MCUSDKAZRTOSGSUG

Getting Started with MCUXpresso SDK for Azure RTOS

Rev. 2.13.0 — 30 November 2022 User guide

Document information

Information	Content
Keywords	AzureRTOS, Example, ThreadX, FileX, NetX, GUIX, USBX
Abstract	This document lists the steps to get started with MCUXpresso SDK for AzureRTOS.

1 Overview

The MCUXpresso Software Development Kit (SDK) is a comprehensive software enablement package designed to simplify and accelerate application development with Kinetis, LPC, and i.MX MCUs based on Arm® Cortex -M cores. The MCUXpresso SDK includes production-grade software with integrated Azure RTOS, which is an advanced Industrial Grade Real-Time Operating System (RTOS) from Microsoft designed specifically for deeply embedded, real-time, and IoT applications. It provides advanced scheduling, communication, synchronization, timer, memory management, and interrupt management facilities. Azure RTOS includes some components, for example, ThreadX, FileX, NetX Duo, GUIX, USBX. For more information, please refer to the Azure RTOS Documentation (https://docs.microsoft.com/en-us/azure/rtos/).

2 Source description

- Section 2.1
- Section 2.2
- Section 2.3
- Section 2.4
- Section 2.5
- Section 2.6

2.1 Azure RTOS ThreadX

Azure RTOS ThreadX provides advanced scheduling, communication, synchronization, timer, memory management, and interrupt management facilities. In addition, it has many advanced features, including its picokernel architecture, preemption-threshold scheduling, event-chaining, execution profiling, performance metrics, and system event tracing.

ThreadX source code is located in `<SDK_DIR>/rtos/azure-rtos/threadx`.

2.2 Azure RTOS FileX (LevelX)

Azure RTOS FileX embedded file system is an advanced, industrial grade solution for Microsoft FAT file formats. It supports all the Microsoft file formats, including FAT12, FAT16, FAT32, and exFAT. FileX also offers optional fault tolerance and FLASH wear leveling via an add-on product called Azure RTOS LevelX. Azure RTOS FileX is fast due to direct data write and cache optimized for speed and is easy to use due to consistent API. It includes extensive out of box examples and has been certified to ASIL D and SIL 4.

FileX source code is located in `<SDK_DIR>/rtos/azure-rtos/filex`.

LevelX source code is located in `<SDK_DIR>/rtos/azure-rtos/levelx`.

2.3 Azure RTOS NetX Duo

Azure RTOS NetX Duo embedded TCP/IP network stack is an advanced, industrial grade dual IPv4 and IPv6 TCP/IP network stack from Microsoft designed specifically for deeply embedded, real-time, and IoT applications. NetX Duo provides embedded applications with core network protocols such as IPv4, IPv6, TCP, and UDP as well as a complete suite of additional, higher-level add-on protocols. Azure RTOS NetX Duo is also secure

via additional add-on security products, including Azure RTOS NetX Secure IPsec and Azure RTOS NetX Secure SSL/TLS/DTLS.

Azure RTOS NetX Duo provides near wire speed and requires minimal CPU usage and it is designed for performance with zero copy and integrate with hardware features. It also provides the fastest possible UDP processing and provides a BSD-compatible socket interface which helps in migrating existing network application code to NetX Duo. NetX Duo has been certified to ASIL D and SIL 4.

NetX Duo source code is located in `<SDK_DIR>/rtos/azure-rtos/netxduo`

2.4 Azure RTOS GUIX

Azure GUIX embedded GUI is an advanced, industrial grade GUI solution from Microsoft designed specifically for deeply embedded, real-time, and IoT applications. Microsoft also provides a full-featured WYSIWYG desktop design tool named Azure RTOS GUIX Studio, which allows developers to design their GUI on the desktop and generate Azure RTOS GUIX embedded GUI code that can then be exported to the target. Azure RTOS GUIX is fully integrated with Azure RTOS ThreadX RTOS and is available for many of the same processors supported by Azure RTOS ThreadX.

The Azure RTOS GUIX package includes various sample user interfaces, including a medical device reference, a smart watch reference, a home automation reference, an industrial control reference, an automotive reference, and various sprite and animation examples.

GUIX source code is located in `<SDK_DIR>/rtos/azure-rtos/guix`.

2.5 Azure RTOS USBX

Azure RTOS USBX is a high-performance USB host, device, and on-the-go (OTG) embedded stack. Azure RTOS USBX is fully integrated with Azure RTOS ThreadX and available for all ThreadX–supported processors.

It is designed for speed and has minimal internal function call layering and support for cache and DMA utilization. It has comprehensive class support and device/host controller integration.

USBX source code is located in `<SDK_DIR>/rtos/azure-rtos/usbx`.

2.6 Azure RTOS ThreadX Modules

To dynamically load separately built modules from the resident portion of the application, the ThreadX Module component provides an infrastructure for applications. The component is especially useful in situations where the application code size exceeds available memory. The ThreadX Module component can also help to add new features after the core image is deployed. Additionally, the dynamically loading modules are used when partial firmware updates are required .

The source code for the ThreadX Modules is at: `<SDK_DIR>/rtos/azure-rtos/threadx/common_modules` and `<SDK_DIR>/rtos/azure-rtos/threadx/ports_module`.

3 Azure RTOS example applications

The SDK provides a set of Azure RTOS-related applications. The examples are written to demonstrate Azure RTOS features and the interaction between peripheral drivers and the RTOS.

For more information about how to use these example projects, see the Getting Started with MCUXpresso SDK document.

The source code of examples is located in `<SDK_DIR>/boards/<board name>/azure_rtos_examples/`.

4 Azure RTOS library projects

To save compile time, the SDK contains pre-built libraries for these Azure RTOS components: **ThreadX**, **FileX**, **GUIX**, **NetXDuo**, **USBX**. Each library has a prefix **lib**, and a suffix **.a** or **.lib**. The libraries are available in the SDK directory, **./rtos/azure-rtos/binary/**.

The default libraries are compiled with double-precision floating-point option. For example, **-mfpu=fpv5-d16**.

The libraries with **_sp** in name are compiled with single-precision floating-point option. For example, **-mfpu=fpv5-sp-d16**.

Note:

- *LPC5500 series MCU and i.MX RT1010 only support single-precision floating-point.*
- *There is no pre-built library for ArmGCC projects. The Arm GCC projects generate libraries at runtime.*

If header files or source code changes, the libraries must be rebuilt. For example, **tx_user.h, fx_user.h,** and **ux_user.h**. These libraries have dependency relationship. For example, **FileX** library depends on **ThreadX** library, **NetX Duo** library depends on **FileX** library, and **ThreadX** library. So, if one library is updated, other dependent libraries must be updated. For example, if **FileX** library changes **NetX Duo** and **USBX** libraries must be updated.

In MCUXpresso IDE, the library order in the **Libraries** panel is important. For example, the library order in the **azure_iot_mqtt** example is **netxduo**, **filex**, and **threadx**. This order is important because **NetX Duo** depends on **FileX** and **ThreadX**, and **FileX** depends on **ThreadX**.

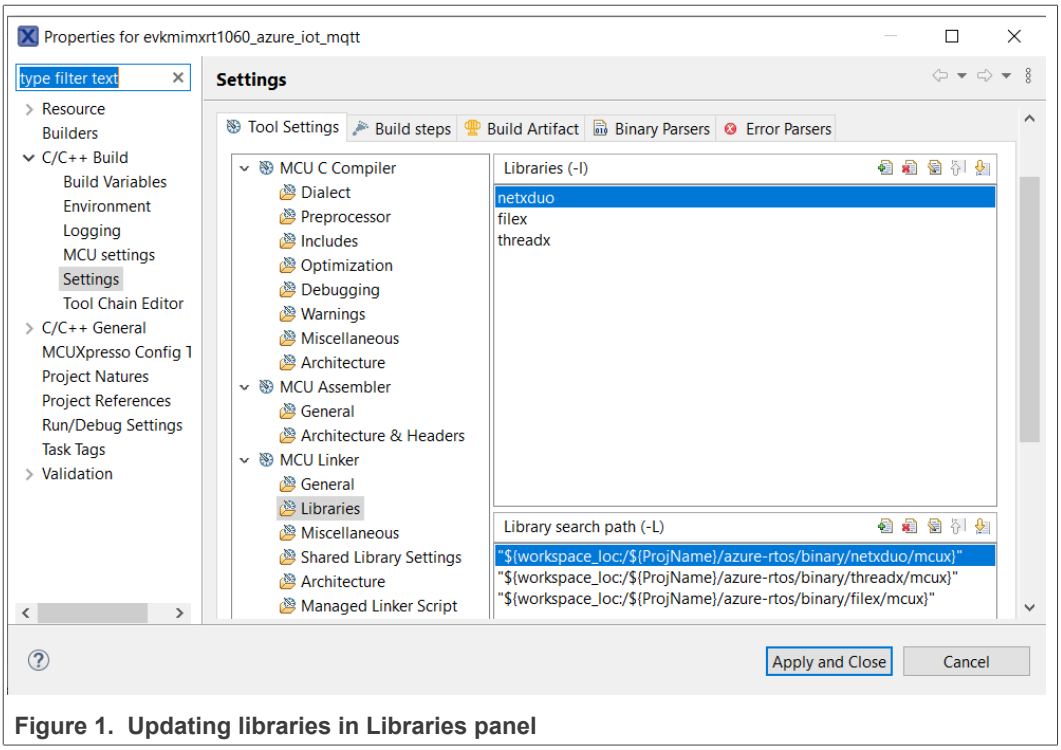

Figure 1. Updating libraries in Libraries panel

4.1 Build and use a new library

To build and use a new library in MCUXpresso IDE, perform these steps. Ensure that MCUXpresso SDK with Azure RTOS is pre-installed.

1. On the **Quickstart Panel**, click **Import SDK example(s)**.
2. In the **Import** dialog box, select a board, click **Next**, and then select a library you want to build. For example, threadx_lib.
3. Import the library project.

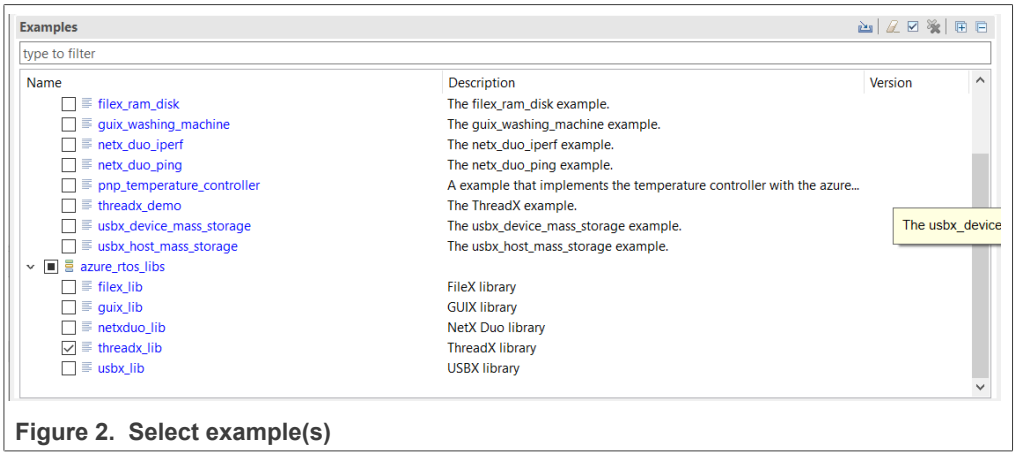

Figure 2. Select example(s)

4. Right-click the threadx_lib project in the workspace and select **Build Project** to start building.
5. After successful build and compilation, a new library appears in the Debug/Release folder. It has a name like **libevkmimxrt1060_threadx_lib.a**.

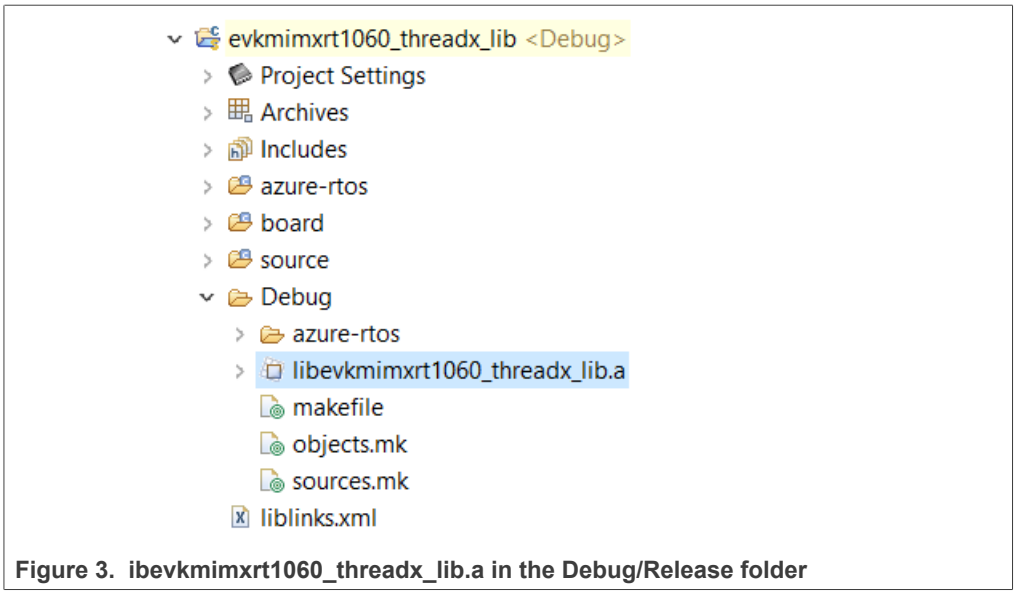

Figure 3. ibevkmimxrt1060_threadx_lib.a in the Debug/Release folder

6. There are two methods to use the new library in an application project. Take the **evkmimxrt1060_threadx_demo project** as an example.
   a. If the new library will not change often, rename it to **libthreadx.a** and copy it to the directory, **azure-rtos/binary/threadx/mcux/**, in the **evkmimxrt1060_threadx_demo project** to replace the original one.

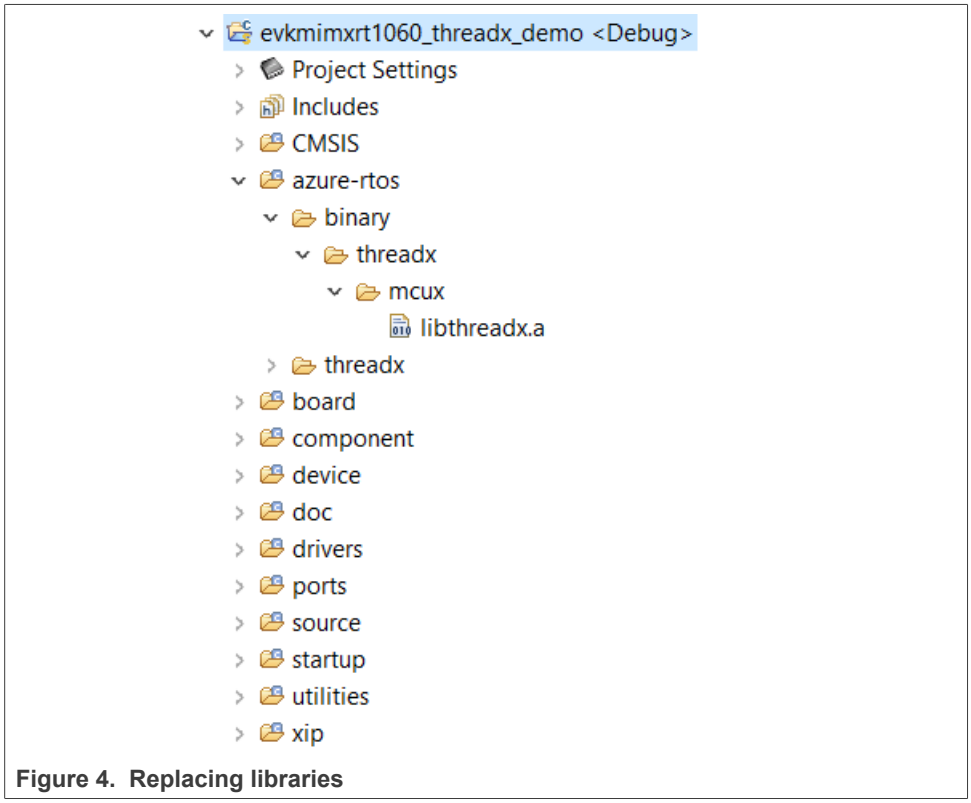

Figure 4. Replacing libraries

   b. If the new library will change frequently, change library settings in application project properties to directly use the new library. In the **Properties** dialog box, change the library name from **threadx** to **evkmimxrt1060_threadx_lib** and

change **Library search path** to **"${workspace_loc:/evkmimxrt1060_threadx_lib/Debug}"**.
Note: *Do not change the order of the libraries.*

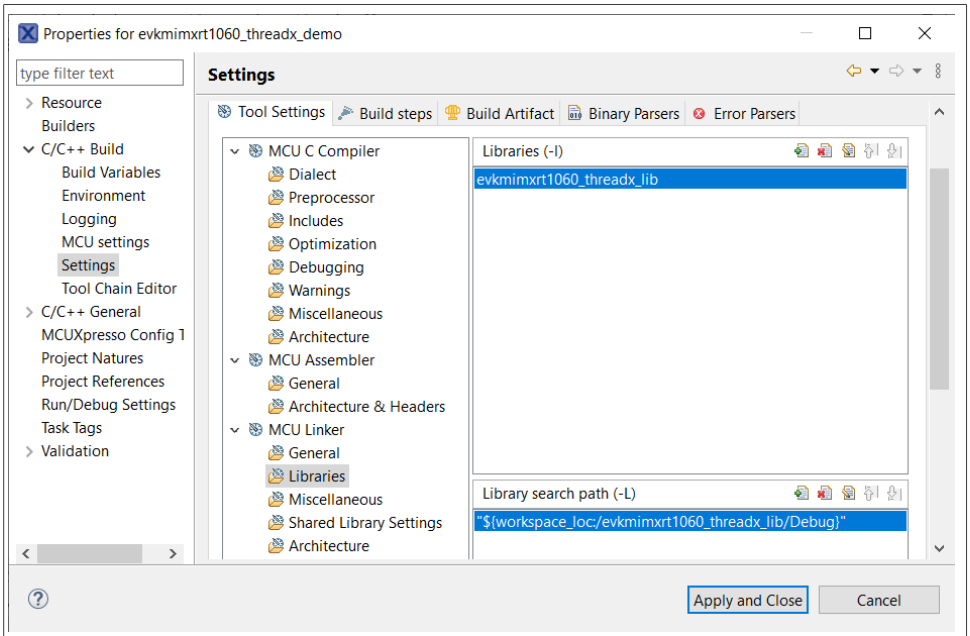

Figure 5. Changing library name

5 Revision history

This table summarizes revisions to this document.

Table 1. Revision history

Revision number	Date	Substantive changes
0	20 December 2020	Initial release
2.10.0	10 July 2021	Updated for 2.10.0
2.12.0	03 June 2022	Updated for 2.12.0
2.13.0	30 November 2022	Updated for 2.13.0

6 Legal information

6.1 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.

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.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://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 data sheet 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.

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.

6.3 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 Overview 2
2 Source description 2
2.1 Azure RTOS ThreadX 2
2.2 Azure RTOS FileX (LevelX) 2
2.3 Azure RTOS NetX Duo 2
2.4 Azure RTOS GUIX 3
2.5 Azure RTOS USBX 3
2.6 Azure RTOS ThreadX Modules 3
3 Azure RTOS example applications 4
4 Azure RTOS library projects 4
4.1 Build and use a new library 5
5 Revision history 7
6 Legal information 8

For more information, please visit: http://www.nxp.com

Date of release: 30 November 2022
Document identifier: MCUSDKAZRTOSGSUG
Document number: