AN12650

Using the PN5180 without library

Rev. 1.0 — 7 January 2020
581710

Application note
COMPANY PUBLIC

Document information

Information	Content
Keywords	PN5180
Abstract	This document describes how to communicate with the PN5180 without a library

Revision history

Rev	Date	Description
1.0	20200107	Initial version

1 Introduction

This document describes how a host processor can interact with the PN5180 via an SPI interface.

The PN5180 understands 24 different host-interface-commands, which are used to alter register values or start specific routines, like sending data via the NFC interface.

8 out of the 24 host-interface-commands are used in this document, which are described in section 3. All other commands can be looked up in the PN5180 data sheet. (Ref. 1)

2 PN5180 structure

The PN5180 is a High-Power NFC frontend. It implements the RF functionality like an antenna driving and receiver circuitry and all the low-level functionality to realize an NFC Forum-compliant reader.

The PN5180 connects to a host microcontroller with a SPI interface for configuration, NFC data exchange and high-level NFC protocol implementation.

Internal registers of the PN5180 state machine store configuration data.

Two types of memory are implemented in the PN5180: RAM and EEPROM.

The RF configuration for dedicated RF protocols is defined by EEPROM data which is copied by a command issued from the host microcontroller - LOAD_RF_CONFIG into the registers of the PN5180.

2.1 SPI Host interface

The interface of the PN5180 to a host microcontroller is based on a SPI interface, extended by signal line BUSY.

The BUSY signal is used to indicate that the PN5180 is not able to send or receive data over the SPI interface.

- *Master in Slave out - (MISO)*
  The MISO line is configured as an output in a slave device. It is used to transfer data from the slave to the master, with the most significant bit sent first.
- *Master out Slave in - (MOSI)*
  The MOSI line is configured as an input in a slave device. It is used to transfer data from the master to a slave, with the most significant bit sent first.
- *Serial clock - (SCK)*
  The serial clock is used to synchronize data movement both in and out of the device through its MOSI and MISO lines.
- *Not slave select - (NSS)*
  The slave select input (NSS) line is used to select a slave device. It shall be set to low before any data transaction starts and must stay low during the transaction.
- *BUSY*
  During frame reception, the BUSY line goes ACTIVE and goes to IDLE when PN5180 is able to receive a new frame or data is available.

There is no chaining allowed, meaning that the whole instruction has to be sent or the whole receive buffer has to be read out. The whole transmit buffer shall be written at once as well.

2.2 Host interface command

A Host Interface Command consists of either 1 or 2 SPI frames depending whether the host wants to write or read data from the PN5180.

An SPI Frame consists of multiple bytes. The protocol used between the host and the PN5180 uses 1 byte indicating the instruction code and additional bytes for the payload (instruction-specific data). The actual payload size depends on the instruction used. The minimum length of the payload is 1 byte.

All commands are packed into one SPI Frame. An SPI Frame consists of multiple bytes. No NSS toggles allowed during sending of an SPI frame.

2.3 Register structure

The PN5180 contains 44 register, which control the behavior of the PN5180 processor.

The size of a register is 4 byte.

The value of a register can be altered by the host processor through 4 different commands: write_register , write_register_and_mask, write_register_or_mask, write_register_multiple

Detailed descriptions of every register and it's purpose can be looked up in the PN5180 data sheet (Ref.1).

3 PN5180 host interface commands

This section describes 8 out of 24 host-interface-commands used in the example in section 4.

All other commands can be looked up in the PN5180 Data sheet. (Ref. 1)

3.1 WRITE_REGISTER

This command writes a 32-bit value into a configuration register.

Table 1. WRITE_REGISTER

Payload	Length	Value/Description
Command Code	1	0x00
Parameter	1	Register address
Parameter	4	Register content

3.2 WRITE_REGISTER_OR_MASK

This command modifies the content of a register using a logical OR operation. The content of the register is read and a logical OR operation is performed with the provided mask. The modified content is written back to the register.

Table 2. WRITE_REGISTER_OR_MASK

Payload	Length	Value/Description
Command Code	1	0x01
Parameter	1	Register address
Parameter	4	logical OR mask

3.3 WRITE_REGISTER_AND_MASK

This command modifies the content of a register using a logical AND operation. The content of the register is read and a logical AND operation is performed with the provided mask. The modified content is written back to the register.

Table 3. WRITE_REGISTER_AND_MASK

Payload	Length	Value/Description
Command Code	1	0x02
Parameter	1	Register address
Parameter	4	logical AND mask

3.4 SEND_DATA

This command writes data to the RF transmission buffer and starts the RF transmission.

Number of valid bits in last byte:

0 … All bits of last byte are transmitted

1-7 … Number of bits within last byte to be transmitted

Table 4. SEND_DATA

Payload	Length	Value/Description
Command Code	1	0x09
Parameter	1	Number of valid bits in last byte
NFC Payload	1....260	Array of up to 260 elements

3.5 READ_DATA

This command reads data from the RF reception buffer, after a successful reception.

Table 5. READ_DATA

Payload	Length	Value/Description
Command Code	1	0x0A
Parameter	1	0x00
Response	1....508	Array of up to 508 elements

3.6 LOAD_RF_CONFIG

This command is used to load the RF configuration from EEPROM into the configuration registers.

Table 6. LOAD_RF_CONFIG

Payload	Length	Value/Description
Command Code	1	0x11
Parameter	1	Transmitter configuration byte
Parameter	1	Receiver configuration byte

3.7 RF_ON

This command switches the internal RF field ON.

Table 7. RF_ON

Payload	Length	Value/Description
Command Code	1	0x16
Parameter	1	Bit0 == 1: disable collision avoidance according to ISO/IEC 18092

3.8 RF_OFF

This command switches the internal RF field OFF.

Table 8. RF_ON

Payload	Length	Value/Description
Command Code	1	0x17
Parameter	1	Dummy byte

4 Interaction

4.1 ISO/IEC 14443

4.1.1 Example Code - ISO/IEC 14443 - REQA

This code snippet initializes the PN5180 to the ISO14443 protocol, sends a REQA command and listens to the response of a tag.

```
1: sendSPI(0x11, 0x00, 0x80);
2: sendSPI(0x16, 0x00);
3: sendSPI(0x02, 0x19, 0xFE, 0xFF, 0xFF, 0xFF);
4: sendSPI(0x02, 0x12, 0xFE, 0xFF, 0xFF, 0xFF);
5: sendSPI(0x00, 0x03, 0xFF, 0xFF, 0x0F, 0x00);
6: sendSPI(0x02, 0x00, 0xF8, 0xFF, 0xFF, 0xFF);
7: sendSPI(0x01, 0x00, 0x03, 0x00, 0x00, 0x00);
8: sendSPI(0x09, 0x07, 0x26);
9: waitForCardResponse();
10: sendSPI(0x0A, 0x00);
11: sendSPI(0x17, 0x00);
```

Every sendSPI function sends an SPI frame with the given values in the arguments.

4.1.2 Line by line description - ISO/IEC 14443 - REQA

1: Loads the ISO 14443 - 106 protocol into the RF registers

2: Switches the RF field ON.

3: Switches the CRC extension *off* in Tx direction

4: Switches the CRC extension *off* in Rx direction

5: Clears the interrupt register IRQ_STATUS

6: Sets the PN5180 into IDLE state

7: Activates TRANSCEIVE routine

8: Sends REQA command

9: Waits until a Card has responded via checking the IRQ_STATUS register

10: Reads the reception buffer. (ATQA)

11: Switches the RF field OFF.

4.1.3 Detailed frame description - ISO/IEC 14443 - REQA

1: sendSPI(0x11, 0x00, 0x80);

Table 9. Line 1

Value	Payload	Description
0x11	Command Code	Executes the command LOAD_RF_CONFIG
0x00	Parameter 1	Transmitter configuration byte 0x00 translates to load protocol ISO 14443-A - 106
0x80	Parameter 2	Receiver configuration byte 0x80 translates to load protocol ISO 14443-A - 106

2: sendSPI(0x16, 0x00);

Table 10. Line 2

Value	Payload	Description
0x16	Command Code	Executes the command RF_ON
0x00	Parameter	Disable collision avoidance according to ISO 18092

3: sendSPI(0x02, 0x19, 0xFE, 0xFF, 0xFF, 0xFF);

Table 11. Line 3

Value	Payload	Description
0x02	Command	Executes the command WRITE_REGISTER_AND_MASK
0x19	Register Address	Lay the logic AND mask onto the register CRC_TX_CONFIG
0xFE	Mask byte 1	Switches the CRC in Tx direction OFF
0xFF	Mask byte 2	all other register settings remain the same
0xFF	Mask byte 3	all other register settings remain the same
0XFF	Mask byte 4	all other register settings remain the same

4: sendSPI(0x02, 0x12, 0xFE, 0xFF, 0xFF, 0xFF);

Table 12. Line 4

Value	Payload	Description
0x02	Command	Executes the command WRITE_REGISTER_AND_MASK
0x12	Register Address	Lay the logic AND mask onto the register CRC_RX_CONFIG
0xFE	Mask byte 1	Switches the CRC in Rx direction OFF
0xFF	Mask byte 2	all other register settings remain the same
0xFF	Mask byte 3	all other register settings remain the same
0xFF	Mask byte 4	all other register settings remain the same

5: sendSPI(0x00, 0x03, 0xFF, 0xFF, 0x0F, 0x00);

Table 13. Line 5

Value	Payload	Description
0x00	Command	Executes the command WRITE_REGISTER
0x03	Register Address	write into register IRQ_STATUS
0xFF	Byte 1	Clear values in the register
0xFF	Byte 2	Clear values in the register
0x0F	Byte 3	Clear values in the register
0x00	Byte 4	Bits [20:31] are RFU therefore they don't need to be cleared

6: sendSPI(0x02, 0x00, 0xF8, 0xFF, 0xFF, 0xFF);

Table 14. Line 6

Value	Payload	Description
0x02	Command	Executes the command WRITE_REGISTER_AND_MASK
0x00	Register Address	Lay the logic AND mask onto the register SYSTEM_CONFIG
0xF8	Mask byte 1	Set the internal state machine to IDLE
0xFF	Mask byte 2	all other register settings remain the same
0xFF	Mask byte 3	all other register settings remain the same
0xFF	Mask byte 4	all other register settings remain the same

7: sendSPI(0x01, 0x00, 0x03, 0x00, 0x00, 0x00);

Table 15. Line 7

Value	Payload	Description
0x01	Command	Executes the command WRITE_REGISTER_OR_MASK
0x00	Register Address	Lay the logic OR mask onto the register SYSTEM_CONFIG
0x03	Mask byte 1	Initiates TRANSCEIVE state
0x00	Mask byte 2	all other register settings remain the same
0x00	Mask byte 3	all other register settings remain the same
0x00	Mask byte 4	all other register settings remain the same

8: sendSPI(0x09, 0x07, 0x26);

Table 16. Line 8

Value	Payload	Description
0x09	Command	Executes the command SEND_DATA
0x07	Parameter	Only the last 7 bit of the last bytes shall be sent (short frame for REQA)
0x26	NFC payload	ISO14443 REQA command

10: sendSPI(0x0A, 0x00);

Table 17. Line 10

Value	Payload	Description
0x0A	Command	Execute the command READ_DATA
0x00	Parameter	This parameter has to be always 0x00

11: sendSPI(0x17, 0x00);

Table 18. Line 11

Value	Payload	Description
0x17	Command	Executes the command RF_OFF
0x00	Dummy byte	This byte may have any value

4.2 ISO/IEC 15693

4.2.1 Example Code - ISO/IEC 15693 - Inventory

This code snippet initializes the PN5180 to the ISO15693 protocol, sends an Inventory command and listens to the response of a tag.

```
1: sendSPI(0x11, 0x0D, 0x8D);
2: sendSPI(0x16, 0x00);
3: sendSPI(0x00, 0x03, 0xFF, 0xFF, 0x0F, 0x00);
4: sendSPI(0x02, 0x00, 0xF8, 0xFF, 0xFF, 0xFF);
5: sendSPI(0x01, 0x00, 0x03, 0x00, 0x00, 0x00);
6: sendSPI(0x09, 0x00, 0x06, 0x01, 0x00);
7: for(SlotCounter = 0; SlotCounter < 16 ; SlotCounter++)
{
8: if(CardHasResponded)
{
9: sendSPI(0x0A, 0x00);
10: readSPI(UIDbuffer);
}
11: sendSPI(0x02, 0x18, 0x3F, 0xFB, 0xFF, 0xFF);
12: sendSPI(0x02, 0x00, 0xF8, 0xFF, 0xFF, 0xFF);
13: sendSPI(0x01, 0x00, 0x03, 0x00, 0x00, 0x00);
14: sendSPI(0x00, 0x03, 0xFF, 0xFF, 0x0F, 0x00);
15: sendSPI(0x09, 0x00);
}
16: sendSPI(0x17, 0x00);
```

Every sendSPI function sends an SPI frame with the given values in the arguments.

ReadSPI function reads the RF reception buffer and saves it into the argument.

4.2.2 Line by line description - ISO/IEC 15693 - Inventory

1: Loads the ISO 15693 protocol into the RF registers

2: Switches the RF field ON.

3: Clears the interrupt register IRQ_STATUS

4: Sets the PN5180 into IDLE state

5: Activates TRANSCEIVE routine

6: Sends an inventory command with 16 slots

7: A loop that repeats 16 times since an inventory command consists of 16 time slots

8: The function CardHasResponded reads the RX_STATUS register, which indicates if a card has responded or not.

9: Reads the reception Buffer

10: Everything in the reception buffer shall be saved into the UIDbuffer array.

11: Send only EOF (End of Frame) without data at the next RF communication.

12: Sets the PN5180 into IDLE state

13: Activates TRANSCEIVE routine

14: Clears the interrupt register IRQ_STATUS

15: Send EOF

16: Switch OFF RF field

4.2.3 Detailed frame description - ISO/IEC 15693 - Inventory

1:sendSPI(0x11, 0x0D, 0x8D);

Table 19. Line 1

Value	Payload	Description
0x11	Command Code	Executes the command LOAD_RF_CONFIG
0x0D	Parameter 1	Transmitter configuration byte 0x0D translates to load protocol ISO 15693
0x8D	Parameter 2	Receiver configuration byte 0x8D translates to load protocol ISO 15693

2: sendSPI(0x16, 0x00);

Table 20. Line 2

Value	Payload	Description
0x16	Command Code	Executes the command RF_ON
0x00	Parameter	Disable collision avoidance according to ISO 18092

3: sendSPI(0x00, 0x03, 0xFF, 0xFF, 0x0F, 0x00);

Table 21. Line 3

Value	Payload	Description
0x00	Command	Executes the command WRITE_REGISTER
0x03	Register Address	write into register IRQ_STATUS
0xFF	Mask byte 1	Clear values in the register
0xFF	Mask byte 2	Clear values in the register
0x0F	Mask byte 3	Clear values in the register
0X00	Mask byte 4	Bits [20:31] are RFU therefore they don't need to be cleared

4: sendSPI(0x02, 0x00, 0xF8, 0xFF, 0xFF, 0xFF);

Table 22. Line 4

Value	Payload	Description
0x02	Command	Executes the command WRITE_REGISTER_AND_MASK
0x00	Register Address	Lay the logic AND mask onto the register SYSTEM_CONFIG

Value	Payload	Description
0xF8	Mask byte 1	Set the internal state machine to IDLE
0xFF	Mask byte 2	all other register settings remain the same
0xFF	Mask byte 3	all other register settings remain the same
0xFF	Mask byte 4	all other register settings remain the same

5: sendSPI(0x01, 0x00, 0x03, 0x00, 0x00, 0x00);

Table 23. Line 5

Value	Payload	Description
0x01	Command	Executes the command WRITE_REGISTER_OR_MASK
0x00	Register Address	Lay the logic OR mask onto the register SYSTEM_CONFIG
0x03	Mask Byte 1	Initiates TRANSCEIVE state
0x00	Mask Byte 2	all other register settings remain the same
0x00	Mask Byte 3	all other register settings remain the same
0x00	Mask Byte 4	all other register settings remain the same

6: sendSPI(0x09, 0x00, 0x06, 0x01, 0x00);

Table 24. Line 6

Value	Payload	Description
0x09	Command	Executes the command SEND_DATA
0x00	Parameter	Everything shall be sent
0x06	NFC payload	Flag Byte defined by ISO15693 - Inventory with 16 Slots
0x01	NFC payload	Inventory Command defined by ISO 15693
0x00	NFC payload	Mask Length is 0 defined by ISO 15693

7: for(SlotCounter = 0; SlotCounter < 16 ; SlotCounter++)

A loop which counts from 0 to 15, which is used to check all time slots in an Inventory request

8: if(CardHasResponded)

The function CardHasResponded reads the RX_STATUS register, which indicates if a card has responded or not.

Bits 0-8 of the RX_STATUS register indicate how many bytes where received. If this value is higher than 0, a Card has responded.

9: sendSPI(0x0A, 0x00);

Table 25. Line 9

Value	Payload	Description
0x0A	Command	Execute the command READ_DATA
0x00	Parameter	This parameter has to be always 0x00

10:readSPI(UIDbuffer);

readSPI reads the reception Buffer of the PN5180 and saves it into the UIDbuffer.

Since the last command was "READ_DATA" the PN5180 will respond the content of the reception Buffer and communicate the values via the MISO line.

11: sendSPI(0x02, 0x18, 0x3F, 0xFB, 0xFF, 0xFF);

Table 26. Line 11

Value	Payload	Description
0x02	Command	Executes the command WRITE_REGISTER_AND_MASK
0x18	Register Address	Lay the logic AND mask onto the register TX_CONFIG
0x3F	Mask Byte 1	Sets bit 7 and 8 to zero - By setting bits 7, 8 and 11 to zero only a EOF is going to be sent at the next SEND_DATA command
0xFB	Mask Byte 2	Sets bit 11 to zero - By setting bits 7, 8 and 11 to zero only a EOF is going to be send at the next SEND_DATA command
0xFF	Mask Byte 3	all other register settings remain the same
0xFF	Mask Byte 4	all other register settings remain the same

12: sendSPI(0x02, 0x18, 0x3F, 0xFB, 0xFF, 0xFF);

Table 27. Line 12

Value	Payload	Description
0x02	Command	Executes the command WRITE_REGISTER_AND_MASK
0x00	Register Address	Lay the logic AND mask onto the register SYSTEM_CONFIG
0xF8	Mask Byte 1	Set the internal state machine to IDLE
0xFF	Mask Byte 2	all other register settings remain the same
0xFF	Mask Byte 3	all other register settings remain the same
0xFF	Mask Byte 4	all other register settings remain the same

13: sendSPI(0x01, 0x00, 0x03, 0x00, 0x00, 0x00);

Table 28. Line 13

Value	Payload	Description
0x01	Command	Executes the command WRITE_REGISTER_OR_MASK
0x00	Register Address	Lay the logic OR mask onto the register SYSTEM_CONFIG
0x03	Mask Byte 1	Initiates TRANSCEIVE state
0x00	Mask Byte 2	all other register settings remain the same
0x00	Mask Byte 3	all other register settings remain the same
0x00	Mask Byte 4	all other register settings remain the same

14: sendSPI(0x00, 0x03, 0xFF, 0xFF, 0x0F, 0x00);

Table 29. Line 14

Value	Payload	Description
0x00	Command	Executes the command WRITE_REGISTER
0x03	Register Address	write into register IRQ_STATUS
0xFF	Mask byte 1	Clear values in the register
0xFF	Mask byte 2	Clear values in the register
0x0F	Mask byte 3	Clear values in the register
0X00	Mask byte 4	Bits [20:31] are RFU therefore they don't need to be cleared

15: sendSPI(0x09, 0x00);

Table 30. Line 15

Value	Payload	Description
0x09	Command	Executes the command SEND_DATA
0x00	Parameter	Whole Bytes shall be sent - Now only sends EOF, because of the adjustments in Register TX_CONFIG

16: sendSPI(0x17, 0x00);

Table 31. Line 16

Value	Payload	Description
0x17	Command	Executes the command RF_ON
0x00	Dummy byte	This byte may have any value

5 References

[1]

High-performance multiprotocol full NFC frontend, supporting all NFC Forum modes

6 Legal information

6.1 Definitions

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

6.2 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.

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 — While NXP Semiconductors has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP Semiconductors accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

6.3 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.

6.4 Trademarks

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

Tables

Contents

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

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

Date of release: 7 January 2020
Document identifier: AN12650
Document number: 581710