AN15072

Implementation of an I3C Secondary Bootloader on MCX N947

Rev. 1.0 — 20 July 2026 Application note

Document information

Information	Content
Keywords	AN15072, bootloader, I3C, MCX N947
Abstract	This application note describes implementing and testing an I3C secondary bootloader on FRDM-MCXN947 using an MCX N236 USB to I3C bridge as controller for transactions.

1 Introduction

The I3C interface is widely used in many scenarios, such as data center, PCs, and optical modules. As a result, I3C-based secondary bootloader for firmware updates have become a common requirement. This application note describes the implementation of an I3C secondary bootloader on the MCX N947.

1.1 I3C bootloader supported commands

Table 1 lists the commands supported by the I3C secondary bootloader on the MCXN947 device.

Table 1. Supported I3C bootloader commands for MCX N947

PC command	Bootloader function
daa	Runs I3C Dynamic Address Assignment (DAA) through the bridge
get-property	Reads bootloader properties
read-memory	Reads data from flash memory or RAM
write-memory	Writes data to flash memory or RAM
flash-erase-all	Erases the application flash region while preserving the protected bootloader region
flash-erase-region	Erases a sector-aligned flash region outside the protected bootloader area
reset	Resets the target device

2 MCX N947 I3C secondary bootloader demo introduction

The MCX N947 I3C secondary bootloader demo is implemented using the NXP *blhost* framework and supports I3C communication speeds of up to 12.5 Mbit/s.

In this setup, a PC communicates with the FRDM-MCXN236 board that runs the USB-to-I3C bridge firmware, available at: Github

The FRDM-MCXN236 board appears on the PC as a USB CDC virtual COM port. The bridge converts host commands to I3C controller transactions. The FRDM-MCXN947 runs the secondary bootloader as an I3C target and receives *blhost* serial framing packets over the I3C bus.

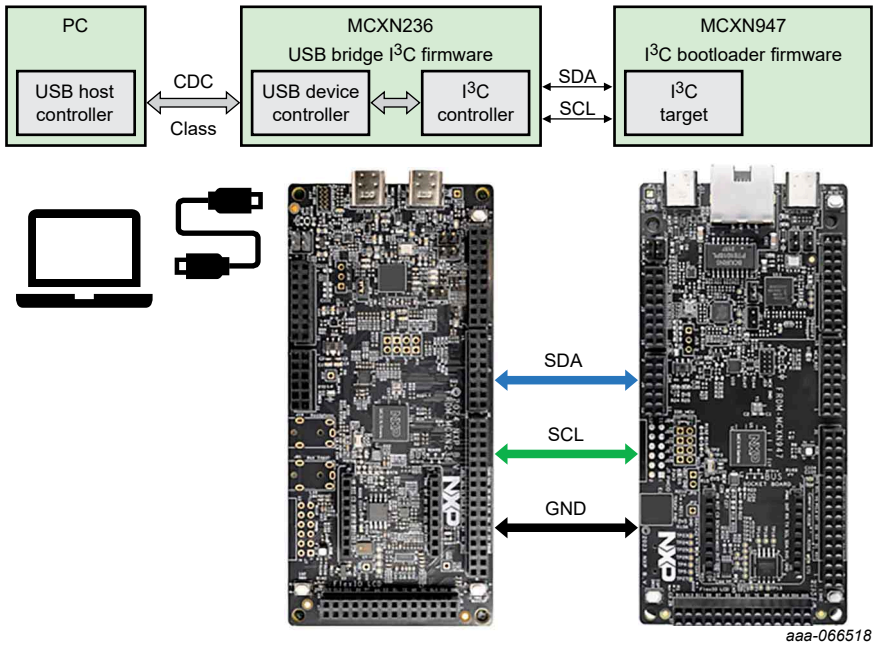

Figure 1. MCX N947 secondary bootloader hardware block diagram

2.1 MCX N947 I3C secondary bootloader software demo introduction

This demo implements the I3C secondary bootloader based on the NXP *blhost* framework. It supports the main commands defined in *Chapter 14* of the MCX N Reference Manual (MCXNP184M150F70RM).

Figure 2 shows the software layout of MCX N947 I3C secondary bootloader demo.

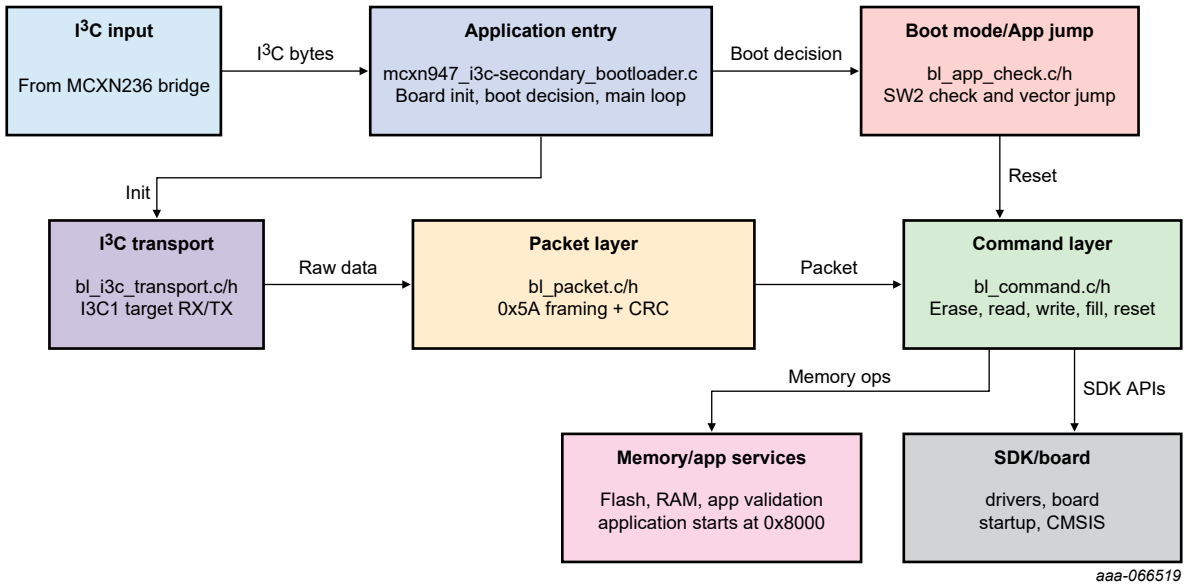

Figure 2. MCX N947 secondary bootloader software layout

The demo includes a Python CLI script, `blhost_i3c_tool.py`, located in the software folder. This script is used to send bootloader commands from the host PC.

The bootloader enters operation mode only when one of the following conditions is met:

- The SW2 button is pressed during reset, or
- No valid application image is present in the application memory region

To ensure bootloader entry, press the SW2 button during any reset event or ensure that the application memory region is empty.

Table 2 outlines the layers of the MCXN947 secondary bootloader firmware.

Table 2. I3C secondary bootloader commands supported on MCX N947

Module	Function
source/mcxn947_i3c_secondary_bootloader.c	Initializes board pins, clocks, debug console, I3C transport, packet layer, and command layer. It also manages bootloader entry and timeout-based application jump.
source/bl_i3c_transport.c/h	Configures I3C1 as target mode, stages RX/TX data, handles I3C interrupts, and supports multi-transaction response reads.
source/bl_packet.c/h	Implements the *blhost* framing packets, including start byte (0x5A), packet type, payload length, CRC16-XMODEM, ACK, NAK, Abort, Ping, Command, and Data packets.
source/bl_command.c/h	Parses *blhost* command payloads, dispatches supported commands, manages data-in and data-out phases, and performs flash/RAM operations.
source/bl_app_check.c/h	Check the application vector table at 0x00008000, validates stack pointer and reset handler, and jumps to the application when required.

Table 3 lists the commands supported by the `blhost_i3c_tool.py` utility along with their corresponding usage formats:

Table 3. Python CLI commands for I3C bootloader operations

Command	Command Format
help	script\blhost_i3c_tool.py help
get-property	script\blhost_i3c_tool.py -p <COMx> get-property <tag>
read-memory	script\blhost_i3c_tool.py -p <COMx> read-memory <address> <length> [--file/-f <file>]
write-memory	script\blhost_i3c_tool.py -p <COMx> write-memory <address> --file/-f <bin>
write-memory	script\blhost_i3c_tool.py -p <COMx> write-memory <address> --data "<hex bytes>"
flash-erase-all	script\blhost_i3c_tool.py -p <COMx> flash-erase-all
flash-erase-region	script\blhost_i3c_tool.py -p <COMx> flash-erase-region <address> <length>
reset	script\blhost_i3c_tool.py -p <COMx> reset
daa	script\blhost_i3c_tool.py -p <COMx> daa <addr_num> [--addrs "0x08 0x09 ..."]

Table 4 describes the command-line options supported by the tool, including default values and their usage.

Table 4. Command-line options for the Blhost I3C tool

Option	Default	Description
--port PORT/-p PORT	none	Serial port; required for all hardware commands
--baud BAUD/-b BAUD	115200	Serial baud rate
--address ADDRESS/-a ADDRESS	0x08	I3C target address
--mode {sdr,i2c,ddr}/-m {sdr,i2c,ddr}	sdr	I3C bus mode

Table 4. Command-line options for the Blhost I3C tool*...continued*

Option	Default	Description
--timeout TIMEOUT/-t TIMEOUT	5000	Serial read timeout in ms
--response-delay DELAY/-d DELAY	50	Delay before reading responses in ms
--verbose/-v	disabled	Accepted for compatibility; packet dumps are not printed

2.2 MCX N947 I3C secondary bootloader demo hardware setup

To perform the MCX N947 I3C secondary bootloader demo, use one FRDM-MCXN236 board as the USB-to-I3C bridge (controller) and one FRDM-MCXN947 board as the I3C secondary bootloader target.

Table 5 lists the hardware connection between the two boards.

Table 5. MCX N947 secondary bootloader firmware layers and functions

Signal	FRDM-MCXN236 (USB-to-I3C bridge)	FRDM-MCXN947 (secondary bootloader target)
I3C SDA	P1_16/I3C_SDA/J9[4]	P1_16/J9[4], configured as I3C1_SDA
I3C SCL	P1_17/I3C_SCL/J9[3]	P1_17/J9[3], configured as I3C1_SCL
Ground	GND	Common GND with FRDM-MCXN236
USB	Connect to PC	Power/debug connection as needed

Figure 3 shows the hardware setup used for the MCXN947 I3C secondary bootloader demo, including the connection between the USB-to-I3C bridge and the target device.

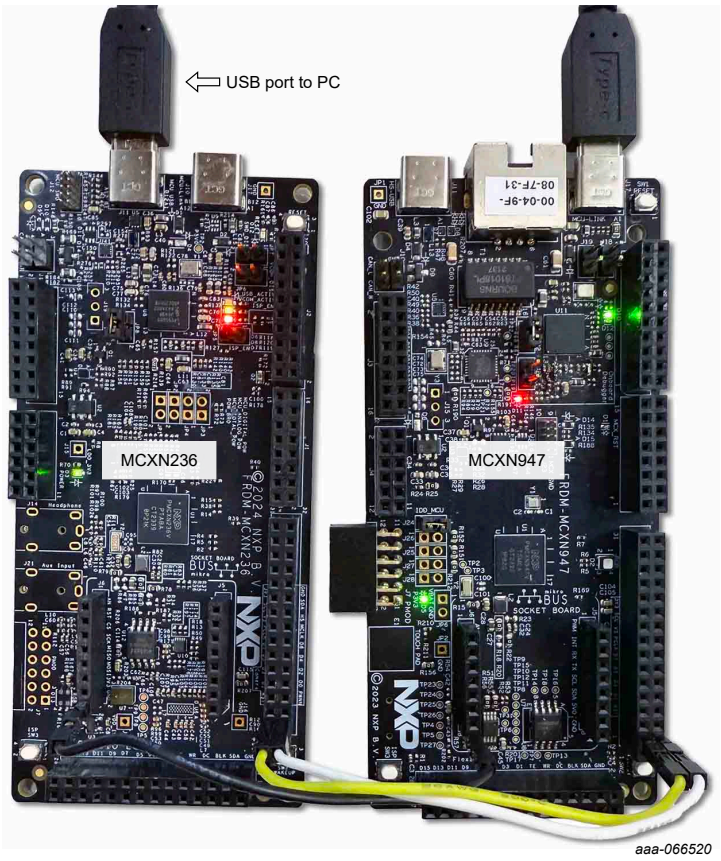

Figure 3. MCX N947 I3C secondary bootloader demo hardware setup

2.2.1 Command test

To test this demo, follow the steps below:

1. To set up the system connection, follow the hardware connections described in Table 5 and Figure 3.
2. Flash the `mcxn236-usb-bridge-to-i3c` firmware to FRDM-MCXN236 board and `mcxn947_bl_secondary_bootloader_i3c` to FRDM-MCXN947 board.
3. Connect the FRDM-MCXN236 USB port (J11) to the PC. The PC counts two USB-CDC COM ports with consecutive COM numbers. On the test system, the PC counts two USB CDC COM ports (COM5 and COM6), as shown in Figure 4. Select the second COM port (COM6).

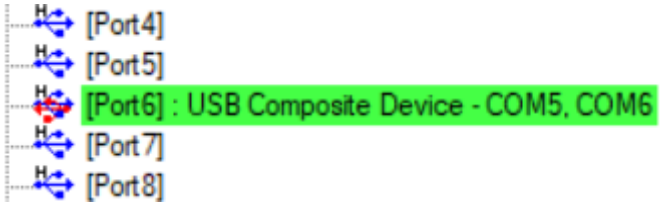

Figure 4. MCX N236 USB bridge I3C COM port

4. Before testing the I3C secondary bootloader, press the SW2 button during reset on the FRDM-MCXN947 board to force the firmware to enter bootloader mode, or perform a mass erase chip before each firmware is downloaded. After the firmware runs into bootloader mode, the terminal displays supported I3C interface parameters, as shown in Figure 5.

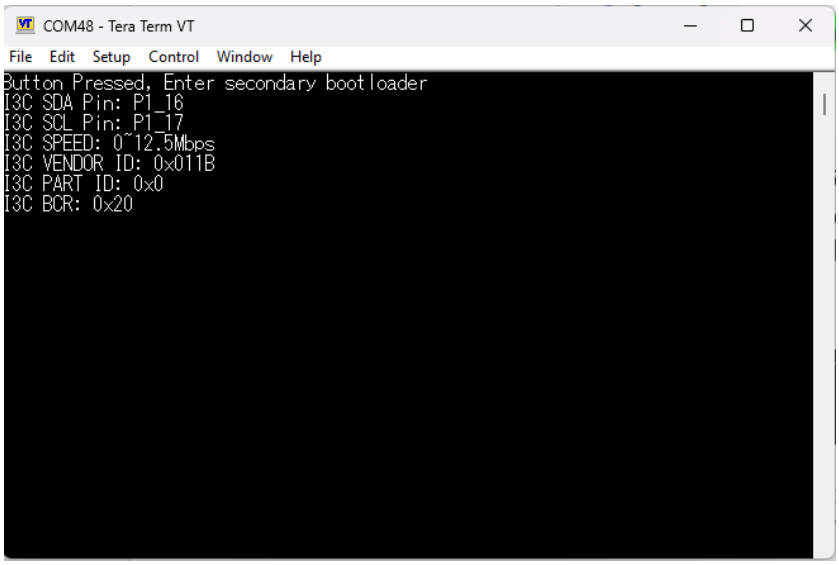

Figure 5. MCXN947 bootloader mode terminal information

5. To run the I3C bootloader, perform DAA to assign a dynamic address to the target I3C device, as shown in Figure 6. DAA can be performed according to the actual I3C target devices on the bus. After DAA is completed, all I3C devices with assigned dynamic addresses can be observed.
   The vendor ID of the I3C interface used by the bootloader is `0x011B`. Based on this vendor ID, the corresponding address (for example, `0x30`) can be determined and used for subsequent commands. The DAA command must be performed after each reset or power cycle.

```
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -v -t 50 -d 50 daa 2 --addrs "0x30 0x31"
daa: requesting 2 device(s), assigning [0x30 0x31]
daa: discovered 2 device(s)
  [0] -> 0x30 vendor=0x011B part=0x00000000 BCR=0x20 DCR=0x00
  [1] -> 0x31 vendor=0x0000 part=0x00000000 BCR=0x00 DCR=0x00
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script>
```

Figure 6. DAA command used by USB-I3C bridge

6. After DAA, execute commands such as *get-property*, *flash-erase-all*, *read-memory*, *write-memory*, and *reset* to validate bootloader functionality.
   Figure 7 shows the *get-property* command. This command can be used to retrieve bootloader information such as firmware version, memory size, device ID, and other details.

```
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -v -t 50 -d 50 -a 0x30 get-property 1
CurrentVersion:
  value[0] = 0x4B020100 (1258422528)
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -v -t 50 -d 50 -a 0x30 get-property 3
FlashStartAddress:
  value[0] = 0x00008000 (32768)
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -v -t 50 -d 50 -a 0x30 get-property 4
FlashSizeInBytes:
  value[0] = 0x001F8000 (2064384)
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -v -t 50 -d 50 -a 0x30 get-property 5
FlashSectorSize:
  value[0] = 0x00002000 (8192)
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script>
```

Figure 7. Get-property command used by USB-I3C bridge

Figure 8 shows the *flash-erase-all* command, which is used to erase all application firmware memory region. For the *flash-erase-all* command, the *-d 300* specifies a delay of 300 ms delay time. In the bootloader code, erasing the memory region from `0x8000` to `0x200000` takes approximately 250 ms based on testing. This delay must follow the MCXN947 data sheet (MCXNP184M150F70) specification for erase sector execution time, which has a typical value of 2 ms according to this data sheet.

```
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -t 50 -d 300 -a 0x30 flash-erase-all
flash-erase-all: Success (0x00000000)
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script>
```

Figure 8. flash-erase-all command used by USB-I3C bridge

Figure 9 shows the read-memory command, which is used to read data from the specified memory region.

```
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -t 50 -d 50 -a 0x30 read-memory 0x8000 256
read-memory: 0x00008000, 256 bytes
  received 256/256 bytes (100.0%)
read-memory: Success (256 bytes received)
00000000  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000010  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000020  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000030  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000040  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000050  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000060  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000070  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000080  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
00000090  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
000000A0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
000000B0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
000000C0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
000000D0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
000000E0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
000000F0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script>
```

Figure 9. read-memory command used by USB-I3C bridge

Figure 10 shows the write-memory command, which is used to download a `.bin` file to the application memory region.

```
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -t 50 -d 50 -a 0x30 write-memory 0x8000 -f .\led_blinky_peripheral.bin
write-memory: 0x00008000, 6136 bytes (48 chunks)
  sent 6136/6136 bytes (48/48, 100.0%)
write-memory: Success (6136 bytes written)
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script>
```

Figure 10. write-memory command used by USB-I3C bridge

Figure 11 shows the reset command. After downloading the application firmware, this command can be used to reset the MCU and run the application code.

```
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script> python -u blhost_i3c_tool.py -p COM6 -t 50 -d 50 -a 0x30 reset
reset: Success
PS C:\_DDM\mcxn947_bl_secondary_bootloader_i3c\script>
```

Figure 11. reset command used by USB-I3C bridge

3 Note about the source code in the document

The example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2026 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR

BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN

ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

4 Revision history

Table 6 summarizes the revisions to this document.

Table 6. Revision history

Document ID	Release date	Description
AN15072 v.1.0	20 July 2026	Initial public release

Legal information

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.

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 https://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.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that 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 vulnerabilities or may support established security standards or specifications with known limitations. 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.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

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

NXP — wordmark and logo are trademarks of NXP B.V.

Contents