AN14916

How to Use SmartDMA to Implement Camera Interface on MCX A366

Rev. 1.0 — 20 April 2026 Application note

Document information

Information	Content
Keywords	AN14916, MCX A366, SmartDMA, FRDM-MCXA366
Abstract	This application note explains how to implement a camera interface on the MCX A366 microcontroller using SmartDMA.

1 Introduction

This application note describes the parallel interface for the camera solution in MCX A366. It provides the introduction, features, API routines, and a demo of the camera interface. MCX A366 features a coprocessor SmartDMA, which can be used to implement the camera interface.

1.1 MCX A366 MCU overview

This application note only focuses on the Digital Video Port (DVP) interface, which is a parallel interface. The MCX A366 MCU features up to 1 MB flash, up to 256 kB SRAM, and a 240 MHz system clock. It also supports SmartDMA and integrates FlexIO, QSPI, and I2C interfaces. The SmartDMA can be used to transfer data from the camera interface to the internal RAM. The FlexIO can be used to transfer the data in RAM to an LCD interface. The QSPI can be used to extend the memory to store the frame data. The internal SRAM can be used to store the temporary frame data. The I2C interface can be used to configure the camera module. It supports one 320x240 frame data size buffer in the RAM.

This application note focuses on the implementation of the camera interface. For the FlexIO implemented LCD interface detail, see *Using FlexIO to Drive 8080 Bus Interface LCD Module* (document AN5313).

1.2 Camera interface

A typical camera interface supports at least one parallel interface, although nowadays many camera interfaces support the MIPI CSI interface.

The camera parallel interface consists of the following lines:

- Data line (D[0:7]): These parallel data lines carry pixel data. The data transmitted on these lines changes with every Pixel Clock (PCLK).
- Horizontal Sync (HSYNC): This signal is a special signal that generates from the camera sensor. An HSYNC signal indicates that one line of the frame has been transmitted.
- Vertical Sync (VSYNC): This signal is transmitted after the entire frame is transferred. This signal is often a way to indicate that one entire frame has been transmitted.
- Pixel Clock (PCLK): This pixel clock changes on every pixel.

This application note only focuses on the DVP interface, which is a parallel interface.

1.3 Target applications

The camera interface can be used as an important part of camera usage that includes:

- Object detection
- Gesture recognition
- Color recognition
- QR code scanning

2 Camera interface features

This application note only focuses on the DVP interface, which is a parallel interface. The supported features are as follows:

- Pixel format: RGB565.
- The maximum image transfer rate is 30 fps for QVGA (320x240). For small RAM parts, reduce the image size and frame rate.
- OV7670 is the tested camera module.

- Other camera modules can be supported if they provide the same signal timings.

3 Functional description

This section describes how SmartDMA interacts with various MCU subsystems to implement the camera pipeline.

3.1 How SmartDMA works

The SmartDMA can access the GPIO in a single system cycle. It reads the data from the camera and stores the data in the RAM. After that, the FlexIO can send the data to an LCD screen.

Some configurations must be made before using the SmartDMA. These configurations include the pin configuration, clock enable, dedicated processor enable, interrupt enable, and so on.

The SmartDMA shares the system clock with the Arm core. To speed up the processing time, configure the system clock to 240 MHz.

3.2 Camera clock source

The camera requires an external clock of approximately 6 MHz, which is supplied by the MCU through its CLKOUT output. Different clock sources can get a different frame rate output.

Note: *Note: The 6 MHz clock is used primarily to achieve a lower pixel clock and frame rate output from the camera. If the camera can be configured to maintain a pixel clock below 12 MHz, the input clock can be defined.*

3.3 MCU8080 LCD interface

The FlexIO can implement an LCD interface. For details, see *Using FlexIO to Drive 8080 Bus Interface LCD Module* (document AN5313).

3.4 I2C interface

The camera is configured through the MCU I2C interface. Before operation, the Arm core configures the camera using the I2C peripheral.

3.5 Memory usage

In this application, the LCD screen resolution is 480x320, and the OV7670 camera module output resolution is 640x480. The SmartDMA crops the size of the image output and only stores the frame size with a resolution of 320x240 in the RAM to adapt to the LCD size. This showcases the flexibility of the SmartDMA. This setup needs about 150 kB (320x240 RGB565 (320x240 × 2 bytes = 153,600 bytes)) of RAM.

To implement sampling of different sizes, use different API functions implemented by the SmartDMA.

A ping-pong buffer can be used to store the partial frame data. It requires timely processing of data to avoid affecting subsequent data storage.

Also, the instruction code of the camera engine must run in the RAM for high performance. This solution uses the SRAMX to store the camera engine code. Since the implementation of SmartDMA is related to the RAM address where the running code is located, the running address must be fixed at the start position of SRAMX. If running SmartDMA code at other RAM addresses, regenerate the code array.

3.6 Other supported camera modules

Other camera modules can be supported if they provide the same signal timing.

1. The camera module must be configured in the RGB565 mode with the timing diagram as shown in Figure 1.

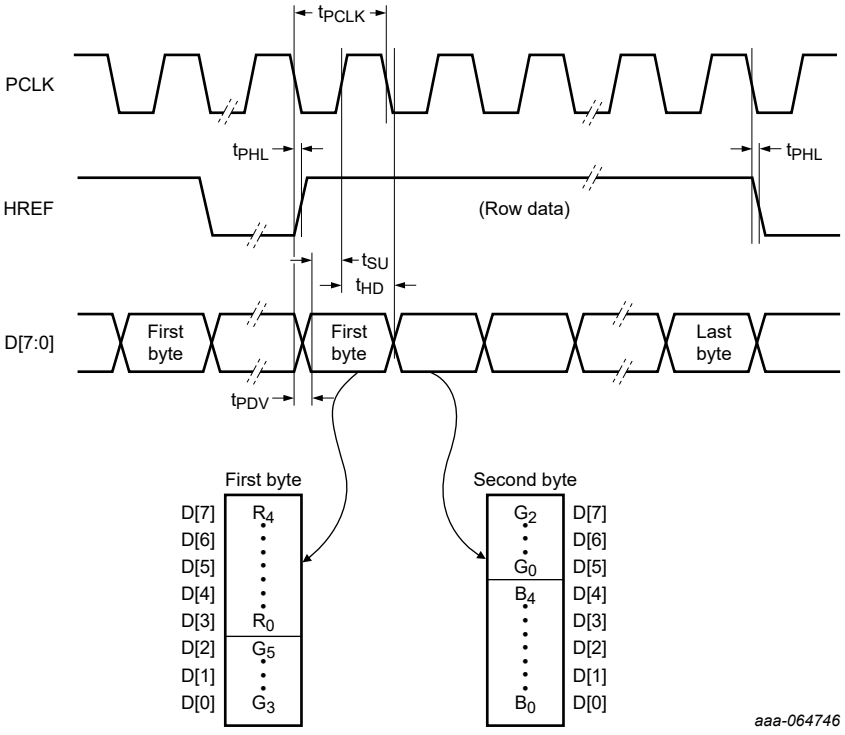

Figure 1. Timing diagram

2. The resolution can be configured as VGA (640x480), QVGA (320x240), QQVGA (160x120), and so on. The SmartDMA code must be modified to adapt to different resolutions.

4 Pin function

This section describes the hardware interface and pin mapping between the MCU and camera module.

4.1 Interface connection

The SmartDMA can access 32 GPIO pins in MCX A366 and use 8 GPIO pins in parallel to read the camera data. The MCU uses I2C to configure the camera. The VSYNC, HREF, PCLK, and PWDN signals can be controlled by SmartDMA. The camera module can be powered by the MCU board with 3.3 V.

4.2 Interface requirements

The D0-D7 are connected to D0-D7 of SmartDMA for the byte reading of data. SIOC and SIOD are connected to the MCU I2C interface for configuration. VSYNC, HREF, and PCLK are connected to the Port0 pins to trigger SmartDMA. XCLK is connected to the clock output pin of the MCU.

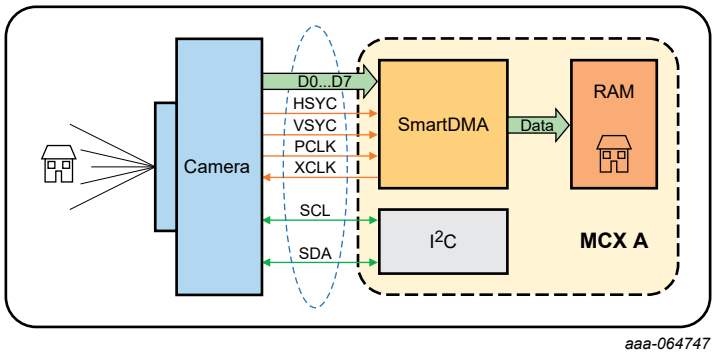

Figure 2. Interface requirements

5 Prerequisites and requirements

This section describes the hardware software prerequisites.

5.1 Hardware requirements

Following is the list of hardware required to run the demo.

- FRDM-MCXA366 development board
- OV7670 (or timing-compatible) DVP camera module
- 8080-style parallel LCD compatible with the provided FlexIO driver
- USB cable for programming/debug
- Jumper wires/ribbon as needed for camera connections

5.2 Software requirements

Following is the list of software required to run the demo.

- MCX A366 SDK with the demo project
- One of the following: MCUXpresso IDE, Keil MDK, or IAR Embedded Workbench
- Board support package for FRDM-MCXA366

6 Software

This section covers the software components, firmware structure, and API routines used in the camera demo.

6.1 Demo code

In the MCX A366 SDK, find the example named `frdmmcxa366_smartdma_camera_flexio_mculcd`.

6.2 SmartDMA code array

As a reduced-instruction-set core, the SmartDMA must execute assembly instructions to perform tasks. To reduce the difficulty of user study, the SmartDMA code is presented in a data-array form in this application. Simply set the SmartDMA address of the array and execute the task. Since there are absolute address jumps instructions, the code array must be placed at a specified RAM address. The code array can be easily integrated in Keil IDE, MCUXpresso IDE, and IAR IDE. In the software code of this application note, the name of this array is `s_smartdmaCameraFirmware`. This firmware can be found in the `fsl_smartdma_fw.c` file.

6.3 API routine

The main purpose of the API routines includes the following:

- Enable the SmartDMA clock
- Configure the IO as a camera interface function
- Initialize the I2C interface
- Enable the interrupt of SmartDMA to indicate that the Arm core data is ready
- Initialize and start the SmartDMA
- LCD initialization
- LCD refresh

6.4 API routine description

Table 1. API routine

Routine	Description
`SMARTDMA_InitWithoutFirmware`	Initialize the SmartDMA clock and reset
`SMARTDMA_InstallFirmware`	Copy the firmware to the RAM
`SMARTDMA_InstallCallback`	Install the complete callback function
`SMARTDMA_Boot`	Boot the SmartDMA to run a program
`SMARTDMA_Deinit`	Deinitialize the SmartDMA
`SMARTDMA_Reset`	Reset the SmartDMA
`SMARTDMA_HandleIRQ`	SmartDMA IRQ
`SDMA_CompleteCallback`	SmartDMA interrupt callback
`DEMO_InitFlexioMcuLcd`	LCD driver initialization
`DEMO_InitPanel`	LCD initialization
`DEMO_DrawWindow`	LCD refresh camera data
`DEMO_InitCamera`	Camera module initialization

6.5 Detailed code description

This section describes the code in detail.

6.5.1 System clock

SmartDMA has limited time to store the data when every pixel edge comes. If the clock frequency of the engine is higher, the time cost is shorter. In this solution, the system clock must be set to 240 MHz when the engine is running. The code to configure the system clock is as follows:

```
BOARD_BootClockPLL240M();
```

6.5.2 I2C interface

For MCX A366, LPI2C2 is used as the I2C function to initialize the camera.

6.5.3 Pin function

Table 2. Pin function

MCX A366	Function number	Input/output	Description
P0_4	7 (SmartDMA function)	Input	Camera engine function DATA0
P0_5	7 (SmartDMA function)	Input	Camera engine function DATA1
P3_2	10 (SmartDMA function)	Input	Camera engine function DATA2
P3_3	10 (SmartDMA function)	Input	Camera engine function DATA3
P3_4	10 (SmartDMA function)	Input	Camera engine function DATA4
P3_5	10 (SmartDMA function)	Input	Camera engine function DATA5
P3_6	10 (SmartDMA function)	Input	Camera engine function DATA6
P3_7	10 (SmartDMA function)	Input	Camera engine function DATA7
P2_26	GPIO	Input	VSYNC
P3_26	GPIO	Input	HSYNC
P4_6	GPIO	Input	PCLK
P4_2	CLKOUT	Output	Camera clock
P1_8	I2C function	Input/output	SDA
P1_9	I2C function	Output	SCL

6.5.4 LCD function

The LCD is used to display the video from the camera in real time. FlexIO and DMA are used to drive the LCD. When SmartDMA completes storing the camera data in the buffer, it gives an interrupt trigger to the Arm core. In the Arm core interrupt, the complete flag (`g_camera_complete_flag`) is set. Once the flag is set, the `DEMO_DrawWindow` routine is called to refresh the LCD.

6.5.5 Callback function

As the other peripheral handler, the SmartDMA handler is implemented by the Arm core when SmartDMA finishes the storage operation.

In the initialization stage, the callback route of the handler is installed. In the callback routine, a flag is set to 1. In the while (1) loop of the main routine, the refreshing operation can be allowed when the flag turns to logic one.

6.5.6 Data buffer

There must be 150 kB of RAM space for one frame of video (240x320 resolution) and the MCX A366 has about 256 kB of RAM. A double buffer is not possible for the whole frame mode. Only one buffer is used. Since the LCD refresh time (33 ms) is shorter than the data storage time (66 ms), the Arm core reads the data for LCD refresh before the SmartDMA write operation is completed. The LCD always displays the previous frame based on the data stored in RAM. Therefore, the media data is not lost.

6.5.7 Timing

The LCD always displays the previous frame data from the camera. Before displaying, the data stored must be optimized by the coprocessor for exchanging the high and low bytes of every pixel. Since the LCD module

display speed is higher than the camera interface data-read speed, this application uses a single data buffer. While the current frame data is stored, the LCD displays the previous frame data.

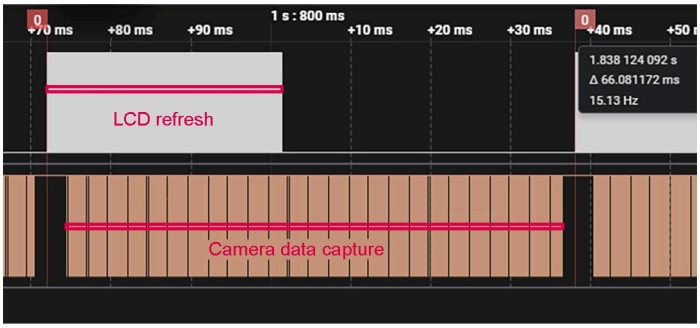

Figure 3. Frame timings between LCD refresh and camera data capture

6.5.8 Demo

Follow the steps below to run the camera interface demo on the FRDM-MCXA366 board:

1. Build the example project.
   Compile the `frdmmcxa366_smartdma_camera_flexio_mculcd` example from the MCX A366 SDK using your preferred IDE.
2. Program the MCU.
   - Connect the FRDM-MCXA366 board to your PC using a USB cable via the Debug Link port.
   - Download the compiled firmware into the MCU.
   - Disconnect the cable from the FRDM-MCXA366 board after programming is complete.
3. Connect the camera module.
   - Attach the OV7670 (or compatible) camera module to the board.
   - Ensure all connections match the pin mapping as shown in Figure 4 ((D0–D7, VSYNC, HREF, PCLK, XCLK, I2C lines, and power).
4. Connect the LCD panel.
   Connect the LCD panel to the FlexIO LCD port on the FRDM-MCXA366 board.
5. Run the demo.
   - Power up the board.
   - The LCD displays the video frame from the camera, as shown in Figure 4.

A single data buffer design is used in this application. While SmartDMA writes the current frame data to RAM, the LCD displays the previous frame data stored in RAM.

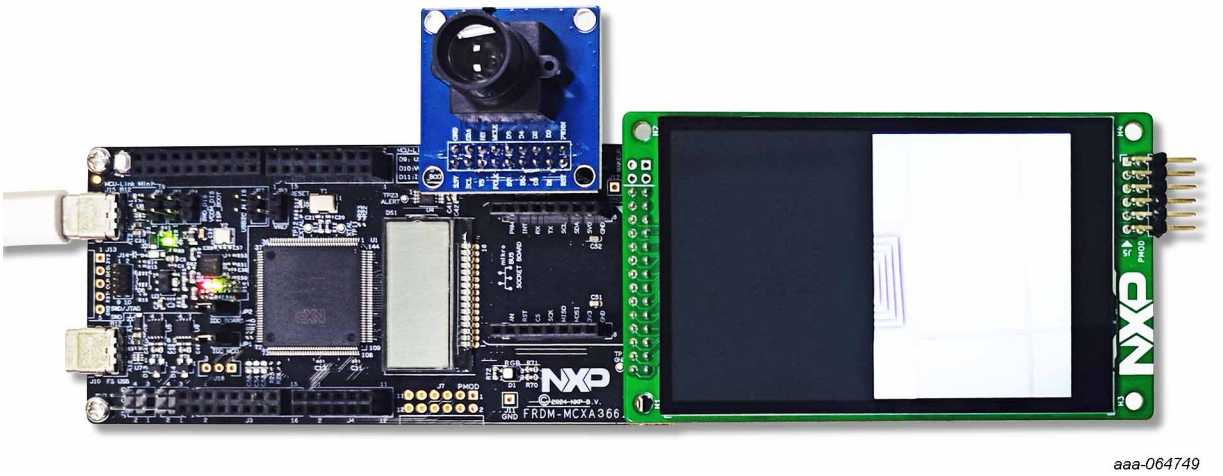

Figure 4. Camera and LCD panel connections

7 Troubleshooting

Table 3 summarizes common LCD issues along with their possible causes and recommended corrective actions.

Table 3. LCD issues, causes, and resolution

Symptom	Possible cause	Action
Blank/white LCD	LCD not initialized; wrong byte order	Initialize panel; confirm RGB565 byte swap is enabled.
Image tearing/rolling	PCLK too high; sync not aligned	Reduce sensor PCLK; verify VSYNC/ HSYNC/HREF polarity.
No frame updates	SmartDMA firmware not at SRAMX base; the callback function not registered	Link firmware array to SRAMX base and install callback.
I2C timeouts	Pull-ups absent/weak; wrong bus speed	Add/verify SDA/SCL pull-ups; set proper LPI2C2 speed.
Incorrect colors	Byte order mismatch per pixel	Ensure a high/low byte swap per pixel before LCD refresh.

8 Acronyms

Table 4 defines the acronyms used in this document.

Table 4. Acronyms

Acronym	Definition
API	Application Programming Interface
CSI	Camera Serial Interface
DVP	Digital Video Port
FlexIO	Flexible Input/Output
GPIO	General-purpose Input/Output
I2C	Inter-Integrated Circuit

Table 4. Acronyms...*continued*

Acronym	Definition
IDE	Integrated Development Environment
IRQ	Interrupt Request
MIPI	Mobile Industry Processor Interface
QSPI	Quad Serial Peripheral Interface
QVGA	Quarter Video Graphics Array
SCL	Serial Clock
SDA	Serial Data
SDK	Software Development Kit

9 Note about the source code in the document

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.

10 Revision history

Table 5 summarizes the revisions to this document.

Table 5. Revision history

Document ID	Release date	Description
AN14916 v.1.0	20 April 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