AN14175

Using FlexIO to emulate Quad SPI Controller

Rev. 2.0 — 21 July 2025 Application note

Document information

Information	Content
Keywords	AN14175, FlexIO module, board interfaces, MCX-N947-EVK, i.MX RT595-EVK hardware platforms, simplex Quad SPI controller, LCD driver
Abstract	This application note describes the implementation of Flex IO peripheral as a simplex Quad SPI controller for LCD driver on MCX-N947-EVK or i.MX RT595-EVK hardware platforms.

1 Introduction

Quad SPI serves as a common interface for flash memory, Wi-Fi modules, and LCD displays. However, some microcontrollers do not support the Quad SPI interface. In such cases, FlexIO offers a versatile alternative.

FlexIO runs on Kinetis, S32K, RT, and MCX microcontroller families. It is highly configurable and emulates a wide range of communication protocols, including Universal Asynchronous Receiver/Transmitter (UART), Inter-Integrated Circuit (I2C), Serial Peripheral Interface (SPI), I2S, and Local Interconnect Network (LIN), as described in *Using FlexIO to emulate communications and timing peripherals* (document AN12174. It also supports other protocols, such as J1850, I3C, and Manchester encoding. J1850, I3C, Manchester.

The key feature of FlexIO is that it enables users to build their own peripherals directly, offering a high degree of flexibility in embedded system design. This application note demonstrates that flexibility by implementing a Quad SPI interface on the RT500-EVK platform, tested using a NOR flash memory (Pmod SF3). A simplified version of this implementation also runs on the MCX-N947, which supports only simplex transmission. Despite this limitation, the design remains suitable for applications such as driving an LCD display.

2 Overview of the FlexIO module

FlexIO module has the following main hardware resources:

- Shifter
- Timer
- Pin

Figure 1 shows a high-level overview of the FlexIO module.

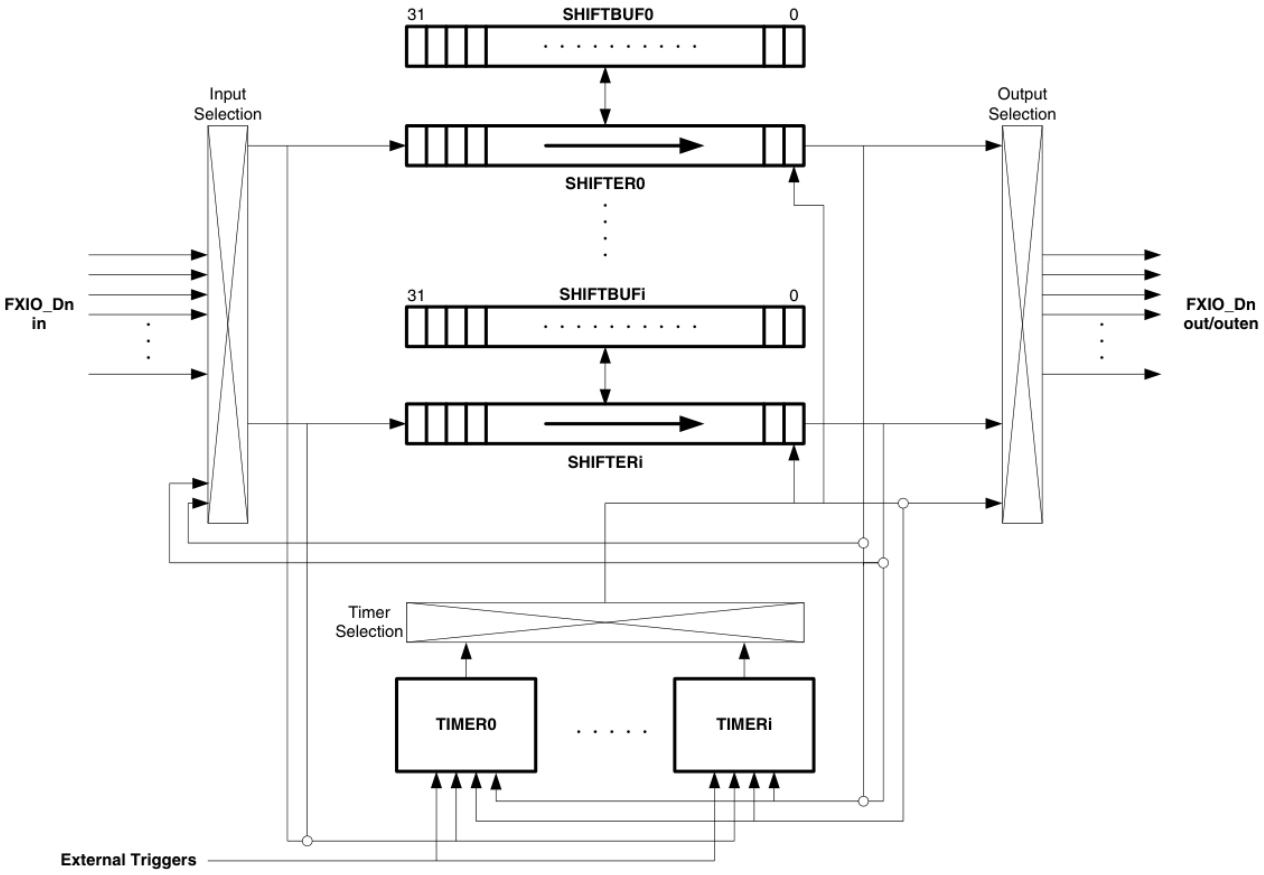

Figure 1. FlexIO block diagram

The following key features are provided:

- 32-bit shifters with transmit, receive, and data match modes
- Double buffered shifter operation
- 16-bit timers with high flexibility support for various internal or external triggers, and reset, enable, disable, and decrement conditions
- Automatic start or stop bit generation, or check
- Interrupt, direct memory access (DMA), or polling mode operation
- Shifters, timers, pins, and triggers can be flexibly combined to operate

Transmit and receive are two basic modes of the shifters. If one shifter is configured to Transmit mode, it loads data from its buffer register and shifts data out to its assigned pin bit by bit. If one shifter is configured to Receive mode, it shifts data in from its assigned pin and stores data in its buffer register. The timer assigned to the shifter controls all the load, store, and shift operations. The timers can also be configured in different operational modes as per your requirement. The timer modes include the dual 8-bit counter baud or bit mode, dual 8-bit counter pulse with modulation (PWM) mode, and a single 16-bit counter mode.

For more details, refer to the application note, *Using FlexIO to emulate communications and timing peripherals* (document AN12174) and *i.MX RT500 Low-Power Crossover MCU Reference Manual* (document IMXRT500RM).

3 Emulating Quad SPI controller using FlexIO

This section describes how to emulate a Quad SPI controller using FlexIO.

3.1 Timing chart

Figure 2 shows a typical timing chart of Quad SPI flash. The falling edge of CLK shifts out the date, and the rising edge of CLK samples it.

The communication starts from the command signal (one byte) and address signals (three bytes). Some dummy cycles are inserted at the head of data.

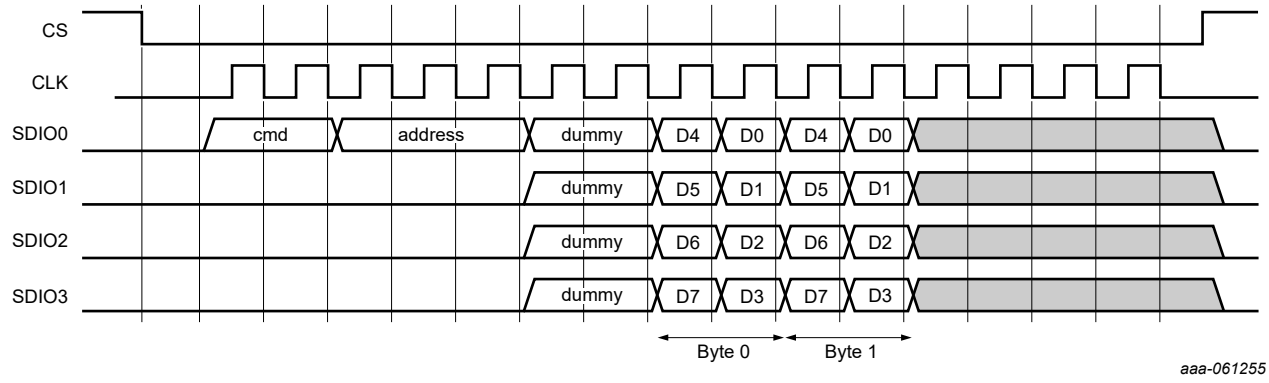

Figure 2. Waveform

3.2 Simplex Quad SPI configuration

The basic concept is the same as the standard SPI explained in *Using FlexIO to emulate communications and timing peripherals* (document AN12174). The key difference lies in the pin width (PWIDTH) register. When the PWIDTH register is set to 3, the system shifts 4 bits per cycle as shown in Figure 3.

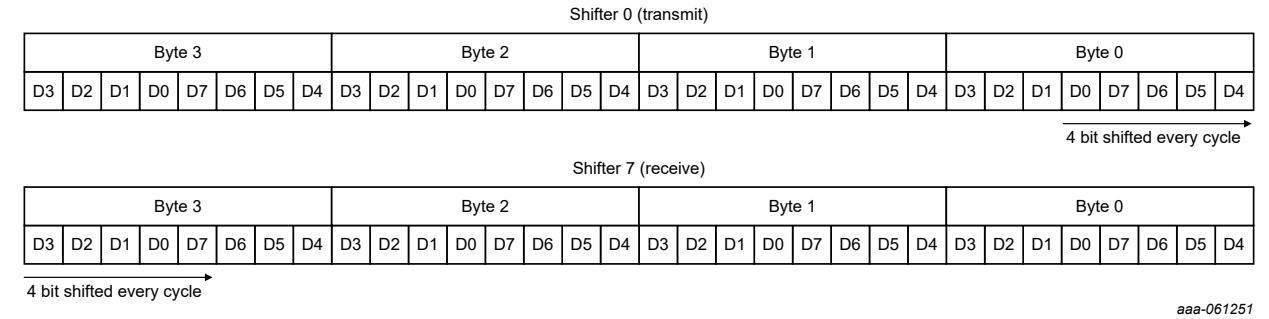

Figure 3. PWIDTH register

To store nibble-swapped data to the shift buffer, the Shifter Buffer N Nibble Byte Swapped register (SHIFTBUFNBS) is used as described in Table 1.

Table 1. SHIFTBUFNBS register

Field	Description
31-0 SHIFTBUFNBS	SHIFTBUFNBS register Acts as a shift buffer alias. Read and write operations on this register result in nibble-swapped within each byte.

Table 1. SHIFTBUFNBS register

Field	Description
	Read return: {SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4]}

Timer 0 and GPIO:

The Quad SPI controller uses Timer 0 to generate SPI_SCK output and load, store, or shift control for the shifter. The SPI controller uses GPIO to generate the SPI_CS output. Refer to the *Using FlexIO to emulate communications and timing peripherals* (document AN12174) for more information on Timer 0 or 1 configuration.

Configurations for shifter 0:

Shifter 0 functions as the transmitter. It uses the following configurations:

Table 2. Configurations for Shifter 0

Items	Configurations
Shifter mode	Transmit
Timer selection	Timer 0
Timer polarity	On the negative edge of the shift clock
Pin configuration	Pin output
Pin polarity	Active high
Pin width	3
Input source	From pin
Start bit	Disabled (transmitter loads data on enable)
Stop bit	Disabled
Buffer used	Nibble byte swapped register

Configurations for Shifter 7:

Shifter 7 functions as the receiver. It uses the following configurations:

Table 3. Configuration for Shifter 7

Items	Configurations
Shifter mode	Receive
Timer selection	Timer 0
Timer polarity	On positive of shift clock
Pin configuration	Pin output disabled
Pin polarity	Active high
Pin width	3
Input source	From pin
Start bit	Disabled
Stop bit	Disabled
Buffer used	Nible byte swapped register

Figure 4 shows the block diagram of the timers and shifters. Shifter 0 acts as the transmitter and shifter 7 acts as the receiver. As both use the same pins, the transmitter disables the output when the receiver reads data. This action tri-states the output of the transmitter, which is achieved by calling `FLEXIO_QSPI_SetDirection` within the interrupt handler.

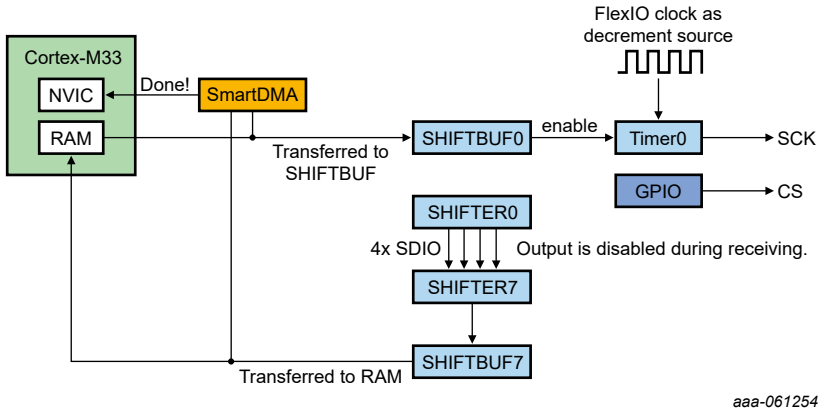

Figure 4. Timers and shifters block diagram

3.3 Software implementation overview

The RT595 software example supports a full-featured SmartDMA-based implementation. The MCX N947 example provides a subset of that functionality using eDMA. As a result, RT595 uses both fsl_flexio_qspi.c/h and fsl_flexio_qspi_smartdma.c/h to enable advanced data operations. In contrast, the MCX N947 relies on fsl_flexio_qspi.c/h and fsl_flexio_qspi_edma.c/h, reflecting its more limited, yet still practical, implementation.

3.3.1 Functional description

The functions available in the drivers of the FlexIO QSPI examples are presented in Table 4, Table 5, and Table 6.

Note: *The start address of the buffer must be 4-byte aligned.*

Table 4. fsl_flexio_qspi.c or fsl_flexio_qspi.h

Function	Description
FLEXIO_QSPI_MasterGetDefaultConfig	Gets the default configuration for the FlexIO QSPI controller
FLEXIO_QSPI_MasterInit	Initializes FlexIO module for FlexIO QSPI controller
FLEXIO_QSPI_MasterTransferCreateHandle	Initializes the FlexIO QSPI controller handle, which is used in Interrupt mode
FLEXIO_QSPI_MasterTransferNonBlocking	Starts transfer in Interrupt mode

Table 5. fsl_flexio_qspi_smartdma.c or fsl_flexio_qspi_smartdma.h

Function	Description
FLEXIO_QSPI_TransferCreateHandle SMARTDMA	Initializes the FlexIO QSPI controller handle, which is used in SmartDMA mode
FLEXIO_QSPI_TransferSMARTDMA	Starts transfer in SmartDMA mode

Table 6. fsl_flexio_qspi_edma.c or fsl_flexio_qspi_edma.h

Function	Description
FLEXIO_QSPI_MasterTransferCreateHandleEDMA	Initializes the FlexIO QSPI controller handle, which is used in eDMA mode
FLEXIO_QSPI_MasterTransferEDMA	Starts transfer in eDMA mode

4 Running the demo

This section describes how to run the associated project on our EVK platform.

4.1 Hardware setup

On MCX-N9XX-EVK, connect Quad SPI controller and the Flexcomm SPI device on the same board, as shown in Table 7. The SPI interface cannot receive data on all lines (SDO0-SDO3) simultaneously but can receive data from one SDO line at a time and validate it.

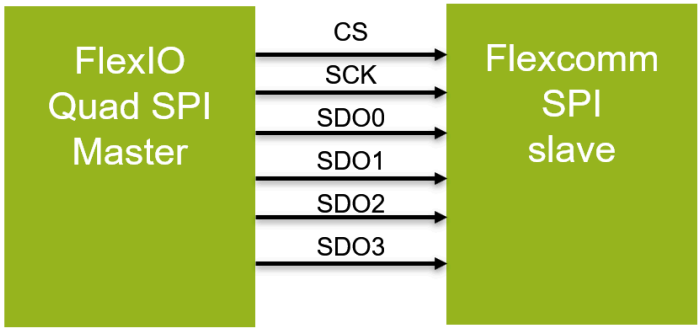

Figure 5. Wire connection

On the RT595-EVK, connect the Quad SPI controller and the flash (Pmod SF3) as shown in Figure 6 and Table 8.

Note: *If you use RT595-EVK, disconnect JS23 from 1 to 2, and connect JS23-2 to JP23-3 to supply 1.8 V to VDDIO_3.*

Table 7. Pin assignment for MCX-N9XX-EVK

Pin assignment	FlexIO Quad SPI	Flexcomm SPI
CS	J20-24	J2-6
SCK	J20-23	J2-12
SDO0/1/2/3	J20-25/J20-26/J20-27/J20-28	J2-10

Table 8. Pin assignment for RT595-EVK

Pin assignment	FlexIO Quad SPI	Pmod SF3
CS	J28-2	J1-1
SCK	J28-1	J1-4
SDO0	J28-3	J1-2
SDO1	J28-4	J1-3

Table 8. Pin assignment for RT595-EVK*...continued*

Pin assignment	FlexIO Quad SPI	Pmod SF3
SDO2	J28-5	J1-9
SDO3	J28-6	J1-10
GND	J28-7	J1-5
VCC	J28-8	J1-6

Note: *The pinout used for the MCX-N9XX-EVK in this example remains fully compatible with the MCX-N5XX-EVK and the FRDM-MCXN947. Therefore, the example runs without modification on any of the MCX evaluation platforms.*

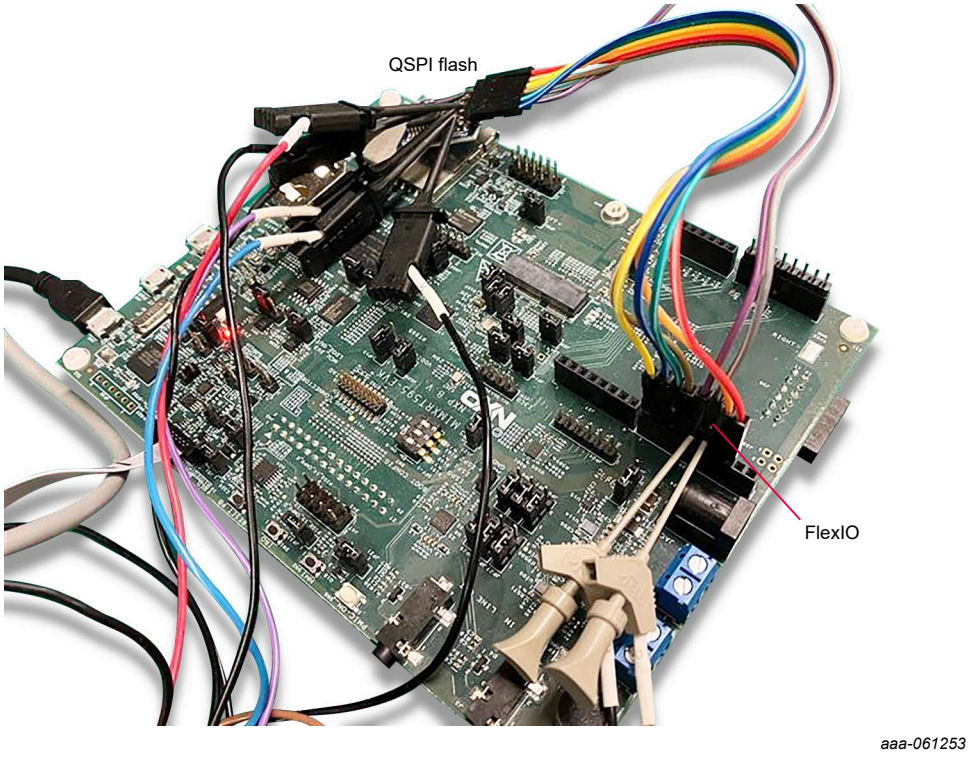

Figure 6. QSPI flash connection

4.2 Software setup

For MCX-N9XX-EVK, install MCUXpresso IDE v11.8.0 and MCUXpresso SDK v2.13.1. For RT595-EVK, install MCUXpresso IDE v24.12.148 and MCUXpresso SDK v24.12.00. Follow the steps below:

1. Open the project from the file system.
2. Connect a mini USB cable between the PC host and the OpenSDA USB port on the board.
3. Open a serial terminal on the PC for the OpenSDA serial device using the following settings:
   - 115,200 baud rate
   - 8 data bits
   - No parity
   - One stop bit
   - No flow control
4. Download the program to the target board.

5. To start running the demo, press the reset button on your board or launch the debugger in the IDE.

4.3 Result

Figure 7 shows the signal for command, address and dummy. It matches with the expected timing chart shown in Figure 2.

Figure 8 shows the signal of read data from flash, and Figure 9 shows the serial console then. This confirms that the software receives correct data.

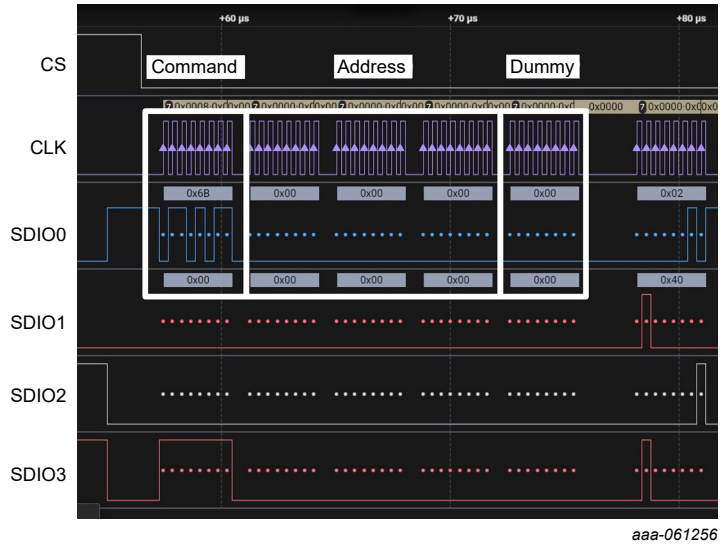

Figure 7. Signal of command, address and dummy

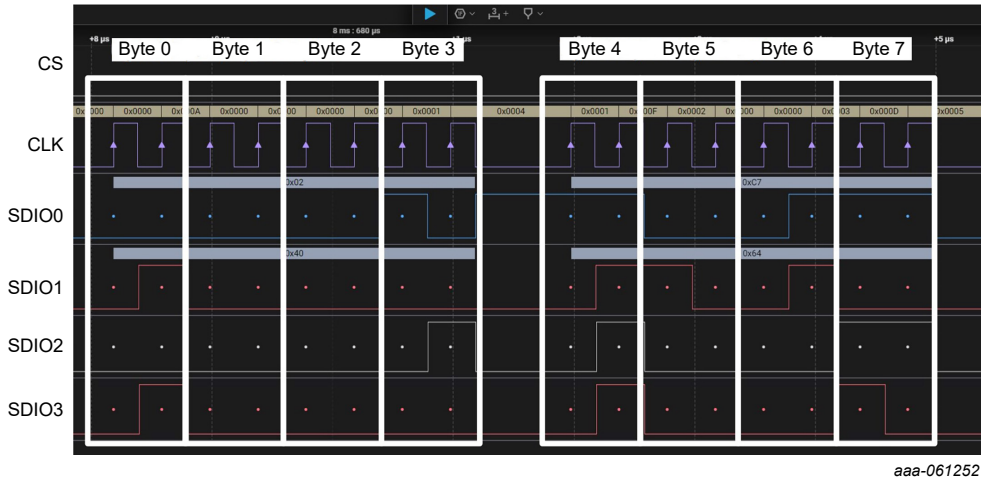

Figure 8. Signal of read data from flash

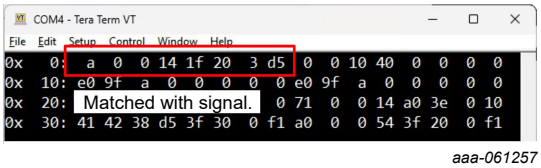

Figure 9. Serial console

5 References

Table 9 lists the references used to supplement this document.

Table 9. Related documentation/resources

Document	Link/how to access
MCX Nx4x Reference Manual (document MCXNX4XRM)	MCXNX4XRM
Using FlexIO to emulate communications and timing peripherals (document AN12174)	AN12174
i.MX RT500 Low-Power Crossover MCU Reference Manual (document IMXRT500RM).	IMXRT500RM

6 Acronyms

Table 10 lists the acronyms used in this document.

Table 10. Acronyms

Acronym	Description
DMA	Direct memory access
PWM	Pulse width modulation
SPI	Serial peripheral interface
QSPI	Quad serial peripheral interface

7 Note about the source code in the document

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

Copyright 2025 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 must be 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.

8 Revision history

Table 11 summarizes the revisions to this document.

Revision history

Document ID	Release date	Description
AN14175 v.2.0	21 July 2025	Updated: Section 1 and Section 3
AN14175 v.1.0	20 January 2024	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

1 Introduction 2
2 Overview of the FlexIO module 2
3 Emulating Quad SPI controller using FlexIO 4
3.1 Timing chart 4
3.2 Simplex Quad SPI configuration 4
3.3 Software implementation overview 6
3.3.1 Functional description 6
4 Running the demo 7
4.1 Hardware setup 7
4.2 Software setup 8
4.3 Result 9
5 References 10
6 Acronyms 10
7 Note about the source code in the document 11
8 Revision history 11
Legal information 12