (ATR_RES). The ATR_RES can now send a new time-out value to the Initiator. This value is then valid for all the consecutive command interchanges. The time-out has the minimum value of 302us and the maximum value of 5s. The default value is 4.8ms. (Please refer to chapter 12.5.2.1 in the ECMA340 standard for more details).

3.6.4 NFCIP-1 Target (passive communication, all three speeds)

According to the ISO/IEC 18092 the ATR_RES has to be sent within the maximum time-out. This value is set to 4,949 seconds. This timeout can be lowered by the byte TO which is conveyed in the ATR_RES command itself. This value is then valid for all the commands, which follow (even for DSL, RLS).

3.6.5 FeliCa™ Card

The initialization phase (including sending the POLLING_RES) is automatically done by the PN51x. Further commands are Sony proprietary and have to be handled exclusively by the host controller.

4. System Synchronization

4.1 General

This is a NFC system consisting of:

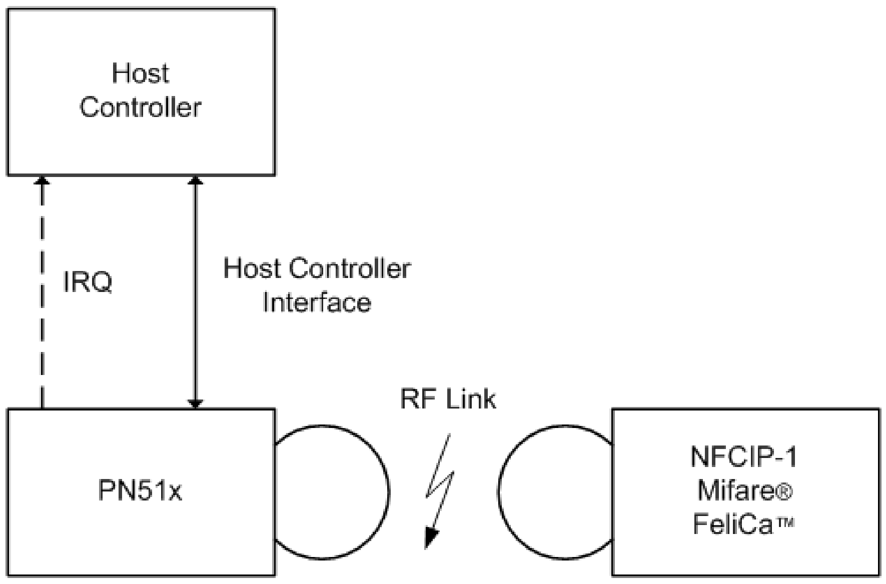

Figure 6 Some Sample NFC System Configuration

The host controller configures the PN51x via the chosen contact interface. The RF communication (contactless communication) is performed via RF Link. In this configuration the PN51x acts as reader (for Mifare® or FeliCa™) or as NFC target.

4.2 Event Notifications

There are several ways how to get notified whether an event has occurred. Following approaches are the most common ones:

4.2.1 Polling the Interrupt Registers (detection by software)

The most convenient (easy to implement) way to check events is to directly check the interrupt registers using read commands. All the relevant events are residing in the register `CommIrqReg` and in the register `DivIrqReg`.[13] This method can be used where energy and communication between host controller and PN51x is not important, because the host controller has to actively poll these registers all the time in order to not miss any relevant event.

4.2.2 Polling the IRQ bit (detection by software)

The IRQ bit in Status1Reg is set to one when at least one of the enabled interrupt bits [14] is set. This logic is different compared to the IRQ pin because only one of the enabled IRQ bits has to be set to one. Further interrupts will leave this value untouched (IRQ pin toggles for every additional interrupt), meaning the value will still have the value equal 1 until the interrupts have been reset. This approach can be seen as an intermediate step between the approach mentioned before and the approach which is described below.

[13] This is the way how it is implemented in the ExampleProject.

[14] Enabled interrupt means that the corresponding interrupt enable bit (in register 2 and 3) are set to 1.

Note that the values `CommIenReg.IrqInv` don't affect the behavior of the IRQ bit in Status1Reg.

4.2.3 Sensing the IRQ pin (detection by hardware)

The PN51x provides the functionality to generate events which are triggered by the PN51x itself or by an external event (e.g. detection of an external RF field). To use this functionality certain registers of the PN51x have to be configured appropriately. Following steps have to be accomplished:

- If the interrupt pin of the PN51x shall actively drive the signal (no open drain), the MSB in the register `CommIenReg` has to be set. Note: By default this bit is set to zero.
- If the host controller is only sensitive to falling edges the MSB of the register `DivIenReg` has to be set.
- The registers `CommIrqReg` and `DivIrqReg` have to be cleared manually[15].

When an interrupt occurred PN51x checks whether the event shall be propagated to the IRQ pin by checking the enable registers. If the masks are set the IRQ pin toggles[16].

4.2.4 BFL Examples

The software package (BFL – the software provided inside the NFC PN51x SDK) shows two different approaches how to evaluate CommIrqReg and DivIrqReg

- Polling the Interrupt Registers (Default)
  Whenever there is a event which is expected to be triggered by the PN51x (waiting for incoming data, waiting for RF detection) both `DivIrqReg` and `CommIrqReg` are polled. When the relevant bit is set the loop is left and further processing is done. This is the most generic approach but it blocks the complete application and leads to big communication effort between the host controller and the PN51x.
- Sensing the IRQ Pin
  The BFL also provides an example how to use the IRQ pin to inform the host controller that a relevant event has occurred. The relevance of the event is specified by setting the appropriate values in `CommIenReg` and `DivIenReg`. There is no need any more to permanently poll the interrupt registers. When the interrupt line signals an event, the interrupt registers shall be read out in order to check which event generated the interrupt.
  This example can be found in the files `ExampleUtilsC.c` and `ExampleUtils.cpp`. These implementations only work in the Windows environment.[17]

4.3 RF Register's Default Configuration

The registers stated below are depending on the antenna setup – the diameter, the number of turns, metal surroundings etc. The settings of the examples below are working with the provided demo readers. They are the result of laboratory evaluations with respect to reading distance and ISO compliancy (signal shaping, field strength required…).

[15] Note that clearing the registers 4 and 5 are performed in a different way. Clearing these registers is done by writing the value 7Fh. For further information please consult the PN51x data sheet.

[16] This means when two interrupts have occurred, the state is the same as before the interrupts have occurred.

[17] Usually this is implemented by using the Win32 function WaitForMultipleObjects(). However tests have shown that this doesn't work stable enough (Windows sometimes rejects interrupts). Therefore the approach with the WaitCommEvent() has been used, which is more stable.

Table 4: Setup Dependent Registers

Register Name	Address	Description
TxControlReg	14h	This is the register setting for a complementary antenna configuration - in case the single-ended option on RF side is used it has to be modified
RxThresholdReg	18h	Selects the threshold on RX pin on which the decision “valid incoming data” is taken
DemodReg	19h	Selects the demodulator options (including I/Q option)
MifareReg	1Ch	Defines ISO/IEC 14443A/ Mifare®/ NFC specific settings in target or card operating mode
TxBitPhaseReg	25h	Selects the TX bit synchronization
RFCfgReg	26h	Configures the RX Sensitivity
GsONReg	27h	Configures the N-MOS conductance of the TX output driver stage
CWGsPReg	28h	Configures the P-MOS conductance during no modulation
ModGsPReg	29h	Defines the P-MOS conductance during modulation

4.4 Communication Mode Dependent Register Settings

Table 4 shows the configuration settings for all the NFC operating modes. The cells which have gray (yellow) background, are default values, i.e. don’t have to be configured after a power-up or a soft reset.

Notes:

- Some settings may differ for different antenna configuration and tuning
- The settings are only valid for a certain hardware version. This tables is true for PN51x versions a.0 (PN511v2) and 8.0. (PN512)
- Prefix & (“AND”) and prefix | (“OR”) are indicating the bit mask which has to be applied to the register given
- Some settings have to be performed in a certain order. Please refer to the JCF files (script files delivered in the software package), the examples in the next chapter or the BFL to determine the correct register-setting sequence

Table 5: Default Register Setup during Communiation

Register Name and Address	Register Reset Value	Target passive	ISO/IEC 14443A PCD	ISO/IEC 14443B PCD	FeliCa™ Reader	NFC Initiator	NFC Target	NFC Initiator	NFC Target	NFC Initiator	NFC Target
Bit Rate (kBit/s)		**Any**	**106**	**106**	**212**	**106**	**106**	**212**	**212**	**424**	**424**
Control 0x0C	0x00	0x00	0x10	0x10	0x10	0x10	0x00	0x10	0x00	0x10	0x00
TxMode 0x12	0x00	0x80	0x80	0x83	0x92	0x81	0x81	0x91	0x91	0xA1	0xA1
RxMode 0x13	0x00	0x80	0x80 &0xF7	0x83	0x92	0x81	0x81	0x91	0x91	0xA1	0xA1
TxControl 0x14	0x80	&0xFC	&0x3F		&0x3F						
TxAuto 0x15	0x00	&0xB0	&0x77 |0x40		&0x37	|0x88 |0x40	|0x88 |0x40	|0x88 &0xBF	|0x88 &0xBF	|0x88 &0xBF	|0x88 &0xBF
RxThreshold 0x18	0x84	0x55	0x55	0x75	0x55	0x55	0x55	0x55	0x55	0x55	0x55
Demod 0x19	0x8D	0x61	0x4D	0x61	0x41	0x61	0x61	0x61	0x61	0x61	0x61
Mifare 0x1C	0x62	0x62									
GsNLoadMod 0x23	0x88	0x6F	0x6F	0x6F	0x6F	0x6F	0x6F	0x6F	0x6F	0x6F	0x6F
ModWidth 0x24	0x26	0x26	0x26	0x26	0x26	0x26	0x26	0x26	0x26	0x26	0x26
TxBitPhase 0x25	0x8F	&0x80 |0x07	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F	&0x80 |0x0F
RFCfg 0x26	0x48	0x59	0x59	0x59	0x59	0x59	0x59	0x59	0x59	0x59	0x59
GsN 0x27	0x88	0xF4	0xF4	0xFF	0xFF	0xF4	0xF4	0xFF	0xFF	0xFF	0xFF
CWGsP 0x28	0x20	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F
ModGsP 0x29	0x20	0x11	0x11	0x17	0x14	0x11	0x11	0x14	0x14	0x14	0x14

This table is true for PN511 and PN512 products. For PN511 not all registers are available. In this case go on with the next available register.

5. Basic Function Library

5.1 Purpose

The Basic Function Library is a layered software library for the PN51x. It provides fundamental sequences, which simplify writing applications for the PN51x. The library is distributed in source so that the developer can decide which components shall be included. The BFL cannot be run out of the box, because some configuration is highly dependent on the platform (such as the type of the microcontroller, type of the interface,...). For further information please refer to the “BFL Reference Manual” which is delivered within the software package.

5.2 Architecture

The design of the BFL is component based. This architecture has the advantage to reject the components, which are not relevant for the desired application. (For example it may not be necessary to support the FeliCa protocol for a Mifare application.)

The core library is written in C. In addition a C++ wrapper is supplied to allow the integration of the BFL in C++ applications. The BFL is ANSI C and ANSI C++ compliant. The implementation of the BFL is platform and Operating System independent.

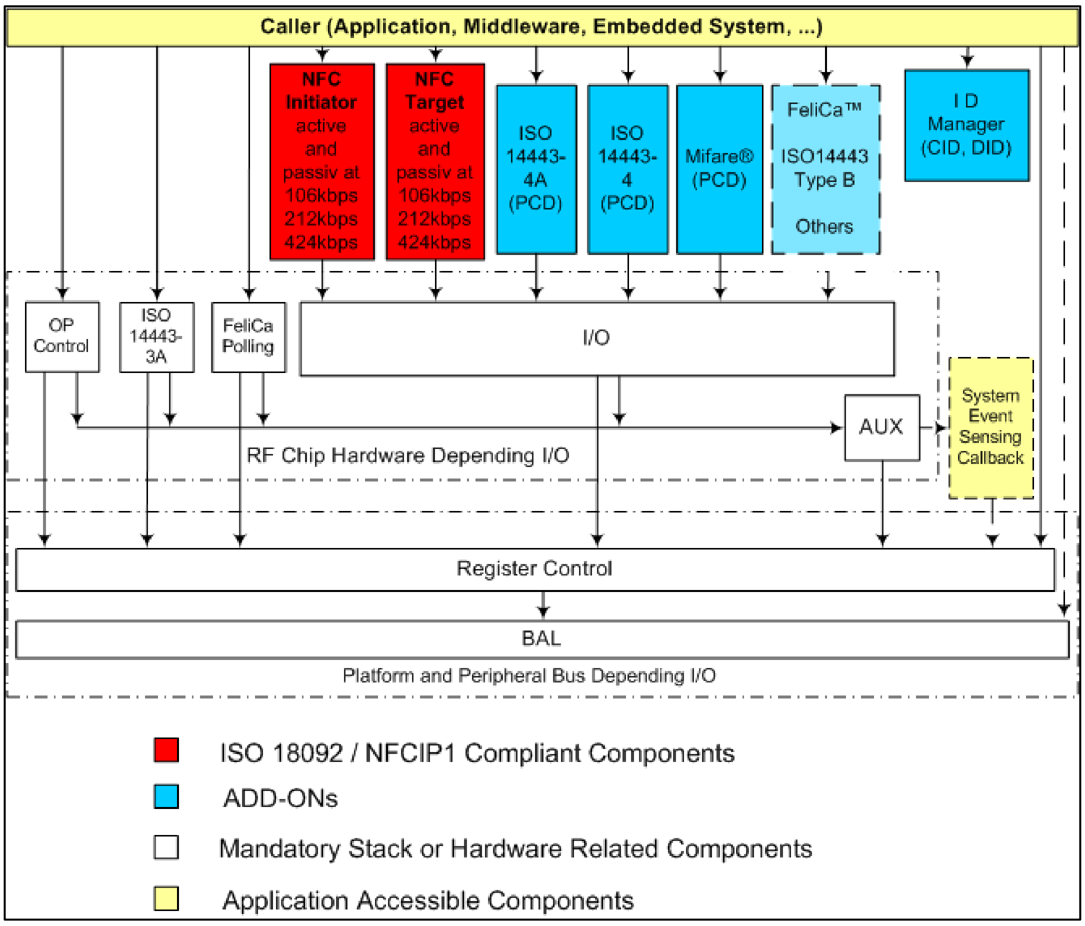
Caller (Application, Middleware, Embedded System, ...)
NFC Initiator active and passiv at 106kbps 212kbps 424kbps
NFC Target active and passiv at 106kbps 212kbps 424kbps
ISO 14443-4A (PCD)
ISO 14443-4 (PCD)
Mifare® (PCD)
FeliCa™
ISO14443 Type B
Others
I D Manager (CID, DID)
OP Control
ISO 14443-3A
FeliCa Polling
I/O
AUX
System Event Sensing Callback
RF Chip Hardware Depending I/O
Register Control
BAL
Platform and Peripheral Bus Depending I/O
🟥 ISO 18092 / NFCIP1 Compliant Components
🟦 ADD-ONs
☐ Mandatory Stack or Hardware Related Components
🟨 Application Accessible Components

Figure 7 Structure of the BFL (Basic Function Library)

The **Bus Abstraction Layer (BAL)** abstracts the physical host controller interface to the register control layer. This mainly covers the implementation of the interface-specific details (like I^2C, RS232...). This component provides two essential commands, which are named `phcsBflBal_Hw1SerWinWriteBus` and `phcsBflBal_Hw1SerWinReadBus`. These functions hide the type of the interface.

The **Register Control** (=RegCtl) provides easy access to the registers. This block consists of commands like `phcsBflRegCtl_SerHw1SetReg` (Set Register) and `phcsBflRegCtl_SerHw1GetReg` (Get Register) and convenience functions like `phcsBflRegCtl_SerHw1ModReg` (Modify Register). This is the first interface independent layer. Time critical issues are done in hardware, hence this block does not have any timing constraints.

The **I/O** block provides higher-level commands like `phcsBflIo_Hw1Transceive`, `phcsBflIo_Hw1Transmit` or `phcsBflIo_Hw1Receive`. Since this block is based on the RegCtl component, it uses the functions mentioned above to transfer the payload from / to the PN51x. This component has real time constraints because data has to be provided in time.

The **Operation Control** provides the possibility to set certain Operating Modes. This is done by using the functions like `phcsBflOpCtl_Hw1SetAttribute` and `phcsBflOpCtl_Hw1GetAttribute`. A command could be to set the PN51x in passive mode. The routines specify static values, hence this component does not have any time critical operations.

The **AUX** block contains a single function called `phcsBflAux_Hw1Command` which sends and receives a command (seen from the RF side). This function is only used internally (from I/O). This component has real time constraints because data has to be provided in time.

The **ISO14443-3A** block provides commands as specified in the ISO14443-3A standard. Consequently this interface provides commands like `phcsBflI3P3A_Hw1RequestA` (for REQA) or `phcsBflI3P3A_Hw1AnticollSelect` (for AC and SEL). Time critical issues are done in hardware, hence this block does not have any timing constraints.

The **FeliCa™ Polling** block provides the `phcsBflPolAct_Hw1Polling` (POLLING_REQUEST) command in order to check for FeliCa cards. Please note that other FeliCa commands are not implemented. Time critical issues are done in hardware, hence this block does not have any timing constraints.

On top of the I/O, higher-level protocols are implemented. This includes the implementation of **Mifare®** commands like MIFARE_READ, AUTHENTICATE which are implemented in the Mifare block. No timing constraints are given.

The **NFC** block provides the commands, which are specified in the ECMA 340 (ISO/IEC 18092) standard for initiator and target side.

The **ID Manager** manages the DIDs and CIDs, which are used in the NFCIP-1 protocol and in the ISO/IEC 14443-4 protocol. The ID Manager does not interact with any PN51x hardware, hence no timing constraints are relevant.

The **Event Sensing Callback** can be used to implement callback instead of busy wait mechanisms. For windows platform the hooks are prepared but not used yet. For other implementations the callback mechanisms are highly platform dependent and have to be tailored to ones needs.

5.3 Additional Notes

- Every component is placed in a dedicated directory, which holds both the C and C++ specific files.

5.4 Example: ExampleProject

The ExampleProject application shows how to use the BFL on Linux and on Windows. Please refer to the PN51x BFL Reference Manual section 2.24 for further instructions. After installing the provided software package the documentation can be found under Start → Programs → Philips Semiconductors → NFC PN51x Tools.

6. ANNEX A – Anti-Collision Mechanism

This chapter describes the anti-collision process for 106kBit/s in passive mode. This is defined in the ISO/IEC 14443-3A and used in ECMA340 and ISO/IEC 18092. For the description the ISO/IEC 14443 terminology is used.

6.1 General

The anti-collision mechanism is needed because the protocol is designed to support multiple cards within the RF field Hence there is a mechanism needed to detect and select every single card.

In the first phase the PCD sends an REQA command and checks if there is any response. If yes, the PCD proceeds with the anti-collision mechanism to receive a complete UID. The anti-collision mechanism is based on a bit – synchronous and on a Manchester-coding checking mechanism. The selection mechanism for a certain PICC can be seen in the following figure:

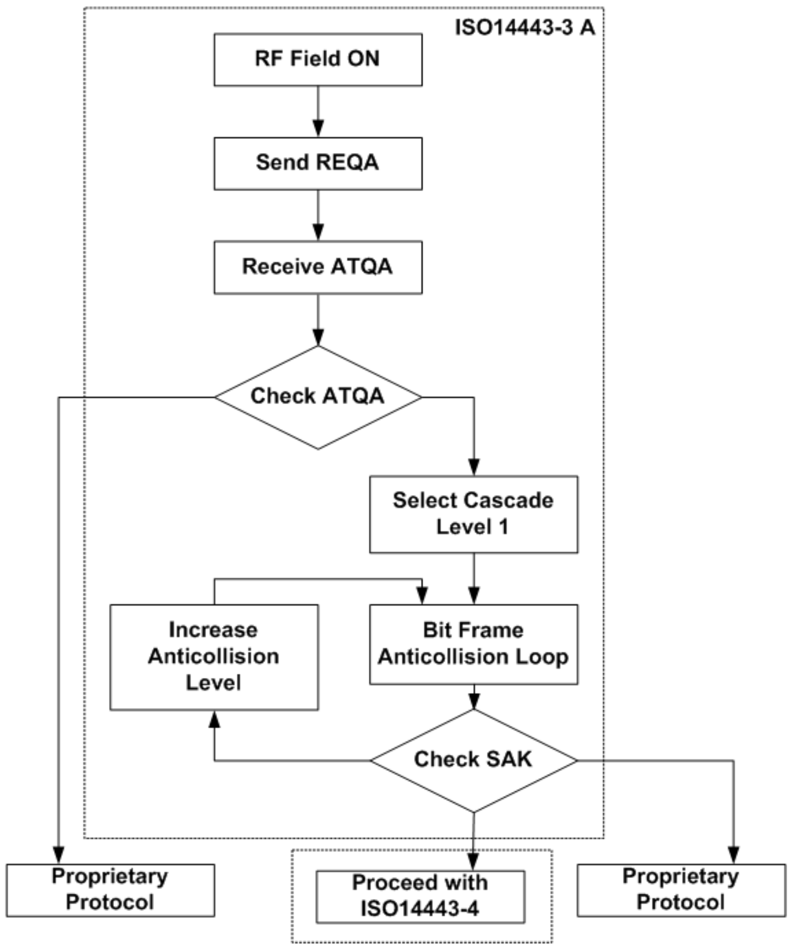

Figure 8 ISO/IEC 14443-3A Anti-Collision Loop

6.2 Anti-collision in Detail

Figure 9 shows the anti-collision procedure in a more detailed way. The flow is seen from the reader's point of view. The text below describes the flow of the protocol.

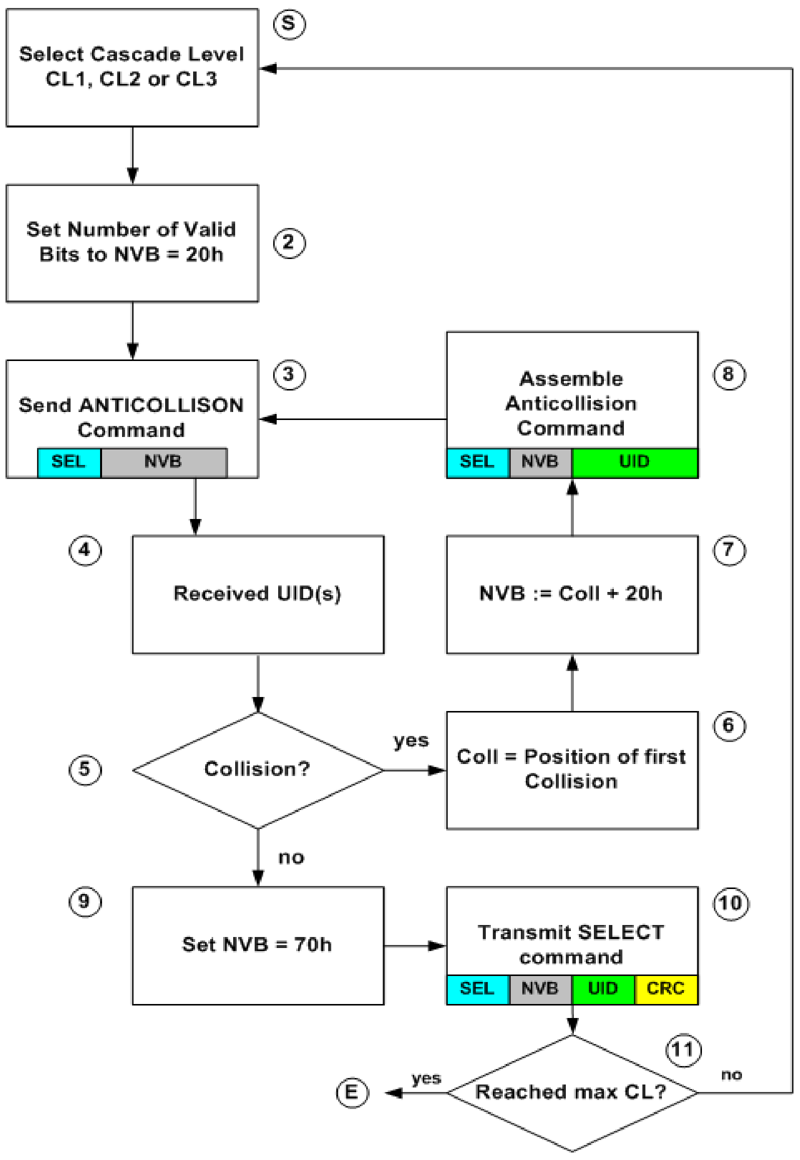
S
Select Cascade Level CL1, CL2 or CL3
Set Number of Valid Bits to NVB = 20h
2
Send ANTICOLLISON Command
SEL
NVB
3
Assemble Anticollision Command
SEL
NVB
UID
8
4
Received UID(s)
NVB := Coll + 20h
7
5
Collision?
yes
Coll = Position of first Collision
6
no
9
Set NVB = 70h
Transmit SELECT command
SEL
NVB
UID
CRC
10
11
E
yes
Reached max CL?
no

Figure 9 More Detailed Anti-Collision Loop

S = Start

E = End

In the first step (S) the cascade level is defined by the reader/writer[18]. Four byte UIDs can be determined by Cascade Level 1. In order to get 7 byte UIDs from cards, Cascade Level 1 and Cascade Level 2 have to be performed to get the complete Unique Identifier. Similarly 10 byte UIDs can be determined by using all three Cascade Levels.

In the second step (2) the value of NVB (Number of Valid Bits) is set to 20h. The higher nibble of this byte defines the number of valid bytes and the lower nibble defines the

18. The anti-collision process always starts with the first level (CL1)

number of valid bits. Since the first anti-collision command is two bytes long, the NVB is set to 20h.

In step 3 the command is assembled (e.g. in the first step: 93h 20h) and then sent via the RF field.

In step 4 the ISO/IEC 14443-3A - compatible devices will respond with their Manchester coded UIDs in a bit-synchronous manner. If no collision has been detected, the complete UID is known by the response of the PICC. Therefore a selection of the PICC can be performed. This is done in step 9 and step 10. Now the SAK byte (which is sent back from the PICC after the selection) gives the PCD the information if there is an additional cascade level. This decision is done in step 11. If there is an additional cascade level, proceed to step S, otherwise the selection process is finished which is denoted by the step E.

Collisions in step 5 are recognized by checking the code validity of the Manchester coding (this is done by hardware). If at least two cards are responding and at least one bit is different (which is always the case in Mifare, since unique IDs are guaranteed), the hardware stores the position, determines the valid bits, and sets NVB to this value. The part of the UID, which has been transmitted before the collision is the valid section. This is determined in step 6. The bit position, where the collision itself has been occurred is specified by the reader and also taken as a valid bit and set by the reader. The new value NVB is defined in step 7.

In step 8 the new ANTICOLLISION command is assembled by concatenating the SEL byte, the new NVB and the known part of the UID including one single bit which is defined by the PCD. By defining this bit, one of the targets will be excluded.

Now step 3 is performed once again. This cycle is performed until the whole UID for the current cascade level is complete. In worst case, if a collision takes place at every bit position, the loop has to be passed 32 times per cascade level (at least 32 cards in the field).

If the anti-collision process has terminated, the PICC is active and the PCD can determine if the PICC is the appropriate one. If not, the PCD can set this PICC into the HALT mode and redo the anti-collision process with the next PICC.

7. Appendix B – Naming in ISO/IEC 14443 and ECMA340

In order to distinguish between the different standards (ISO/IEC 14443 and ECMA340), the NFCIP-1 standard uses a different terminology, however the structures behind them are very similar. The following table gives an overview of the command description in the ECMA 340 and the corresponding expression in the ISO 14443-3A. Table 6 also describes some relationships between smart card related terms.

Table 6: Command comparison between ISO 14443-3A and ECMA 340

ISO 14443-3		ECMA 340	
Command / Response	**Description**	**Command / Response**	**Description**
REQA	Request Command (sent by PCD)	SENS_REQ	Sense Request (sent by Initiator)
WUPA	Wake-Up Command (sent by PCD)	ALL_REQ	All Request (sent by Initiator)
ATQA	Response command to REQA and WUPA (sent by PICC)	SENS_RES	Sense Response Response to SENS_REQ and ALL_REQ (sent by Target)
ANTICOLLISION	Anti-collision command (sent by PCD)	SDD_REQ	Single device detection request (sent by Initiator)
SELECT	Select command (sent by PCD)	SEL_REQ	Select Request (sent by Initiator)
SAK	Select Acknowledge Response command to SELECT Command (sent by PICC)	SEL_RES	Select Response (sent by Target)
HLTA	Halt command (sent by PCD)	SLP_REQ	Sleep Request (sent by Initiator)
	No response to the HLTA command.		No response to the SLP_REQ command.
DESELECT	Deselecting Request in the ISO/IEC 14443-4 protocol.	DSL_REQ	Deselect Request (sent by Initiator)
		DSL_RES	Deselect Response (sent by Target)

Table 7: PICC and Target States

ISO 14443-3	ECMA 340	Description
ACTIVE State	SELECTED State	PICC / Target has been selected, listens to higher layer messages
ACTIVE* State	SELECTED* State	PICC / Target has been selected, listens to higher layer messages
HALT State	SLEEP State	Listens for commands and reacts for certain commands
IDLE State	SENSE State	PICC / Target waits for a request from the PCD / Initiator
POWER-OFF State	POWER-OFF State	No carrier

ISO 14443-3	ECMA 340	Description
READY State	RESOLUTION State	State for Anticollision / Single Device Detection
READY* State	RESOLUTION* State	State for Anticollision / Single Device Detection

8. Glossary

Table 8: Command comparison between ISO 14443-3A and ECMA 340

ISO 14443-3		ECMA 340	
Abbreviation	Expression	Abbreviation	Expression
AC	Anticollision	SDD	single device detection (by using RF collision avoidance)
BCC	UID CLn check byte, calculated as exclusive-or over the 4 previous bytes, Type A	BCC	NFCID1 CLn checkbyte, same calculation as in ISO/IEC 14443
CT	Cascade Tag, Type A	CT	Cascade Tag
CRC_A	Cyclic Redundancy Check error detection code Type A	CRC	
E	End of communication, Type A	End	End of communication
FDT	Frame Delay Time, Type A	FRT	Frame Response Time
Frame delay time PICC to PCD		Frame Response Time Target to Initiator	
NVB	Number of Valid Bits, Type A	SEL_PAR	Select Parameter Byte
P	Odd Parity bit, Type A	P	Odd Parity bit
PCD	Proximity Coupling Device	I	Initiator
PICC	Proximity Card	T	Target
Request Guard Time	Minimum time between the start of two consecutive REQA commands	Sense Guard Time	Minimum time between the start of two consecutive SENS_REQ commands
S	Start of communication, Type A	Start	Start of communication
SEL	SELect code, Type A (defines the Cascade Level)	SEL_CMD	Select Command Byte (defines the Cascade Level)
UID	Unique Identifier, Type A	NFCID1	Random Identifier
uid	Byte number n of Unique IDentifier, n ≥ 0	ncid1n	Byte number n of NFCID1

Table 9: General Abbreviations

Abbreviation	Description
DIF	Dual InterFace Card
f_c	Carrier Frequency (13,56MHz)
CLn	Cascade Level n (0<n<4)
RFCA	RF-Collision Avoidance, prevents that two devices switch on their RF-field at the same time
BFL	Basic Function Library – the software which is delivered by Philips to control all registers necessary on PN51x ICs.

9. Bibliography

- General Information
  http://www.semiconductors.philips.com/markets/identification/products/nfc/index.html
- ECMA Near Field Communication (NFC) White Paper (PDF)
  http://www.ecma-international.org/activities/Communications/2004tg19-001.pdf
- ECMA-340 Near Field Communication – Interface and Protocol (NFCIP1)
  This 2nd edition fully matches ISO/IEC 18092.
  http://www.ecma-international.org/publications/standards/Ecma-340.htm
- ISO/IEC 14443 Specification (fee based)
  http://www.iso.org/iso/en/CombinedQueryResult.CombinedQueryResult?queryString=14443
- ISO/IEC 18092 Specification (fee-based)
  http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38578&COMMID=&scopelist=
- Link to freely available ISO standards:
  http://isotc.iso.org/livelink/livelink/fetch/2000/2489/Ittf_Home/PubliclyAvailableStandards.htm
  Find <18092> in the page and download it by performing a right-mouse-click on the link choosing Save Target As.
- ISO/ IEC 22536 (ECMA 356) NFCIP2
  These standards can be found the given (free) sites, too.
- PN511 – Transmission module, Short Form Specification (PDF)
  http://www.semiconductors.philips.com/acrobat_download/other/identification/sfs_pn511_rev2.0.pdf
- PN51x Data Sheet
  This is only delivered by Philips Controlled Document System – please contact your local Philips Support Desk

10. Table of Figures

11. Disclaimers

12. Licenses

13. Patents

14. Trademarks

15. Contents

Date of release:11. May 2006
Document number: 108711

Published in The Netherlands