

AN14645

How to Use Random Static Device Address for Bluetooth Application

Rev. 1.0 — 16 April 2025

Application note

Document information

Information	Content
Keywords	AN14645, random static address, bluetooth low energy
Abstract	This document introduces how to enable Random Static Device Address for a Bluetooth Low Energy application.



1 Introduction

This document introduces how to enable Random Static Device Address for a Bluetooth Low Energy application. The default device address type in the SDK is Public Device Address.

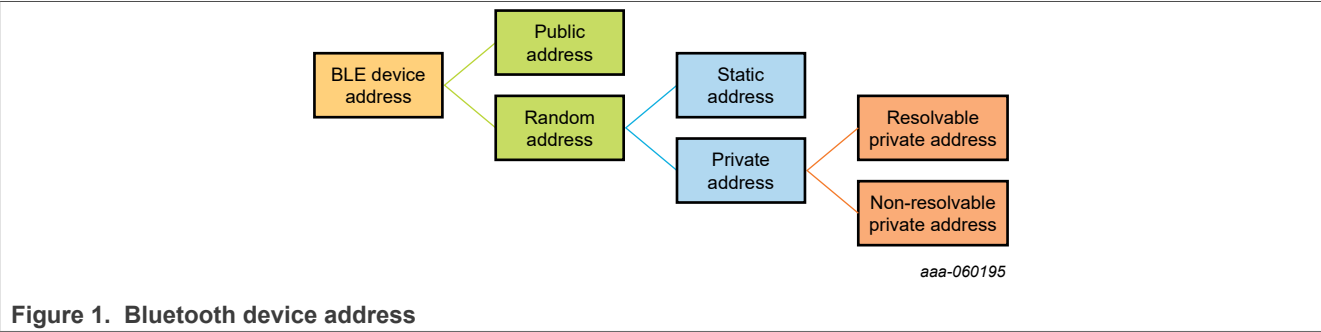
The chips involved in this application note include KW45/KW47/KW38/KW36/QN908x/MCX W71/MCX W72.

The readers of this document are expected to have a basic knowledge of Bluetooth Low Energy.

2 Why do we need random static device address

Bluetooth devices are identified using a device address. The device addresses may be either a public device address or a random device address.

The identity address of a device is a public device address or random static device address that it uses in packets it transmits. If a device is using resolvable private addresses, it also has an Identity Address.



So, for any Bluetooth device, a public device address or random static device address is needed. The SDK uses a public device address by default, and has a fixed OUI. If you use this OUI, there may be a duplicate address for different devices. To keep using the public device address, purchase the address from Bluetooth SIG, and you will have your own OUI.

```
#define BD_ADDR_OUI 0x37U, 0x60U, 0x00U
```

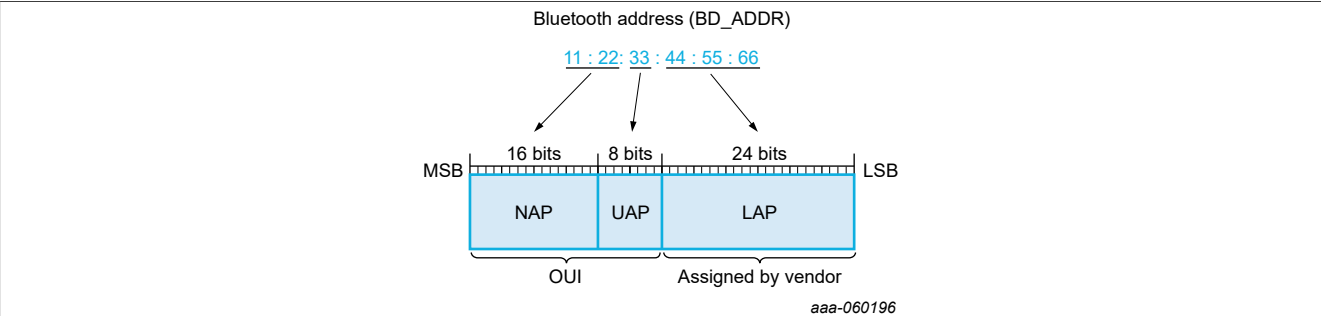


Figure 2. Bluetooth device address OUI

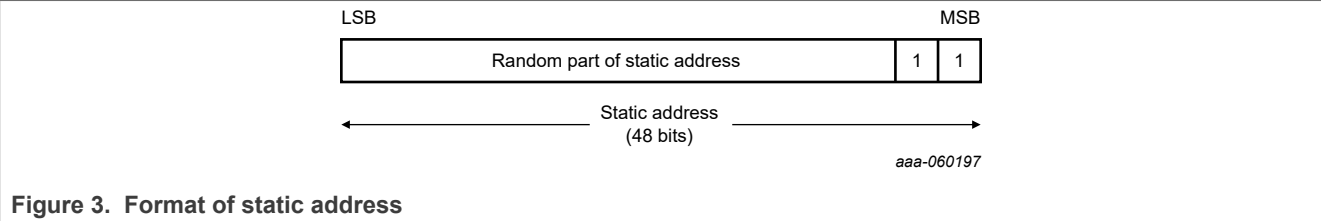
Another option is to use a random static address. It is free and easy to configure.

A static address is a 48-bit randomly generated address and meet the following requirements:

- The two most significant bits of the address are equal to 1.
- At least one bit of the random part of the address is 0.
- At least one bit of the random part of the address is 1.

The format of a static address is shown in [Figure 3](#).

How to Use Random Static Device Address for Bluetooth Application



3 How to set random static address

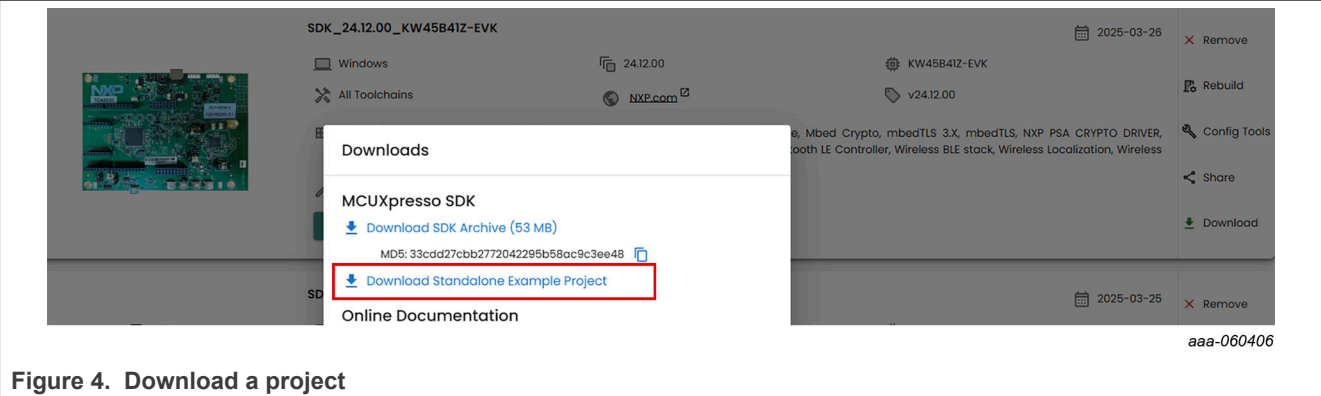
The wireless MCUs of NXP use the same SDK architecture. Below demonstrates how to enable Random Static Address with KW45.

SDK: SDK_24_12_00_KW45B41Z-EVK

Project: wireless_uart_freertos

To set a random static address, perform the following steps:

- 1. Download a project from MCUXpresso.



- 2. Modify the address type.

```
} gapAdvertisingParameters_t gAdvParams = {
    /* minInterval */      gGapAdvertisingIntervalRangeMinimum_c,
    /* maxInterval */      gGapAdvertisingIntervalRangeMinimum_c,
    /* advertisingType */   gAdvConnectableUndirected_c,
    /* addressType */        gBleAddrTypeRandom_c,
    /* directedAddressType */ gBleAddrTypeRandom_c,
    /* directedAddress */    {0, 0, 0, 0, 0, 0},
    /* channelMap */         (gapAdvertisingChannelMapFlags_t) (gGapAdvertisingChannelMapDefault_c),
    /* filterPolicy */        gProcessAll_c
};
```

Figure 5. Modify the address type

How to Use Random Static Device Address for Bluetooth Application

```

gapScanningParameters_t gScanParams =
{
    /* type */          gScanTypeActive_c,
    /* interval */      gGapScanIntervalDefault_d,
    /* window */        gGapScanWindowDefault_d,
    /* ownAddressType */ gBleAddrTypeRandom_c,
    /* filterPolicy */   gScanAll_c,
    /* scanning PHY */   gLePhy1MFlag_c
};

/* Default Connection Request Parameters */
gapConnectionRequestParameters_t gConnReqParams =
{
    .scanInterval = 36,
    .scanWindow = 18,
    .filterPolicy = gUseDeviceAddress_c,
    .ownAddressType = gBleAddrTypeRandom_c,
    .connIntervalMin = 16,
    .connIntervalMax = 16,
    .connLatency = 0,
    .supervisionTimeout = 0x0C80,
    .connEventLengthMin = 0,

```

aaa-060408

Figure 6. Modify the address type

3. Enable bonding and pairing

```

/*! Enable/disable use of bonding capability */
#define gAppUseBonding_d          1

/*! Enable/disable use of pairing procedure */
#define gAppUsePairing_d          1

```

Figure 7. Enable bonding and pairing

4. Save the MAC address to the main flash. To generate a new address, flash erasing is required.

```

#ifdef gHwParamsProdDataPlacement_c
#define gHwParamsProdDataPlacement_c gHwParamsProdDataMainFlashMode_c
// #define gHwParamsProdDataPlacement_c gHwParamsProdDataMainFlash2IfnMode_c
// #define gHwParamsProdDataPlacement_c gHwParamsProdDataIfnMode_c
#endif

```

aaa-060409

Figure 8. Erase memory

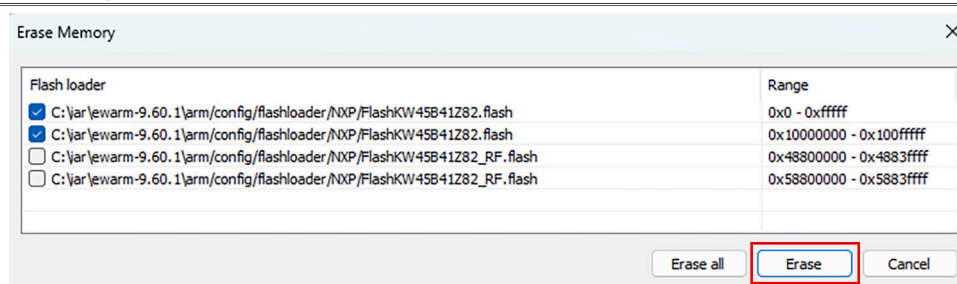


Figure 9. Erase memory

5. Do not set BDAAddr.

How to Use Random Static Device Address for Bluetooth Application

```

bleResult_t Ble_Initialize
(
    gapGenericCallback_t gapGenericCallback
)
{
    #if defined(gUseHciTransportDownward_d) && gUseHciTransportDownward_d
        /* HCI Transport Init */
        if (gHciSuccess_c != HciInit(Ble_HciRecvFromIsr))
        {
            /*
             * Set BD Address in Controller. Must be done after HCI init
             * and before Host init.
             */
            // Ble_SetBDAddr();

            /* Check for available memory storage */
            if (!Ble_CheckMemoryStorage())
        }
    #endif
}

```

aaa-060411

Figure 10. BDAddr

6. Generate a random static address.

```

#define PLATFORM_BLE_BD_ADDR_RAND_PART_SIZE BLE_MAC_ADDR_SZ
#define PLATFORM_BLE_BD_ADDR_OUI_PART_SIZE MAC_ADDR_OUI_PART_SIZE
#define PLATFORM_BLE_BD_ADDR_FULL_SIZE BLE_MAC_ADDR_SZ

```

aaa-060412

Figure 11. Generate a random static address

```

static void PLATFORM_GenerateNewBDAddr(uint8_t *bleDeviceAddress)
{
    uint8_t macAddr[PLATFORM_BLE_BD_ADDR_RAND_PART_SIZE] = {0U};

    #if (gPlatformUseUniqueDeviceIdForBdAddr_d != 0)
    {
        int ret;

        ret = RNG_Init();
        assert(ret == 0);
        (void)ret;

        #ifndef FWK_RNG_DEPRECATED_API
            assert(num == PLATFORM_BLE_BD_ADDR_RAND_PART_SIZE);
            (void)num;
        #endif

        /* Set random static address */
        macAddr[PLATFORM_BLE_BD_ADDR_RAND_PART_SIZE-1] |= 0xC0;
        FLib_MemCpy((void *)bleDeviceAddress, (const void *)macAddr, PLATFORM_BLE_BD_ADDR_RAND_PART_SIZE);

        /* Set 3 MSB from OUI */
        // FLib_MemCpy((void *)&bleDeviceAddress[PLATFORM_BLE_BD_ADDR_RAND_PART_SIZE], (const void *)gBD_ADDR_OUI_c,
        //             PLATFORM_BLE_BD_ADDR_OUI_PART_SIZE);
    }
}

```

aaa-060413

Figure 12. Generate a random static address

7. Set the random static address.

```

void BleConnManager_GapCommonConfig(void)
{
    /* Read public address from controller */
    // (void)Gap_ReadPublicDeviceAddress();

    #include "fwk_platform_ble.h"
    uint8_t bleDeviceAddress[gBleDeviceAddressSize_c] = {0};
    PLATFORM_GetBDAddr(bleDeviceAddress);
    Gap_SetRandomAddress(bleDeviceAddress);
    gSmpKeys.addressType = gBleAddrTypeRandom_c;
    gSmpKeys.aAddress = bleDeviceAddress;

    #if (defined(gAppUsePairing_d) && (gAppUsePairing_d == 1U))
        /* Register security requirements if pairing is used */
        (void)Gap_RegisterDeviceSecurityRequirements(&deviceSecurityRequirements);
    #endif
}

```

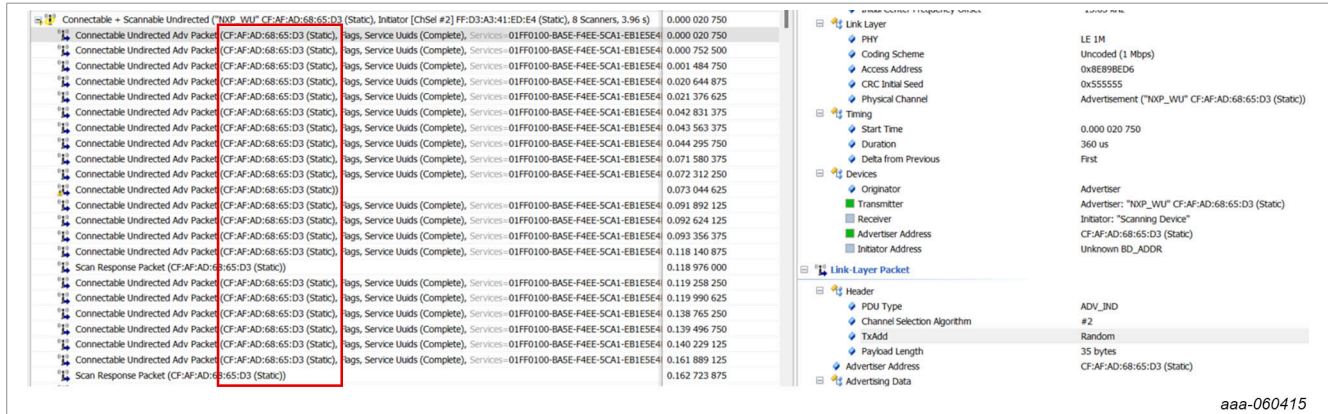
aaa-060414

Figure 13. Set a random static address

8. If a resolvable private address is needed for advertising, set the macro gAppUsePrivacy_d to 1.

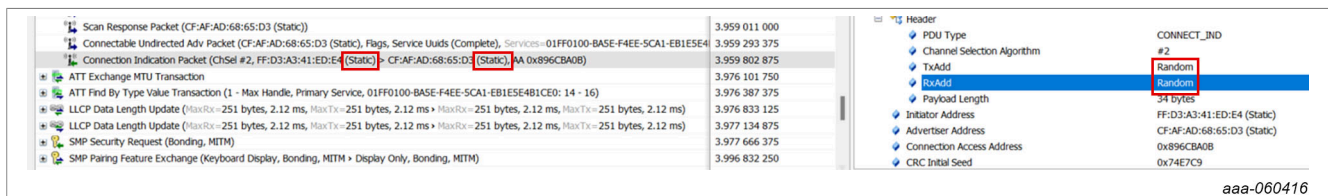
4 Check the result

Using the sniffer to capture the air log, we can see that the address type is **Random Static**.



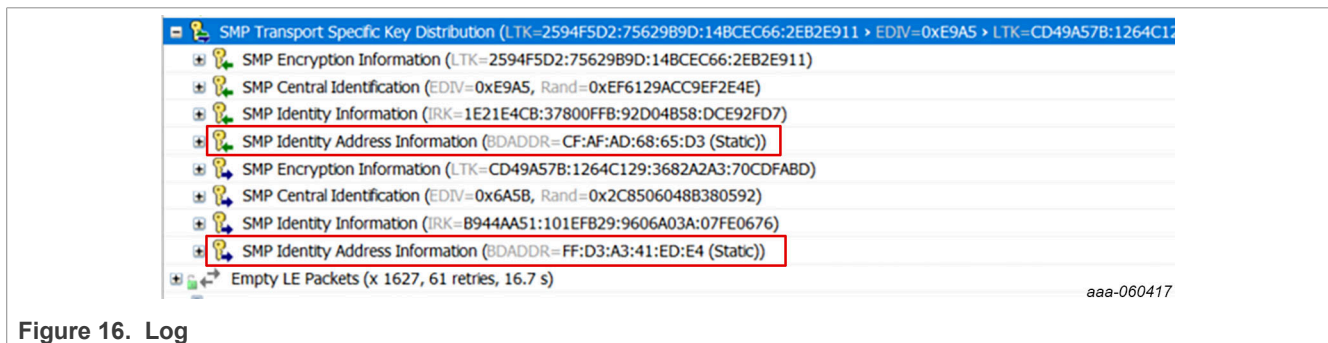
aaa-060415

Figure 14. Log



aaa-060416

Figure 15. Log



aaa-060417

Figure 16. Log

5 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 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.

6 Revision history

[Table 1](#) summarizes the revisions to this document.

Table 1. Revision history

Document ID	Release date	Description
AN14645 v1.0	16 April 2025	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.

How to Use Random Static Device Address for Bluetooth Application

Bluetooth — the Bluetooth wordmark and logos are registered trademarks owned by Bluetooth SIG, Inc. and any use of such marks by NXP Semiconductors is under license.

Microsoft, Azure, and ThreadX — are trademarks of the Microsoft group of companies.

Contents

1 Introduction 2

2 Why do we need random static device address 2

3 How to set random static address 3

4 Check the result 6

5 Note about the source code in the document 7

6 Revision history 7

Legal information 8

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