AN14796

Migration Guide from the KW45 to the KW47

Rev. 1.0 — 17 April 2026 | Application note

Document information

Information	Content
Keywords	AN14796, KW47, migration guide, KW45, NBU, ISP, SDK, SPSDK
Abstract	This document describes the procedure to migrate from KW45B41Z to KW47 with emphasis on the connectivity software.

1 Introduction

This document describes the procedure to migrate from KW45B41Z to KW47 with emphasis on the connectivity software. The document is intended for software engineers, software testers, software integrators, and customers designing their own hardware.

2 Comparison

Table 1 shows some of the principal similarities and differences between the two wireless MCUs.

Table 1. Differences and common features - KW45 and KW47

Features	KW45	KW47
Application core	Arm Cortex-M33 core up to 96 MHz	Arm Cortex-M33 core up to 96 MHz
Bluetooth Low Energy (LE) radio core	• Dedicated CM3 core running up to 64 MHz	• Dedicated **CM33 core** running up to 64 MHz
EdgeLock Secure Enclave core	Isolated Arm Cortex-M0+ core	Isolated Arm Cortex-M0+ core
Software	FreeRTOS Bluetooth Low Energy 5.3 EdgeLock Secure Enclave	FreeRTOS **Bluetooth LE 6.0** EdgeLock Secure Enclave **Bluetooth Channel Sounding**
Package	• 6 x 6 mm 40HVQFN with “Wettable” flanks • 7 x 7 mm 48HVQFN with “Wettable” flanks (pin compatible with KW47)	7 x 7 mm 48HVQFN with “Wettable” flanks (pin compatible with KW45)
Application memory (Flash/SRAM)	8 kB code cache 1 MB program flash 128 kB SRAM	**16 kB code cache** **2 MB flash memory** **256 kB SRAM**
Radio subsystem memory (Flash/RAM)	256 kB Radio flash 88 kB Radio RAM	512 kB Flash 171 kB SRAM
Radio core)	2.4 GHz Bluetooth LE radio with upgradable core version 5.3, up to 24 simultaneous connections (125 kbit/s, 500 kbit/s, 1 Mbit/s, and 2 Mbit/s) in any central/peripheral combination. • Modulation Types: Frequency-shift keying (FSK), gaussian frequency-shift keying (GFSK), minimum-shift keying (MSK), gaussian minimum-shift keying (GMSK), offset quadrature phase-shift keying (OQPSK)	2.4 GHz **Bluetooth LE 6.0 radio core**, up to 24 simultaneous connections (125 kbit/s, 500 kbit/s, 1 Mbit/s, and 2 Mbit/s) in any central/peripheral combination. • Modulation types: FSK, GFSK, MSK, GMSK, OQPSK
Radio TX power	• Programmable transmit output power up to +10 dBm • On-chip balun with single-ended bidirectional RF port	
Radio power (Rx/Tx@0 dBm)	Typical TX at 0 dBm: 4.6 mA	
Radio sensitivity	Bluetooth LE radio core • 106 dBm 125 kbit/s long range receive sensitivity • 102 dBm 500 kbit/s long range receive sensitivity • 97.5 dBm 1 Mbit/s receive sensitivity	

Table 1. Differences and common features - KW45 and KW47_...continued_

Features	KW45	KW47
	• 95 dBm 2 Mbit/s receiver sensitivity	
Communication interfaces	1x FlexCAN w/CAN FD support 2x LPUART w/LIN 2x LPSPI 1x I3C 2x LPI2C supporting SMBus 1x FlexIO	**2x FlexCAN w/CAN FD support** 2x LPUART w/LIN 2x LPSPI 1x I3C 2x LPI2C supporting SMBus 1x FlexIO
Clocks	32 MHz RF OSC 32.768 kHz OSC 192 MHz FRO 6 MHz FRO 32 kHz FRO	32 MHz RF OSC 32.768 kHz OSC 192 MHz FRO 32 kHz FRO
Security	EdgeLock Secure Enclave, Core Profile AES-128, AES-192, AES-255, supporting ECB, CBC, CTR AEAD (AES GCM, AES CCM) SHA-224, SHA-256, SHA-384, SHA-512 CMAC-AES, HMAC ECDSA P-224, P-256, P-384, P-521, X25519 TRNG with DRBG-CTR, Key generation via DRBG Key Derivation: LTK and Session Key, ECDH(E) (P-224, P-256, P-384, P-521, Ed25519) SESIP L2/PSA L2/RED, Tamper Protection, Trust Provisioning.	EdgeLock Secure Enclave, Core Profile AES-128, AES-192, AES-256 – supporting ECB, CBC, CTR, AEAD (AES GCM, AES CCM) **SHA-1**, SHA-224, SHA-256, SHA-384, SHA-512, **SHA3** **Multipart hashing, 4 operations in parallel, SHA2xx, SHA3 context import/export, and context clone** CMAC-AES, HMAC, **Multipart MAC, 4 operations in parallel** ECDSA P-224, P-256, P-384, P-521, **Brainpool**, X25519 TRNG with DRBG-CTR, Key generation via DRBG Key Derivation: LTK and Session Key, **CKDF (CTR), HKDF, SPAKE2+ (Matter)**, ECDH(E) (P-224, P256, P-384, P-521, Ed25519, **Brainpool**), **NBU Advance Key Protection**, SESIP L2/PSA L2/RED, Tamper Protection, Trust Provisioning, **ISO 21434**.
Automative qualification	AEC-Q100 Grade 2	

2.1 Peripherals instantiation

The KW45 devices in 48-pin HVQFN packages are pin-to-pin compatible with the KW47 devices. The KW45 devices are also available in 40-pin “wettable” HVQFN packages.

The main difference between the KW45 and KW47 is that the KW47 supports more pins with wake-up capabilities and two FlexCAN instances.

2.1.1 Pinout diagrams

This section illustrates the following pinout diagrams:

- 40-pin HVQFN package - KW45
- 48-pin HVQFN package - KW45 and KW47

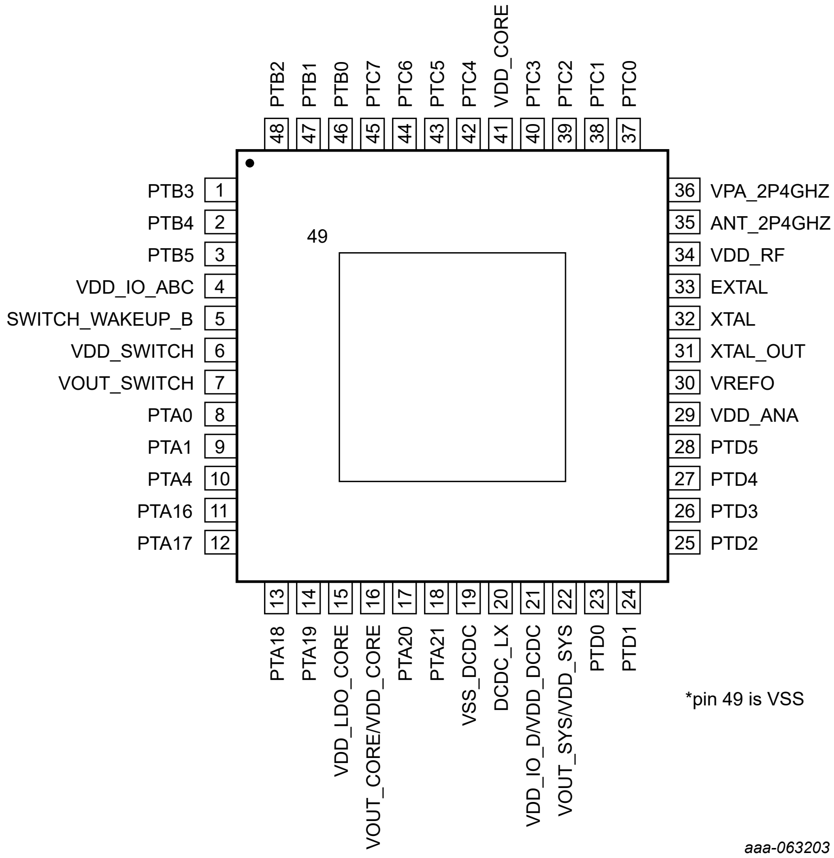


Figure 1. 48-pin HVQFN package pinout diagram (KW47 and KW45)

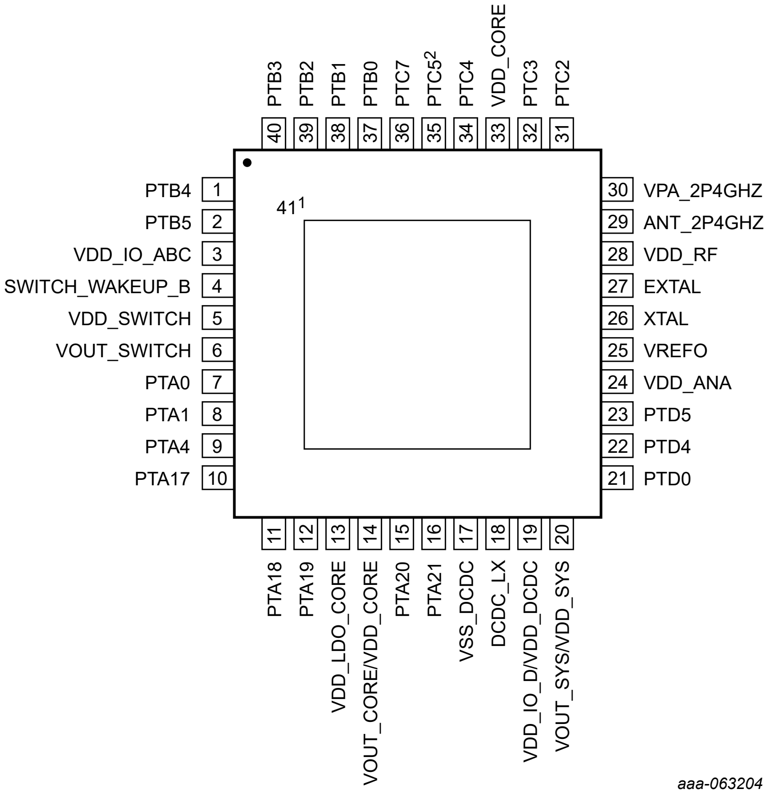


Figure 2. 40-pin HVQFN package pinout diagram (KW45)

The **bold** alternatives are available only for the KW47 devices.

Table 2. Pin mux assignment for KW45 and KW47

KW45 QFN 40	KW45-KW47 QFN48	Pin Name	Pin mux Assignment	
	1	PTB3	ALT0 - ADC0_B13 ALT1 - PTB3 ALT2 - LPSPI1_SOUT ALT3 - LPUART1_RX	ALT5 - TPM1_CH3 ALT9 - FLEXIO0_D29 WUU0_P14
1	2	PTB4	ALT1 - PTB4 ALT2 - LPSPI1_PCS3 ALT3 - LPUART1_CTS_b ALT4 - LPI2C1_SDA ALT5 - I3C0_SDA	ALT6 - TRGMUX0_IN0 ALT9 - FLEXIO0_D30 WUU0_P15
2	3	PTB5	ALT1 - PTB5 ALT2 - LPSPI1_PCS2 ALT3 - LPUART1_RTS_b ALT4 - LPI2C1_SCL	ALT5 - I3C0_SCL ALT6 - TRGMUX0_OUT0 ALT9 - FLEXIO0_D31
3	4	VDD_IO_ABC	ALT0 - VDD_IO_ABC	
4	5	SWITCH_ WAKEUP_B	ALT0 - SWITCH_WAKEUP_B	
5	6	VDD_SWITCH	ALT0 - VDD_SWITCH	

Table 2. Pin mux assignment for KW45 and KW47...*continued*

KW45 QFN 40	KW45-KW47 QFN48	Pin Name	Pin mux Assignment	
6	7	VOUT_SWITCH	ALT0 - VOUT_SWITCH	
7	8	PTA0	ALT1 - PTA0 ALT2 - CMP0_OUT ALT3 - LPUART0_CTS_b ALT4 - RF_GPO_11	ALT5 - TPM0_CH4 ALT6 - FLEXIO0_D0 ALT7 - SWD_DIO WUU0_P0
8	9	PTA1	ALT1 - PTA1 ALT2 - CMP1_OUT ALT3 - LPUART0_RTS_b ALT4 - RF_GPO_10	ALT5 - TPM0_CH5 ALT6 - FLEXIO0_D1 ALT7 - SWD_CLK/TP_HCLK
9	10	PTA4	ALT0 - ADC0_A10/CMP0_IN0/ATX0 ALT1 - PTA4 ALT3 - RF_GPO_9 ALT4 - TPM0_CLKIN	ALT5 - TRACE_SWO ALT6 - FLEXIO0_D4 ALT7 - BOOT_CONFIG WUU0_P2/RF_XTAL_OUT_ENABLE
	11	PTA16	ALT0 - ADC0_A12/NVM_TM2 ALT1 - PTA16 ALT2 - LPSPI0_PCS0 ALT3 - EWM0_OUT_b ALT4 - LPI2C0_SCLS ALT5 - TPM0_CH4	ALT6 - LPUART0_RX ALT7 - RF_GPO_8 ALT9 - FLEXIO0_D5 **WUU0_P19**/RF_NOT_ALLOWED
10	12	PTA17	ALT0 - ADC0_A13/NVM_TM3 ALT1 - PTA17 ALT2 - LPSPI0_SIN ALT3 - EWM0_IN ALT4 - LPI2C0_SDAS ALT5 - TPM0_CH5	ALT6 - LPUART0_TX ALT7 - RF_GPO_7 ALT8 - RF_GPO_8 ALT9 - FLEXIO0_D6 ALT11 - RF_EXT_XTAL_REQUEST/ RF_GPO_7 WUU0_P3/RF_NOT_ALLOWED
11	13	PTA18	ALT0 - CMP1_IN1 ALT1 - PTA18 ALT2 - LPSPI0_SOUT ALT3 - LPUART0_CTS_b ALT4 - LPI2C0_SDA	ALT5 - TPM0_CH3 ALT6 - RF_GPO_0 ALT10 - LPUART0_RX ALT11 - SPC0_LPREQ **VDD SYS - WUU0_P20**
12	14	PTA19	ALT0 - CMP1_IN0/ATX1 ALT1 - PTA19 ALT2 - LPSPI0_SCK ALT3 - LPUART0_RTS_b ALT4 - LPI2C0_SCL	ALT5 - TPM0_CH2 ALT6 - RF_GPO_1 WUU0_P4
13	15	VDD_LDO_ CORE	ALT0 - VDD_LDO_CORE	
14	16	VOUT_CORE/ VDD_CORE	ALT0 - VOUT_CORE	
15	17	PTA20	ALT0 - ADC0_A14/CMP0_IN3 ALT1 - PTA20 ALT2 - LPSPI0_PCS2 ALT3 - LPUART0_TX	ALT5 - TPM0_CH1 ALT6 - RF_GPO_2 ALT8 - FLEXIO0_D7

Table 2. Pin mux assignment for KW45 and KW47 *...continued*

KW45 QFN 40	KW45-KW47 QFN48	Pin Name	Pin mux Assignment	
			ALT4 - EWM0_IN	**ALT11 - LPUART0_RTS_b**
16	18	PTA21	ALT0 - ADC0_A15/CMP0_IN2 ALT1 - PTA21 ALT2 - LPSPI0_PCS3 ALT3 - LPUART0_RX ALT4 - EWM0_OUT_b ALT5 - TPM0_CH0	ALT6 - RF_GPO_3 ALT7 - RF_GPO_7 ALT8 - FLEXIO0_D8 ALT9 - RF_GPO_10 **ALT11 - LPUART0_CTS_b** WUU0_P5
17	19	VSS_DCDC	ALT0 - VSS_DCDC	
18	20	DCDC_LX	ALT0 - DCDC_LX	
19	21	VDD_IO_D/ VDD_DCDC	ALT0 - VDD_DCDC	
20	22	VOUT_SYS/ VDD_SYS	ALT0 - VDD_SYS	
21	23	PTD0	ALT0 - ADC0_A5 ALT1 - PTD0 ALT3 - RESET_b	
	24	PTD1	ALT0 - ADC0_B5 ALT1 - PTD1 ALT2 - SPC0_LPREQ ALT3 - NMI_b	ALT4 - RF_GPO_4 **ALT5 - RF_GPO_7** **ALT7 - LPTMR2_TRIG_OUT_b**
	25	PTD2	ALT0 - ADC0_A6 ALT1 - PTD2 ALT2 - LPTMR0_ALT3 ALT3 - TAMPER0	ALT4 - RF_GPO_5 **ALT5 - TPM2_CH0** **ALT7 - LPTMR1_TRIG_OUT_b**
	26	PTD3	ALT0 - ADC0_B6 ALT1 - PTD3 ALT2 - LPTMR1_ALT3 ALT3 - TAMPER1 ALT4 - RF_GPO_6	**ALT5 - TPM2_CH1** ALT6 - TRGMUX0_IN2 **ALT7 - LPTMR0_TRIG_OUT_b**
22	27	PTD4	ALT0 - XTAL32K ALT1 - PTD4 ALT2 - LPTMR0_ALT2	ALT3 - TAMPER2
23	28	PTD5	ALT0 - EXTAL32K ALT1 - PTD5 ALT2 - LPTMR1_ALT2	
24	29	VDD_ANA	ALT0 - VDD_ANA	
25	30	VREFO/VREF_ OUT	ALT0 - VREFO	
	31	XTAL_OUT	ALT0 - XTAL_OUT	
26	32	XTAL	ALT0 - XTAL	
27	33	EXTAL	ALT0 - EXTAL	

Table 2. Pin mux assignment for KW45 and KW47 *...continued*

KW45 QFN 40	KW45-KW47 QFN48	Pin Name	Pin mux Assignment	
28	34	VDD_RF	ALT0 - VDD_RF	
29	35	ANT_2P4GHZ	ALT0 - ANT_2P4GHZ	
30	36	VPA_2P4GHZ	ALT0 - VPA_2P4GHZ	
	37	PTC0	ALT1 - PTC0 ALT2 - LPSPI1_PCS2 ALT3 - CAN0_TX 1 ALT4 - I3C0_SDA ALT5 - TPM1_CH0	ALT7 - LPI2C1_SCL ALT9 - FLEXIO0_D16 WUU0_P7
	38	PTC1	ALT1 - PTC1 ALT2 - LPSPI1_PCS3 ALT3 - CAN0_RX 1 ALT4 - I3C0_SCL ALT5 - TPM1_CH1	ALT7 - LPI2C1_SDA ALT9 - FLEXIO0_D17 WUU0_P8
31	39	PTC2	ALT1 - PTC2 ALT2 - LPSPI1_SOUT ALT3 - LPUART1_RX ALT4 - LPI2C1_SCLS ALT5 - TPM1_CH2	ALT7 - I3C0_PUR ALT9 - FLEXIO0_D18 **ALT11 - CAN1_RX** WUU0_P9
32	40	PTC3	ALT1 - PTC3 ALT2 - LPSPI1_SCK ALT3 - LPUART1_TX ALT4 - LPI2C1_SDAS	ALT5 - TPM1_CH3 ALT9 - FLEXIO0_D19 **ALT11 - CAN1_TX 1**
33	41	VDD_CORE	ALT0 - VDD_CORE	
34	42	PTC4	ALT1 - PTC4 ALT2 - LPSPI1_SIN ALT3 - CAN0_TX 1 ALT4 - LPI2C1_SCL	ALT6 - TPM2_CH0 ALT9 - FLEXIO0_D20 WUU0_P10
35	43	PTC5	ALT1 - PTC5 ALT2 - LPSPI1_PCS0 ALT3 - CAN0_RX 1 ALT4 - LPI2C1_SDA	ALT5 - TPM1_CH4 ALT6 - TPM2_CH1 ALT9 - FLEXIO0_D21 **VDD SYS - WUU0_P22**
	44	PTC6	ALT0 - ADC0_A8 ALT1 - PTC6 ALT2 - LPSPI1_PCS1 ALT5 - TPM1_CH5	ALT9 - FLEXIO0_D22 **ALT11 - CAN1_RX** WUU0_P11
36	45	PTC7	ALT0 - DISABLED ALT1 - PTC7 ALT2 - TRGMUX0_IN3 ALT3 - TRGMUX0_OUT3 ALT4 - SFA0_CLK ALT5 - TPM1_CLKIN	ALT6 - TPM2_CLKIN ALT7 - CLKOUT ALT9 - FLEXIO0_D23 **ALT10 - NMI_b** **ALT11 - CAN1_TX** WUU0_P12/NMI_b/RF_NOT_ALLOWED
37	46	PTB0	ALT0 - ADC0_B10	ALT5 - TPM1_CH0

Table 2. Pin mux assignment for KW45 and KW47*...continued*

KW45 QFN 40	KW45-KW47 QFN48	Pin Name	Pin mux Assignment	
			ALT1 - PTB0 ALT2 - LPSPI1_PCS0 **ALT3 - CAN1_RX 1**	ALT9 - FLEXIO0_D26 WUU0_P13
38	47	PTB1	ALT0 - ADC0_B11 ALT1 - PTB1 ALT2 - LPSPI1_SIN **ALT3 - CAN1_TX 1**	ALT5 - TPM1_CH1 ALT9 - FLEXIO0_D27
39	48	PTB2	ALT0 - ADC0_B12 ALT1 - PTB2 ALT2 - LPSPI1_SCK ALT3 - LPUART1_TX	ALT5 - TPM1_CH2 ALT9 - FLEXIO0_D28
41	49	VSS	ALT0 - VSS	

For more details, see the Chapter “Signal Muxing and Pinout” in the *KW47 Reference Manual* (KW47RM).

3 Memory

This section describes the memory comparison between the KW45 and KW47.

3.1 System memory map

System memory map comparison between the KW45 and KW47. The differences are highlighted in **bold**.

Table 3. System memory map comparison

Description	KW45		KW47		Secure/Non-Secure	Cached
	Address	Size	System 32-bit Address range	Size		
Program Flash	0x0000_0000	1024 kB	0x0000_0000	**2 MB**	Non-Secure	Code Cache
Reserved	0x0010_0000	-	**0x0020_0000**	-	-	-
Flash Logical Window	0x0100_0000	1024 kB	**0X0100_0000**	**2 MB**	Non-Secure	Code Cache
Reserved	0x0110_0000	-	**0x0120_0000**	-	-	-
IFR0	0x0200_0000	32 kB	0x0200_0000	32 kB	Non-Secure	No
Reserved	0x0200_8000	-	0x0200_8000	-	-	-
IFR1	0x0210_0000	8 kB	0x0210_0000	8 kB	Non-Secure	No
Reserved	0x0210_2000	-	0x0210_2000	-	-	-
Tightly Coupled Memory - Code	0x400_0000	16 kB	0x400_0000	**32 kB**	Non-Secure	No
Reserved	0x0400_4000	-	**0x0400_8000**	-	-	-
Read-Only Memory - Boot	0x0480_0000	96 kB	0x0480_0000	96 kB	Non-Secure	No
Reserved	0x0481_8000	-	0x0481_8000	-	-	-

Table 3. System memory map comparison*...continued*

Description	KW45		KW47		Secure/Non-Secure	Cached
	Address	Size	System 32-bit Address range	Size		
Program Flash	0x1000_0000	1024 kB	0x1000_0000	**2 MB**	Secure	Code Cache
Reserved	0x1010_0000	-	**0x1020_0000**	-	-	-
Flash Logical Window	0x1100_0000	1024 kB	0x1100_0000	**2 MB**	Secure	Code Cache
Reserved	0x1110_0000	-	0x1120_0000	-	-	-
IFR0	0x1200_0000	32 kB	0x1200_0000	32 kB	Secure	No
Reserved	0x1200_8000	-	0x1200_8000	-	-	-
IFR1	0x1210_0000	8 kB	0x1210_0000	8 kB	Secure	No
Reserved	0x1210_2000	-	0x1210_2000	-	-	-
Tightly Coupled Memory - Code	0x1400_0000	16 kB	0x1400_0000	**32 kB**	Secure	No
Reserved	0x1400_4000	-	**0x1400_8000**	-	-	-
Read-Only Memory - Boot	0x1480_000	96 kB	0x1480_0000	96 kB	Secure	No
Reserved	0x1480_0000	-	0x1481_8000	-	-	-
Tightly Coupled Memory - System	0x2000_0000	112 kB	0x2000_0000	**232 kB**	Non-Secure	No
Reserved	0x2001_C000	-	**0x2003_A000**	-	-	-
Tightly Coupled Memory - System	0x3000_0000	112 kB	0x3000_0000	**232 kB**	Secure	No
Reserved	0x3001_C000	-	**0x3003_A000**	-	-	-
KW45 Peripheral Bridge 2	0x4000_0000	512 kB	0x4000_0000	512 kB	Non-Secure	No
KW47 Peripheral Bridge 0						No
Reserved	0x4008_0000	-	0x4008_0000	-	-	-
Fast peripherals 0	0x4800_0000	8 MB	0x4800_0000	8 MB	Non-Secure	No
Fast peripherals 1	0x4880_0000	8 MB	0x4880_0000	8 MB	Non-Secure	No
Reserved	0x4900_0000		0x4900_0000			
KW45 Peripheral Bridge 2	0x5000_0000	512 kB	0x5000_0000	512 kB	Secure	No
KW47 Peripheral Bridge 0						No
Reserved	0x5008_0000	-	0x5008_0000	-	-	-
Fast peripherals 0	0x5800_0000	8 MB	0x5800_0000	8 MB	Secure	No
Fast peripherals 1	0x5880_0000	8 MB	0x5880_0000	8 MB	Secure	No
Reserved	0x5900_0000	-	0x5900_0000	-	-	-

Table 3. System memory map comparison...*continued*

Description	KW45		KW47		Secure/Non-Secure	Cached
	Address	Size	System 32-bit Address range	Size		
Private peripheral Bus Internal	0xE000_0000	256 kB	0xE000_0000	256 kB	Mixed	No
Private peripheral Bus External	0xE004_0000	768 kB	0xE004_0000	768 kB	Mixed	No

For more details, see section "3.4 System memory map" in the *KW47 Reference Manual* (KW47RM).

3.1.1 ROM bootloader

The ROM is 96 kB for both KW45 and KW47 and supports UART, CAN, SPI, and I2C in-system programming (ISP) functions. In these devices, the following alternate speeds through the ROM bootloader feature are:

- Normal boot: KW45 (32 MHz), **KW47 (48 MHz)**.
- Fast boot: KW45 (96 MHz), KW47 (96 MHz).

The ROM behavior is different at different lifecycles. The lifecycle fuse must be programmed correctly. If not, the ROM bootloader does not boot.

3.1.2 ROM bootloader memory usage comparative

Figure 3 and Figure 4 compare the ROM bootloader memory usage of the KW45 and KW47.

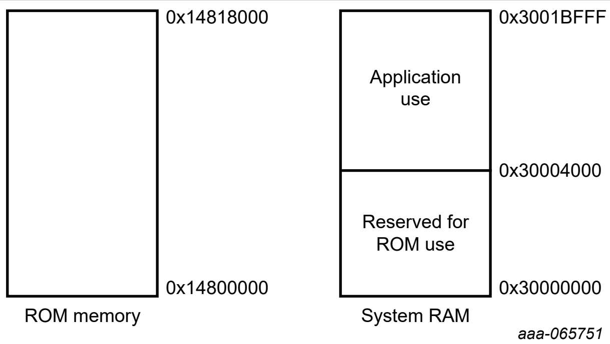


Figure 3. KW45 ROM bootloader memory usage

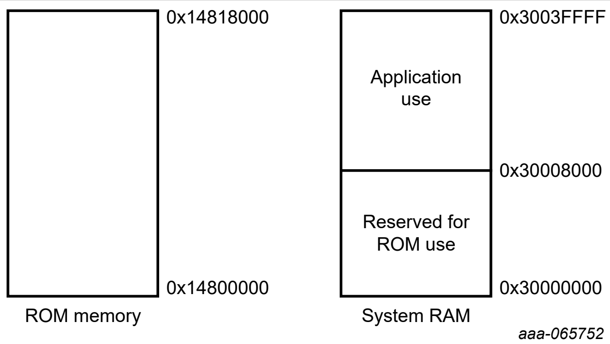


Figure 4. KW47 ROM bootloader memory usage

3.1.3 Lifecycle and fuses

Table 4 describes the Lifecycle state and fuses differences between the KW45 and KW47.

Table 4. Lifecycle - KW45 and KW47

Lifecycle	Lifecycle State	Debug authentication required to debug CM33?		Debug authentication required to debug radio core?	
		KW45	KW47	KW45	KW47
0000_0000	Blank	NA	NA	NA	NA
0000_0111	OEM Open	No	No	Yes	No
0000_1111	OEM Secure World Closed	Yes	Yes	Yes	Yes
0001_1111	OEM Closed	Yes	Yes	Yes	Yes
1001_1111	OEM Locked	Yes	NA	NA	NA
0011_1111	OEM Return	NA	No	Yes	NA
11xx_xxxx	Brick	NA	NA	NA	NA

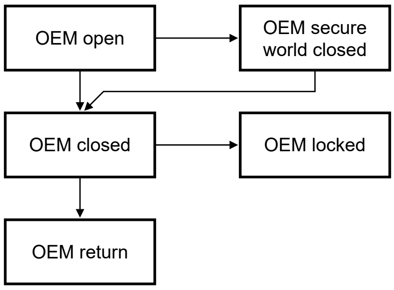


Figure 5. Lifecycle transitions

In addition to the lifecycles fuses, there are a handful of other fuses that the ROM uses. Table 5 shows the minor differences between the KW45 and KW47.

Table 5. Differences between KW45 and KW47

Fuses	KW45	KW47	Description
SECURE_PHANTON_ CONFIG	-	0x18	Token configuration
OEM_Enablement_Token	-	0x21	256-bit root key table hash (RoTKTH) typically used for radio firmware authentication. If a valid token is intended for programming the SECURE_PHANTOM_CONFIG fuse.
GD_EN	0x30	-	0b - Core VDD glitch detection does not generate a hardware reset. 1b - Boot ROM configures/enables Core VDD glitch detection-based reset
HVD_EN	0x2F	-	0b - Core VDD high-voltage event does not generate a hardware reset. 1b - Boot ROM configures/enables Core VDD high-voltage detection-based reset.
NBU_SEC_BOOT_EN	-	0x33	Narrowband unit (NBU) Secure Boot Enable 0b - NBU secure boot is disabled. 1b - NBU secure boot is enabled.
OEM_SEC_BOOT_EN	-	0x35	OEM Secure Boot Enable

Table 5. Differences between KW45 and KW47*...continued*

Fuses	KW45	KW47	Description
			0b - OEM secure boot is disabled. 1b - OEM secure boot is enabled.

For more details, see table “Additional fuse fields of interest” in the “ROM Bootloader” Chapter in the *KW47 Reference Manual* (KW47RM).

3.1.4 ROM bootloader configuration

Sector 0 of user IFR is ROM and ISP configuration, which provide configuration for the ROM bootloader.

The KW47 implements a boot configuration to modify the DC-DC power for a Low-power boot mode, and save the energy with an offset value [0x0050]. The new features for the KW47 are highlighted in **bold**.

- Bit 0 - Enable ISP or not
  - 0 = BOOT_CFG pin disabled
  - 1 = BOOT_CFG pin enabled (default)
- Bit 2:1 - Boot Speed
  - 00 = Normal boot (32 MHz for KW45 and **48 MHz for KW47**)
  - 10 = Fast boot (96 MHz for KW45 and 96 MHz for KW47)
  - 01,11 = Normal boot (default)
- **Bit 3 - DC-DC boot power mode**
  - **0 = Lower power boot mode**
  - **1 = Normal power mode**

The bit 0 of the boot configuration field is used to enable ISP path for the ROM bootloader. If this bit is set, the BOOT_CFG pin is enabled. The pull-down on the BOOT_CFG pin is to ensure that the device does not enter the ISP by default. When the BOOTCFG pin is enabled and the BOOT_CFG pin is not pulled down, the ROM bootloader enters the ISP path.

To decide on how to configure the power and clock, the ROM bootloader loads the boot speed. The bit 1:2 of the boot configuration indicates the boot speed. The new features for the KW47 are highlighted in **bold**.

Table 6. Lifecycle - KW45 and KW47

Boot speed value in IFR0	Clock source	CPU_CLK/BUS_CLK/SLOW_CLK	DC-DC and CORE_LDO Voltage
11/00/01: Normal Boot (Default)	FRO_192M	48/48/24 MHz	• CORE_LDO: 1.0 V • **DC-DC: 1.8 V or 1.25 V**
10: Fast Boot	FRO_192M	96/96/24 MHz	• CORE_LDO: 1.1 V • **DC-DC: 1.8 V or 1.35 V**

Also, the KW47 offers new fields for ISP mode in offset [0x140]:

- Go ISP mode options: If you enter the ISP mode, the log is recorded on the corresponding pin when this bit and go ISP mode fail alert pin are enabled.
  - Bit 0 - Reserved
  - Bit 1 - Secured boot option
    - 0 = Enable

  - 1 = 1 Disable (default)
- Go ISP mode fail alert pin: Set the ISP log pin.
  - Bit [4:0] - Pin selection
    - 0x0 ~ 0x1F = Pin0 ~ Pin31
  - Bit [7:5] - Port selection
    - 0x0 ~ 0x4 = PortA ~ PortE
    - 0x5 ~ 7 - Reserved

For more details, see “Table 62 ROM Configuration fields” and “Table 63 for Boot speed at each DC-DC configuration” in the *KW47 Reference Manual* (KW47RM).

3.2 ISP assignments

The ROM bootloader provides an in-system programming (ISP) utility that operates over a serial connection on the MCUs. It enables quick and easy programming of MCUs through the entire product lifecycle, including application development, final product manufacturing, and beyond.

In KW45 and KW47, the ROM bootloader supports automated booting from internal flash and downloading image from serial interface (LPUART, LPI2C, LPSPI, and CAN). The UART interface is up to 4 Mbit/s.

3.2.1 ISP path

To make the ROM bootloader enter the ISP path, press the BOOT_CONFIG pin PTA4. The BOOT_CONFIG pin is enabled by default in the information flash region (IFR0).

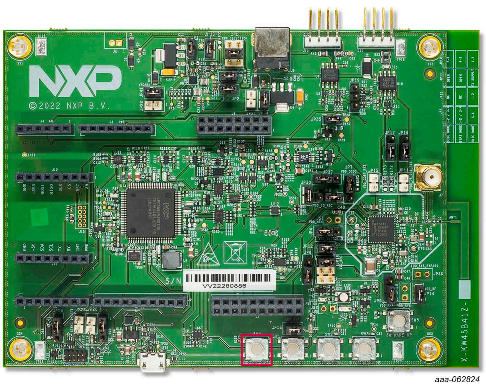


Figure 6. KW45B41Z-EVK

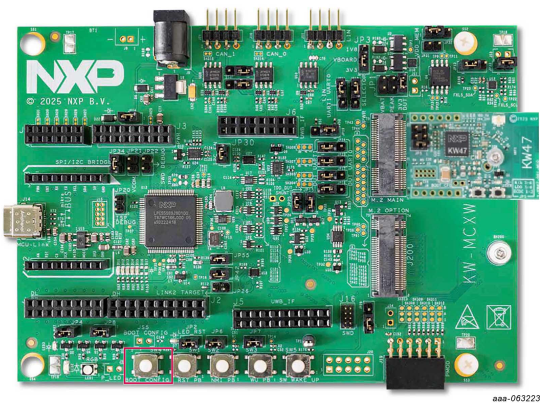


Figure 7. KW47-EVK

Table 7 shows the peripheral instances and pin assignments that the ISP uses.

Table 7. Peripheral instances and pin assignments

Peripheral instances	Pin name	Pin assignment	ALT
LPUART1	LPUART1_TX	PTC3	3
	LPUART1_RX	PTC2	3
LPSPI1	LPSPI1_PCS0	PTB0	2
	LPSPI1_SIN	PTB1	2
	LPSPI1_SCK	PTB2	2
	LPSPI1_SOUT	PTB3	2
CAN0	CAN0_TX	PTC4	3
	CAN0_RX	PTC5	3
LPI2C1	LPI2C1_SCL	PTB5	4
	LPI2C1_SDA	PTB4	4
Boot pin	BOOT_CONFIG	PTA4	Default

For more details on configuring the BOOT_CONFIG pin, see the Section 3.1.4. The ROM bootloader enters the ISP path when no boot image is found, PC and SP are not valid, and some other boot failure conditions exist.

For more details on the ISP path, commands, and usage, see section “15.2.7 ISP path” in the *KW47 Reference Manual* (KW47RM).

4 Hardware considerations for development boards

This section summarizes the main differences and considerations of using the development boards for the KW45 and KW47.

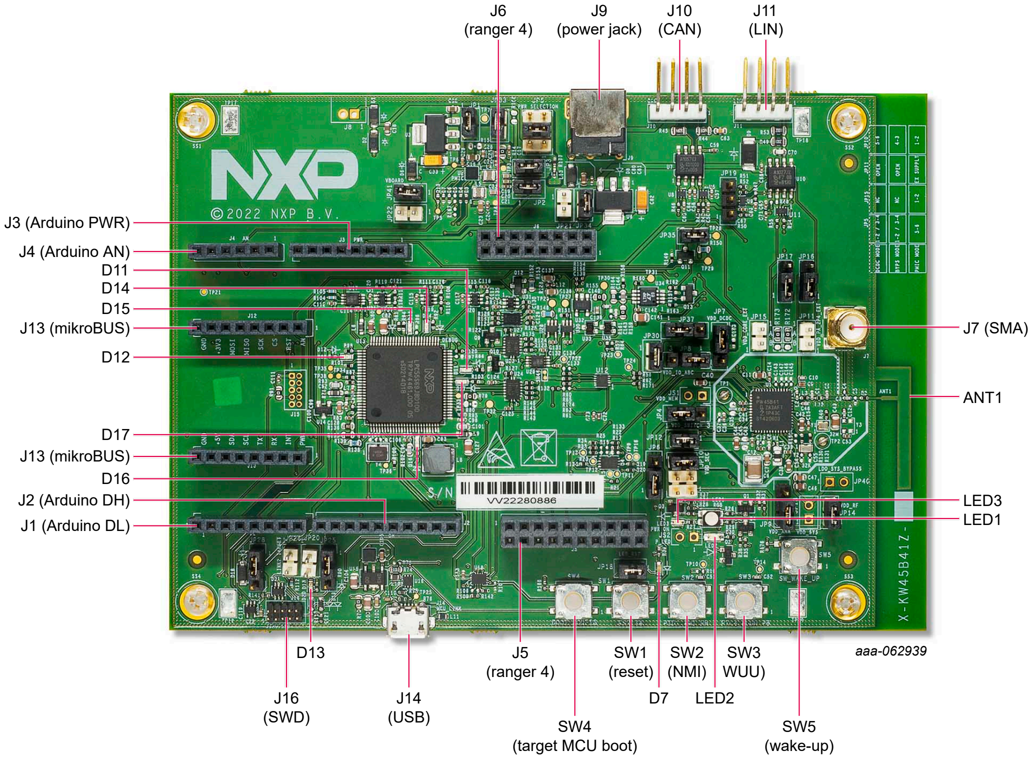


Figure 8. KW45B41Z-EVK connectors, push buttons, and LEDs

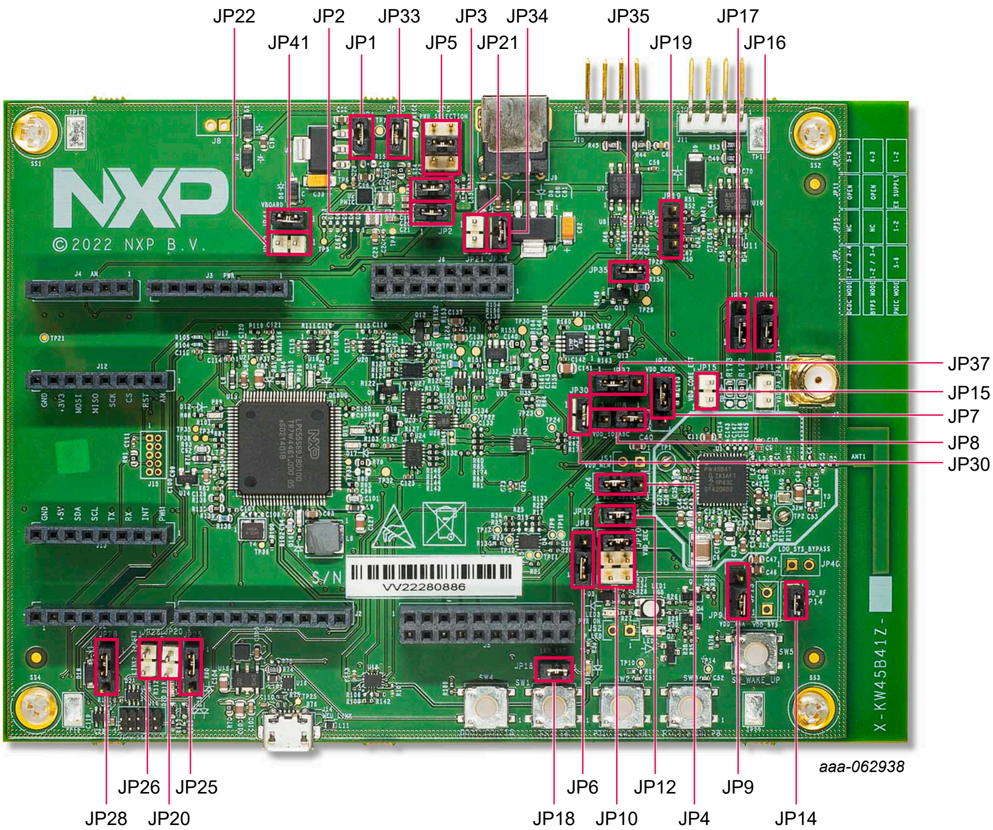


Figure 9. KW45B41Z-EVK jumpers

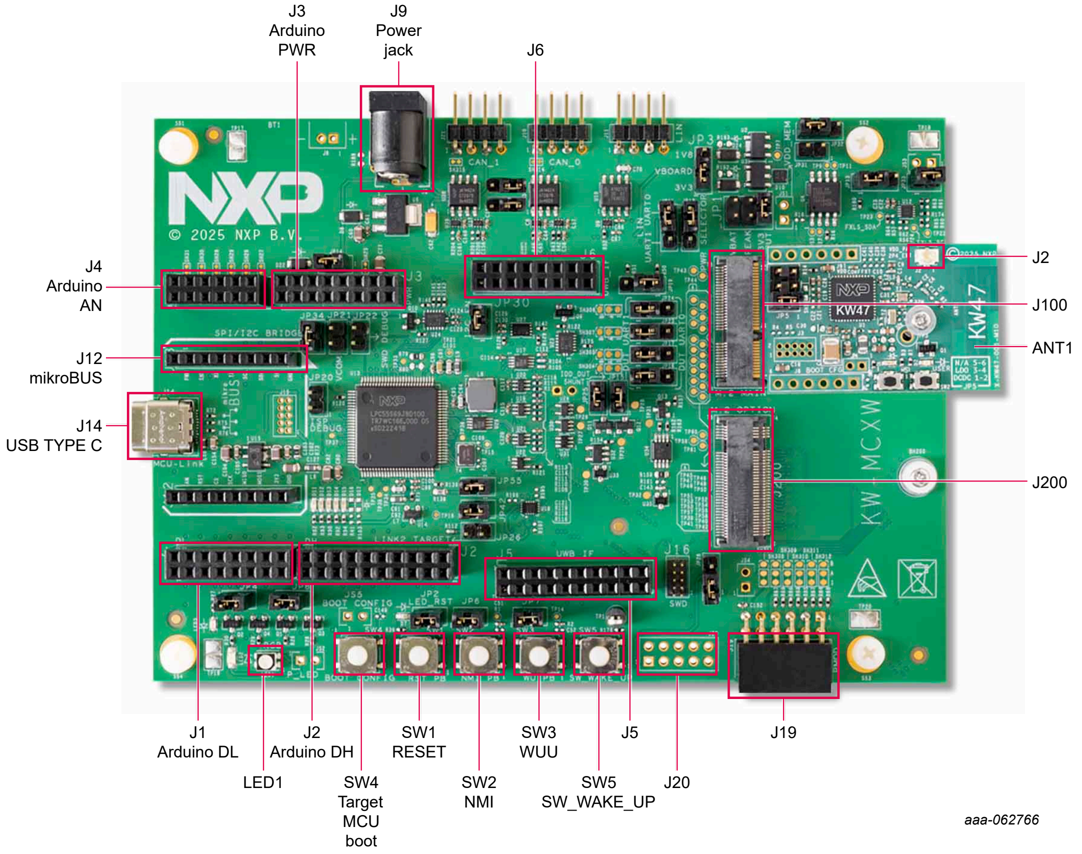


Figure 10. KW47-EVK connectors

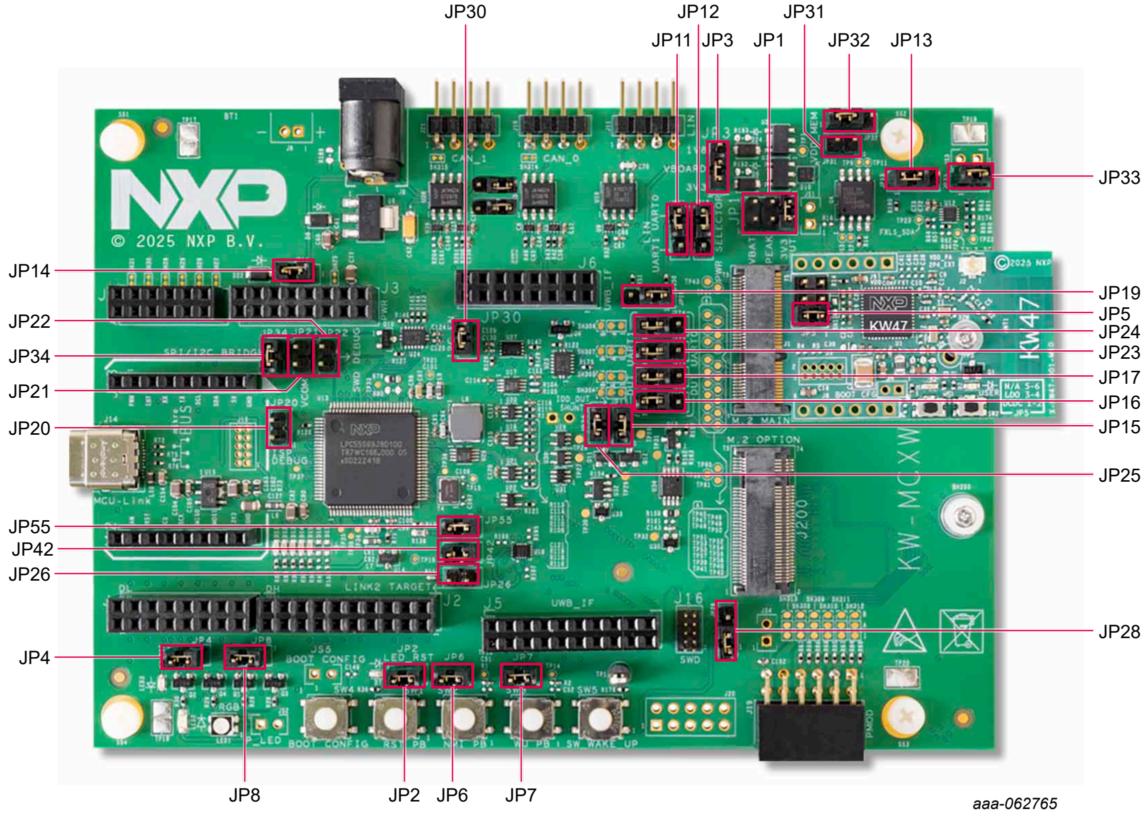


Figure 11. KW47-EVK jumpers

***Note:** Before executing any test, ensure that the board has the jumper configuration given in Figure 11.*

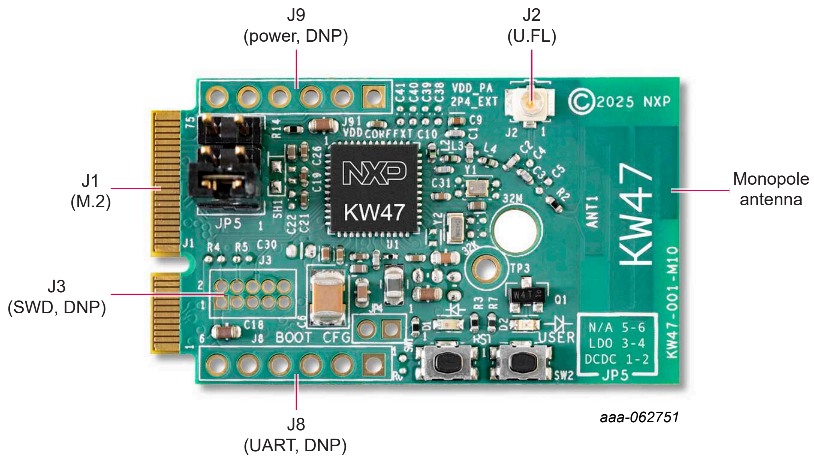


Figure 12. KW47-001-M10 module

4.1 Layout guidelines

To have a successful wireless hardware development, the proper device footprint, RF layout, circuit matching, antenna design, and RF measurement capability are essential. The RF circuit design, RF layout, and antenna design are specialties requiring investment in tools.

For KW45, the design and board layout considerations are described in the *Hardware Design Considerations for KW45B41Z and K32W148 Bluetooth LE Devices* (AN13227) and The best way to build a PCB first time right with KW45 (Automotive) or K32W1/MCXW71 (IoT/Industrial).

For KW47, the design and board layout considerations are described in the *KW47 Hardware Design Guide* (UG10127) and The best way to build a PCB first time right with KW45 (Automotive) or K32W1/MCXW71 (IoT/Industrial).

4.2 Power supply configurations

The KW45B41Z-EVK board can be powered up from any of the following:

- An external coin battery through USB connector (J14)
- An external DC supply (pin 1 of header J8/J9 connector and pin8 of J3 connector)

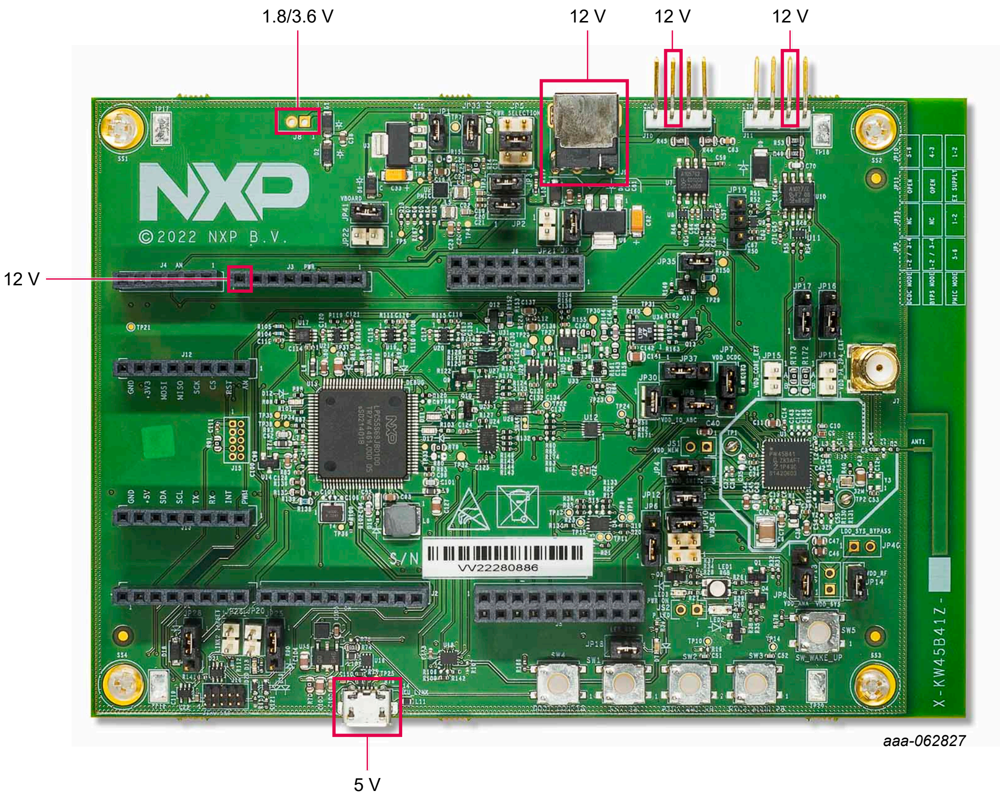


Figure 13. KW45 power supply configuration

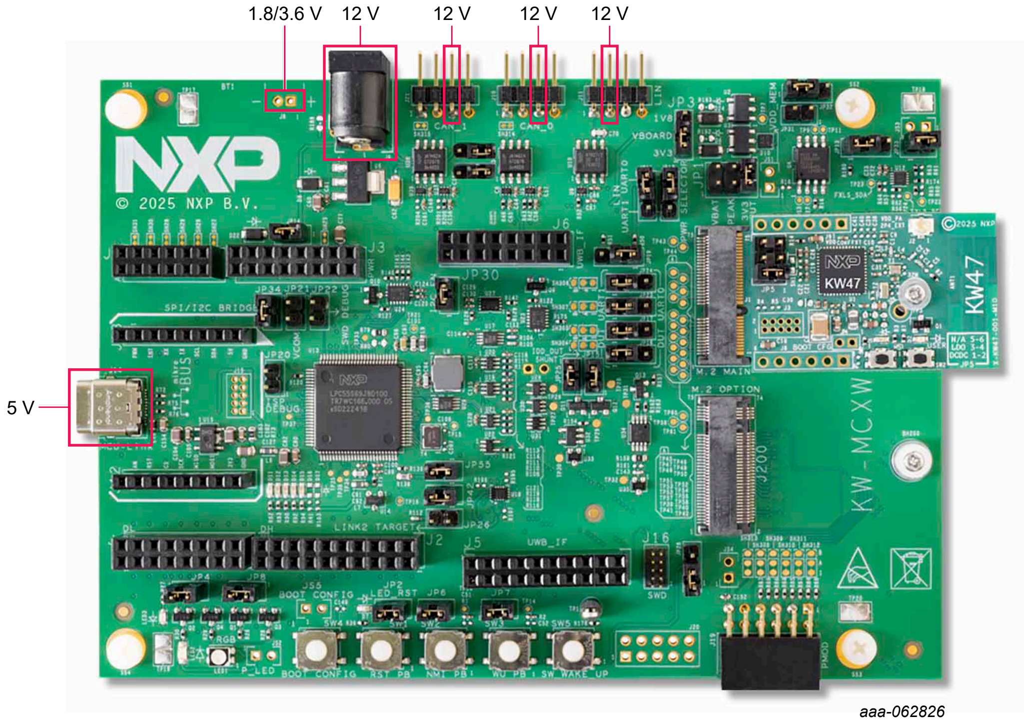


Figure 14. KW47 power supply configuration

The KW47-EVK board can be powered up from any of the following:

- An external coin battery through USB-Type C connector (J14)
- An external DC supply (pin 1 of header J8/J9 connector and pin8 of J3 connector).

The KW47-EVK board has an additional CAN.

4.3 Programming interfaces

To use an external debugger with the KW45B41Z-EVK and KW47-EVK board, the MCU-link on the board must be disabled. To disable the MCU-link, short the JP22 and then connect the external debugger to the serial wire debug (SWD) connector (J16).

Note: *Do not attach a debugger when the WAKE power domain is in Deep-sleep, Power down, or Deep-power Down. As the JTAG/SWD logic is powered off.*

The KW47-EVK has an onboard debugger MCU-link probe with CMSIS-DAP and SEGGER J-Link protocol options.

4.4 Power supply modes

The KW45 and KW47 has the following Power modes:

- DC-DC buck
- DC-DC bypass (LDO mode)
- Smart power switch

The PMIC supply mode is supported only on KW45.

For more details on switching between different Power modes, see section 2.1.2 in the *KW45B41Z-EVK Board User Manual* (KW45B41Z-EVKUM) and section 14 “Power Configuration” in the *KW47-EVK Board User Manual* (UM12094).

The KW45 and KW47 have several power rails that can be configured with independent power supplies to tradeoff between cost and power efficiency. These configurations involve the DC-DC, LDO-CORE, LDO-SYS, and smart power switch.

For more information on power management, see the *KW45/K32W148 - Power Management Hardware* (AN13831) for KW45 and see the *Power Management Hardware for the KW47* (AN14709) for KW47.

4.5 CAN

The KW45B41Z-EVK board allows communication with CAN0 module, which allows external CAN communication at 1x4 pin header (J10):

- 1 header: High-level CAN bus line connection
- 2 header: Low-level CAN bus line connection
- 3 header: Power connection (P12V)
- 4 header: Ground (GND)

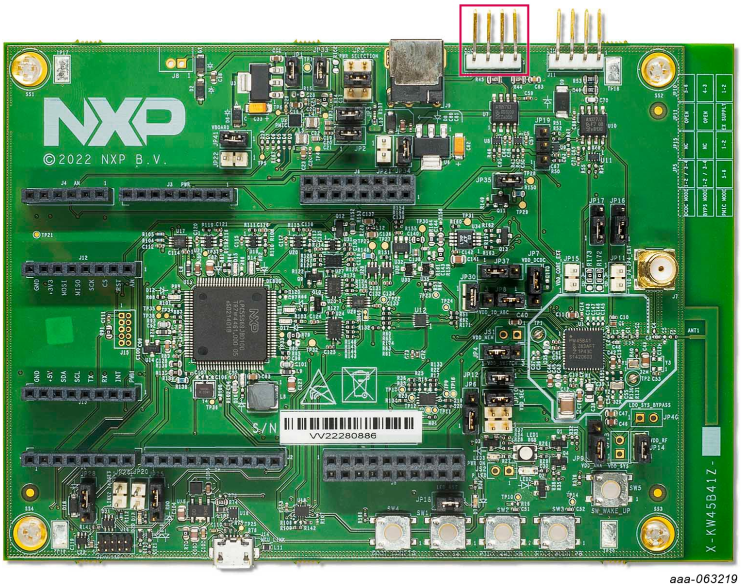


Figure 15. Pin header J10 for KW45

The CAN interface of the KW45B41Z-EVK board is functional if the board is powered up using the P12V supply. The power supply can be produced through any of the following:

- J9
- Pin 3 of J10
- Pin 2 of J11

The KW47-EVK board allows communication with an additional module, CAN0 and CAN1 module, which allows external CAN communication at 1x4 pin header (CAN_0-J10 and CAN_1-J21):

- 1: High-level CAN bus line connection
- 2: Low-level CAN bus line connection
- 3: Power connection (P12V)
- 4: Ground (GND)

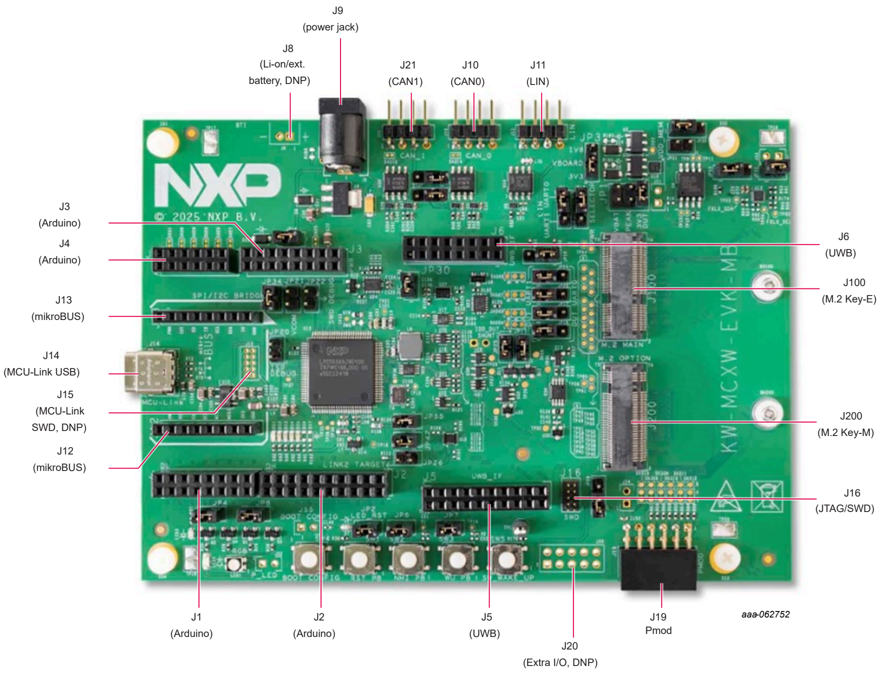


Figure 16. Pin header CAN_0-J10 and CAN_1-J21 for KW47

The CAN interface of the board is functional if the board is powered up using the P12V supply. The power supply can be produced through any of the following:

- J9
- Pin 3 of J10
- Pin 2 of J11
- Pin 3 of J21

4.6 LIN

The KW45B41Z-EVK board supports connections to both LPUART modules with LIN support. By default, the LIN header (J11) is connected to LPUART0, but it can be connected to LPUART1 after reconfiguring R172 and R173.

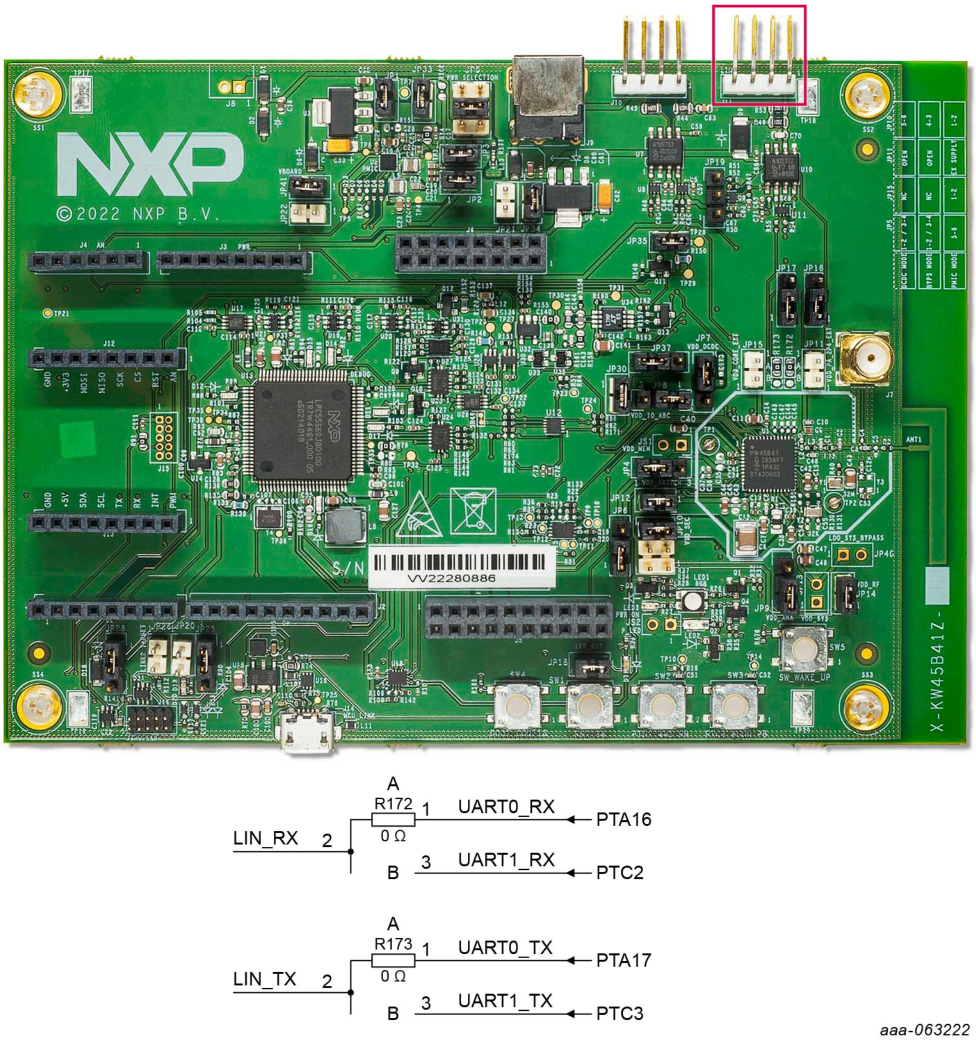


Figure 17. LIN header J10 for KW45

The LIN interface of the board is functional if the board is powered up using the P12V supply. The power supply can be produced through any of the following:

- J9
- Pin 3 of J10
- Pin 2 of J11

The KW47-EVK board supports connections to both LPUART modules with LIN support. By default, the LIN header (J11) is connected to LPUART0, but it can be connected to LPUART1 and CAN1 after reconfiguring JP11 and JP12.

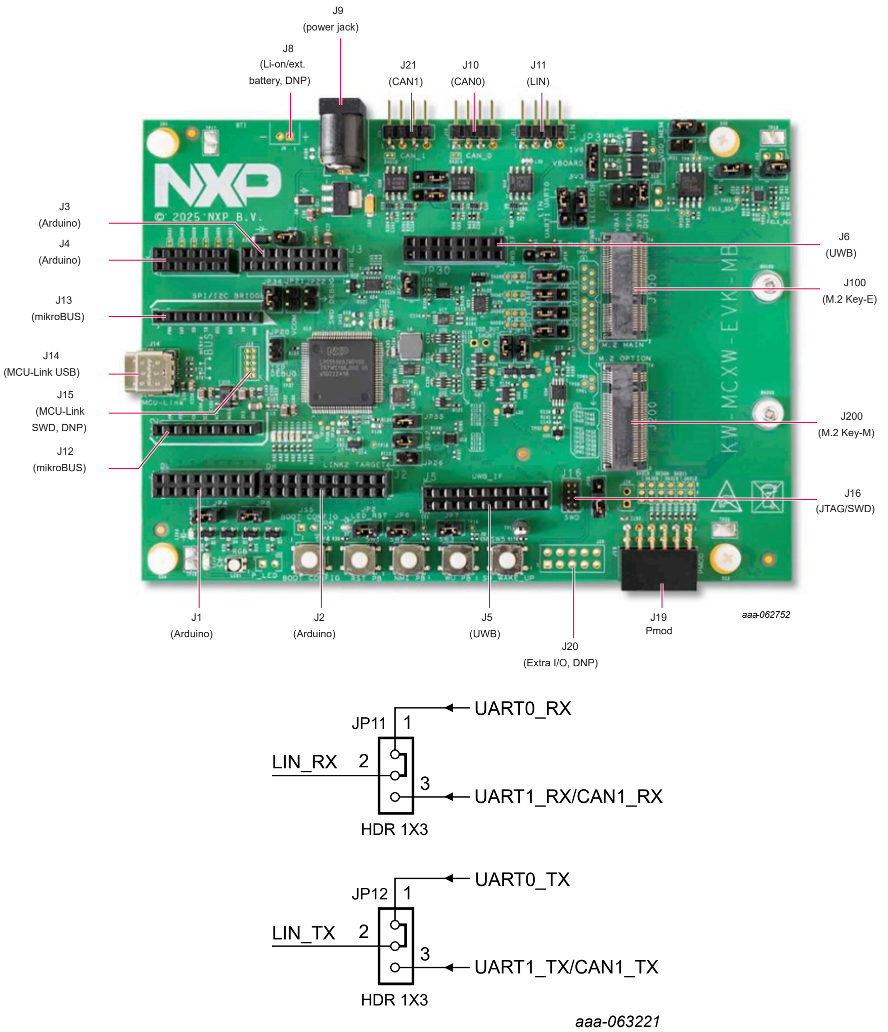


Figure 18. LIN header J11 for KW47

The LIN interface of the board is functional if the board is powered up using the P12V supply. The power supply can be produced through any of the following:

- J9
- Pin 3 of J10
- Pin 2 of J11
- Pin 3 of J21

4.7 Current measurement

Both the KW45B41Z-EVK and KW47-EVK boards support the following two methods for measuring the current:

- Using the onboard MCU-Link current measurement circuit.
- Connecting an ampere meter to two of the pins of a power supply jumper.

For more details, see section 2.1.1 in the *KW45B41Z-EVK Board User Manual* (KW45B41Z-EVKUM) and section 12 in the *KW47-EVK Board User Manual* (UM12094).

5 Software considerations

This section contains the software considerations.

5.1 Low-power specifications

Both KW45 and KW47 devices support Active, Sleep, Deep Sleep, Power Down, and Deep Power Down modes. Any reset brings the chip back to the Active mode.

There is also an additional power mode called “VBAT ON” which can be referred to as a “Smart Power Switch”. For more information, refer to Chapter 31 Smart Power Switch (VBAT) of the KW47 Reference Manual and the AN14684 for the Features, Usage, and Capabilities of Smart Power Switch on the KW47 Microcontroller.

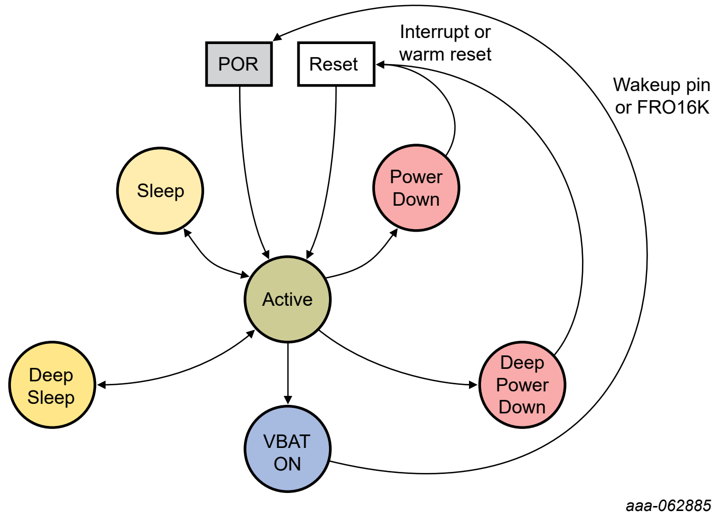


Figure 19. KW45 and KW47 low-power modes

For more details on the Power modes and its description, see Chapter 30.2 in the *KW45 Reference Manual* (KW45RM) and Chapter 28.3 in the *KW47 Reference Manual* (KW47RM).

5.2 Download and install the Software Development Kit (SDK)

This section shows how to download the SDK for the KW45 and KW47 devices.

5.2.1 Importing MCUXpresso SDK repository to VS Code (remote)

To import the NXP SDK into visual studio code, import a remote repository as follows:

1. Select the **Import Repository** in the quick start panel of MCUXpresso for VS Code.

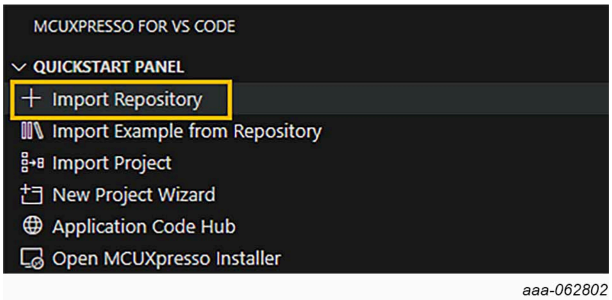


Figure 20. MCUXpresso for VS Code

2. Select the MCUXpresso SDK Repository, in the **REMOTE** tab.
3. Select the option from using the NXP GitHub repository that has revisions from 24.12 or later.
4. In the **Revision** field, by default **main** is set. If needed, select the required revision number from the drop-down list.
   - In the **Location** field, select an address that is the closest possible to the main device (C:).
5. Select **Import**.

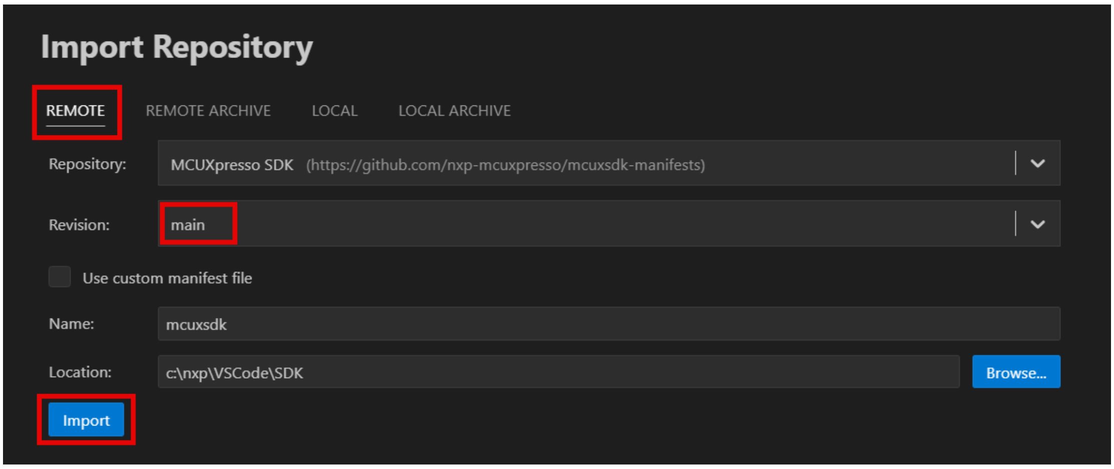


Figure 21. Import repository

6. When this process is finished, the **MCUXpresso SDK Repository** appears in the imported repositories window.

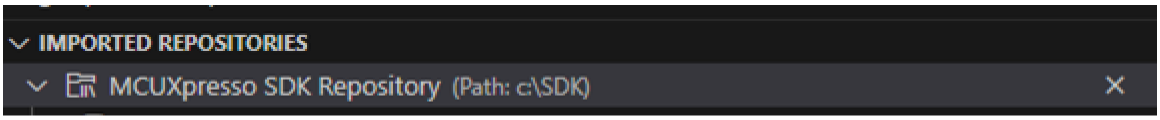


Figure 22. MCUXpresso SDK repository

6 Program NBU firmware for wireless examples

The KW45 and KW47 runs the Radio LL software from the dedicated NBU. It includes a Cortex-M3 in the KW45 and a Cortex-M33 in the KW47.

The NBU firmware is included in the SDK folder:

- On KW45, as a secure binary firmware (SB3) preprogrammed for NXP default keys, and as a XIP file to generate an SB firmware based on custom keys.

- On KW47, it is included as a binary (.bin) firmware in the SDK folder.

The main difference between the NBU on the KW45 and KW47 lies in how each device manages access and security during the boot process. On KW45, access to the NBU flash is restricted to the Boot ROM, which means that the device always requires an SB container (SB3 file) to authenticate and boot correctly. This SB3 file must be signed using the OEM authentication keys already provisioned on the device and security keys for encryption.

In contrast, KW47 supports an open NBU, which provides direct access to the NBU.There is no need for an SB3 file when communicating with the boot ROM during development. Use standard SB files (.bin) provided in the SDK to update the NBU firmware using ISP commands, as long as the MCU remains in the development lifecycle state. If the lifecycle is advanced to production, an SB3 file is required for updates, similar to KW45.

The NBU binary file is located in the SDK folder. Go to the SDK root folder and open the following path: \main \mcuxsdk\mcuxsdk\middleware\wireless\ble_controller\bin.

For KW45, the SB3 and XIP files are included for the binary file “kw45b41_nbu_ble_hosted”.

For KW47, the binary file “kw47_nbu_ble_all_hosted” is available.

Note: *It is necessary to work with the matching NBU image for the SDK version of the application you are working with. After downloading the SDK, update the NBU image with the binaries provided in the SDK folder before loading any wireless SDK example.*

6.1 Load NBU firmware in KW45

The NBU firmware is distributed within the SDK as an `.sb3` file or as a binary file (`*.xip`). The following are the two options to update the NBU firmware.

1. For the KW45B41Z-EVK board, use a prepared SB file with NBU firmware. The NBU is programmed with a signed image with provisioned NXP keys. NXP provides the default keys in the fuse (SB3KDK and RoTKTH). The `*.sb3` file related to these keys is included in each SDK.
2. The other way is to prepare a custom SB file that loads the NBU firmware. The samples are not programmed with any keys. The samples for which the user can create custom keys are included in the `.xip` file.

For the KW45 chip received from the factory, by default the SBKDK and RoTKTH keys in the fuse are null. Therefore, writing these two keys to fuse is essential.

The following sections guide you through programming the NBU software for the KW45B41Z-EVK board with two different method applications.

6.1.1 Bootloader Host application (blhost)

The user can upload or change the NBU firmware using the Bootloader Host Application (blhost), setting the board into ISP mode.

The following steps guide the user through programming the NBU software for the KW45B41Z-EVK board:

1. Open a command prompt.
2. Change directory to the location of blhost.exe file `(BLHost_root_location)\blhost_2.6.7\bin\win`.
3. Place your device in the ISP mode. For this example, the UART peripheral is used by connecting a USB cable to J14. On the EVK board, the user can enter ISP by the following method:
   a. Press and hold SW4.
   b. Press and release reset.
   c. Release SW4.

4. To check the MCU-Link COM, open the device manager. In this example, the COM port assigned is "COM23".

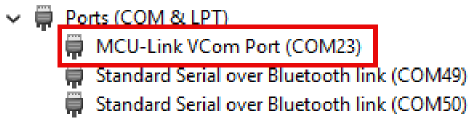


Figure 23. MCU-Link COM port

5. In the command prompt, write the next command to verify if the communication is working. Do not forget to replace the **COMX** with your COM device:

```
blhost -p COMX get-property 1
```

6. Update the NBU firmware using the next commands. The path for the binary is required, or you can copy the binary file to the same folder as blhost.exe:

```
blhost -p COMX,57600 receive-sb-file kw45b41_nbu_ble_hosted.sb3
```

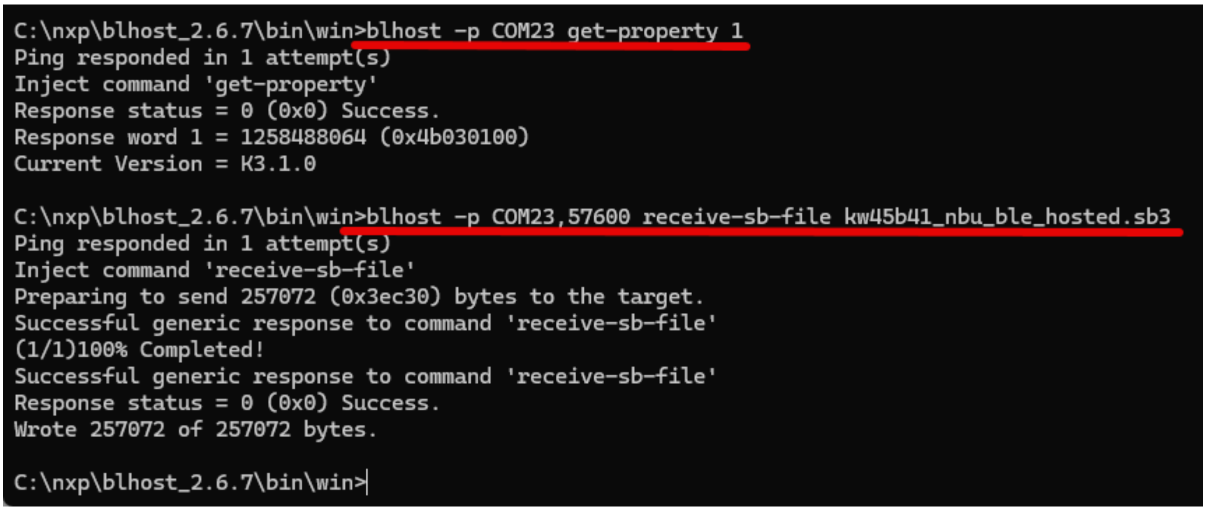


Figure 24. Update the NBU firmware

6.1.2 Secure Provisioning Tool (SEC Tool)

The NBU firmware can also be uploaded or changed using the NXP Secure Provisioning Tool (SEC Tool). The GUI-based application can be downloaded from: MCUXPRESSO-SECURE-PROVISIONING.

To upload the SB3 file with NXP keys on the KW45B41Z-EVK using SEC Tool, follow these steps:

1. Create or open a workspace for KW45:

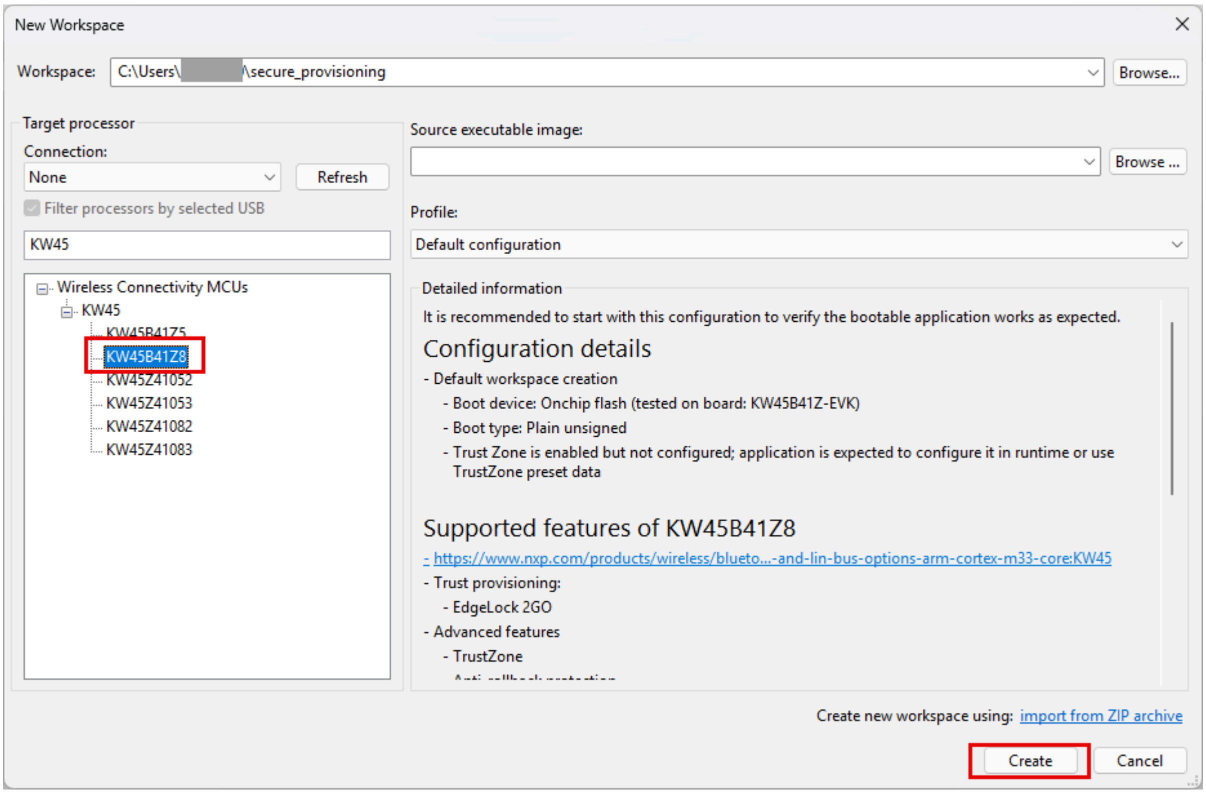


Figure 25. Create a workspace for KW45

2. In the menu bar, select Tools > Manufacturing Tool.
3. Select the Apply SB file operation.
4. Provide the SB file. When using the KW45B41Z-EVK, the SB file is provided in the following folder of the SDK: `/path_to_SDK/middleware/wireless/ble-controller/bin`
5. To detect a connected device, click the **Auto detect** and **Test connection** buttons.

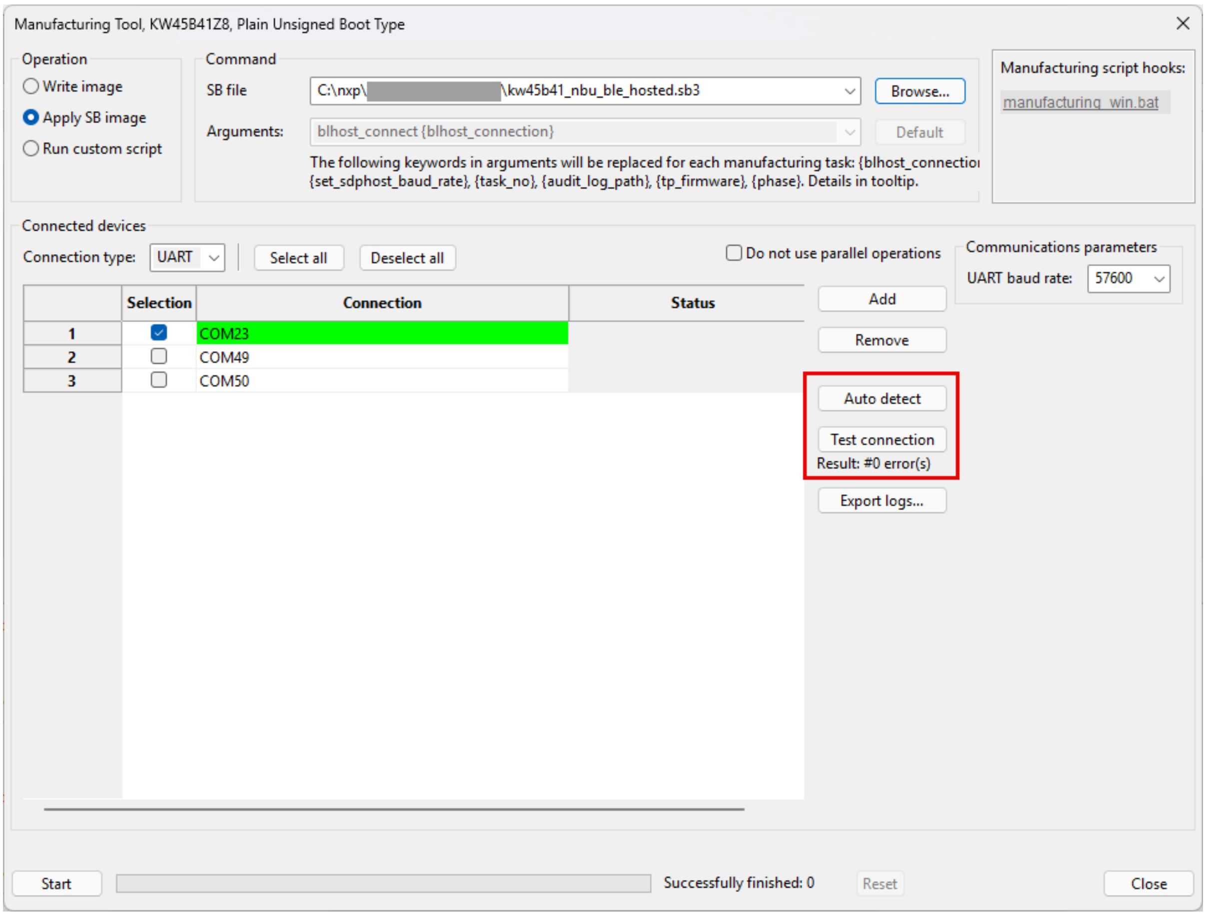


Figure 26. Detect a connected device

6. To load the SB file, click the **Start** button. If the SB file is uploaded correctly, a success message appears.

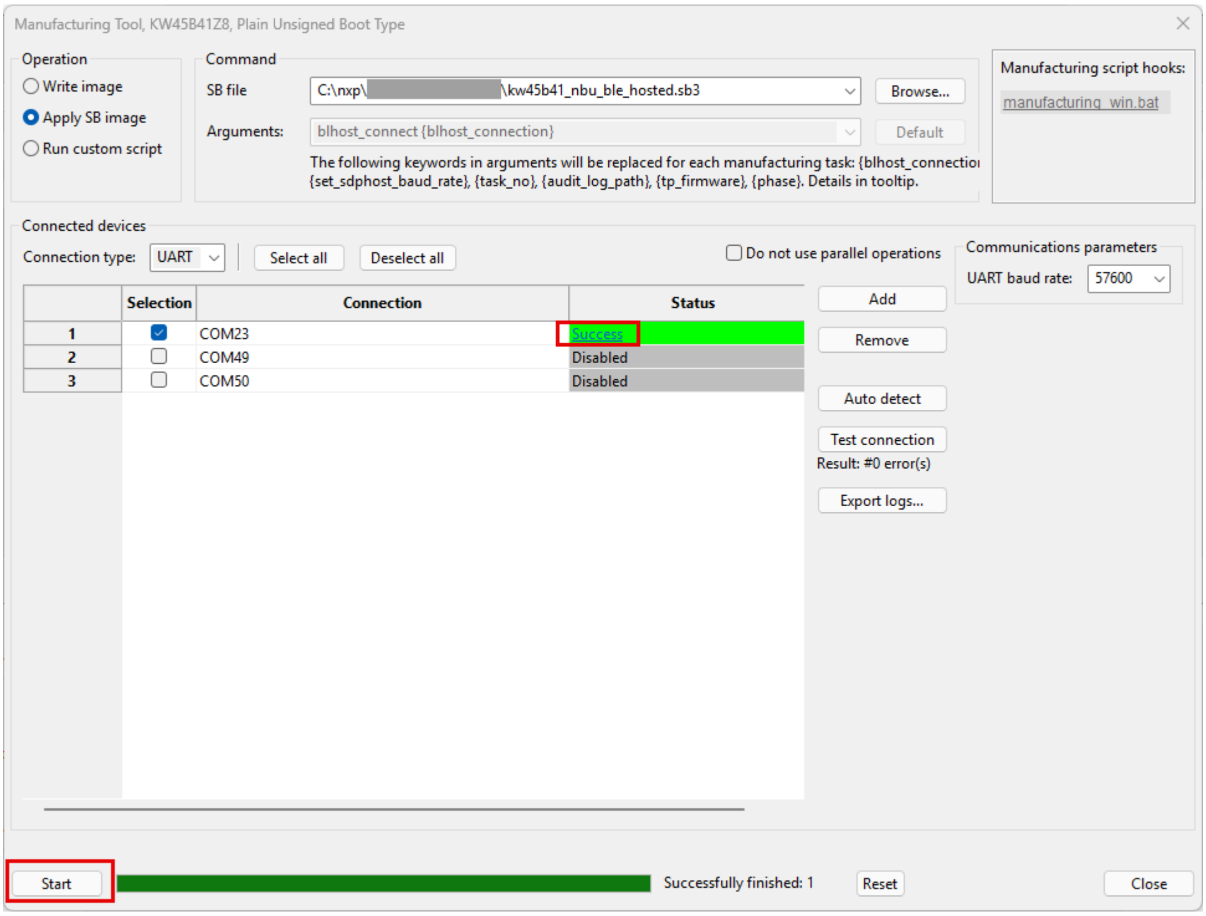


Figure 27. Load the SB file

6.2 Load NBU firmware in KW47

For KW47, the NBU firmware is included in the SDK folder as binary firmware at the following path: `SDK_X_KW47-EVK\middleware\wireless\ble_controller\bin\kw47_nbu_ble_hosted.bin`

6.2.1 Bootloader Host application (blhost)

The following steps guide the user through programming the NBU software for the KW47-EVK board using the Bootloader Host Application (blhost):

1. To enter the ISP mode, hold the SW4 on the KW47-EVK board, attach the USB connector J14 to the computer. Then, release the SW4 after plugging the USB cable on the computer.

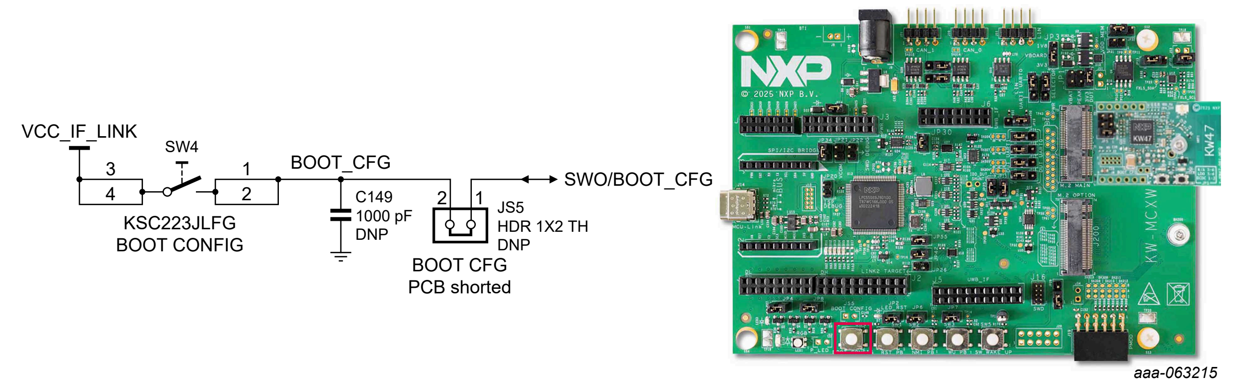


Figure 28. Enter the ISP mode

2. Verify which COM port is assigned to the KW47-EVK board. To consult the COM Port assigned in the Windows "Device Manager" program; search for "Ports (COM & LPT)" and save the COM Port number.

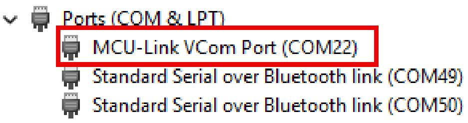
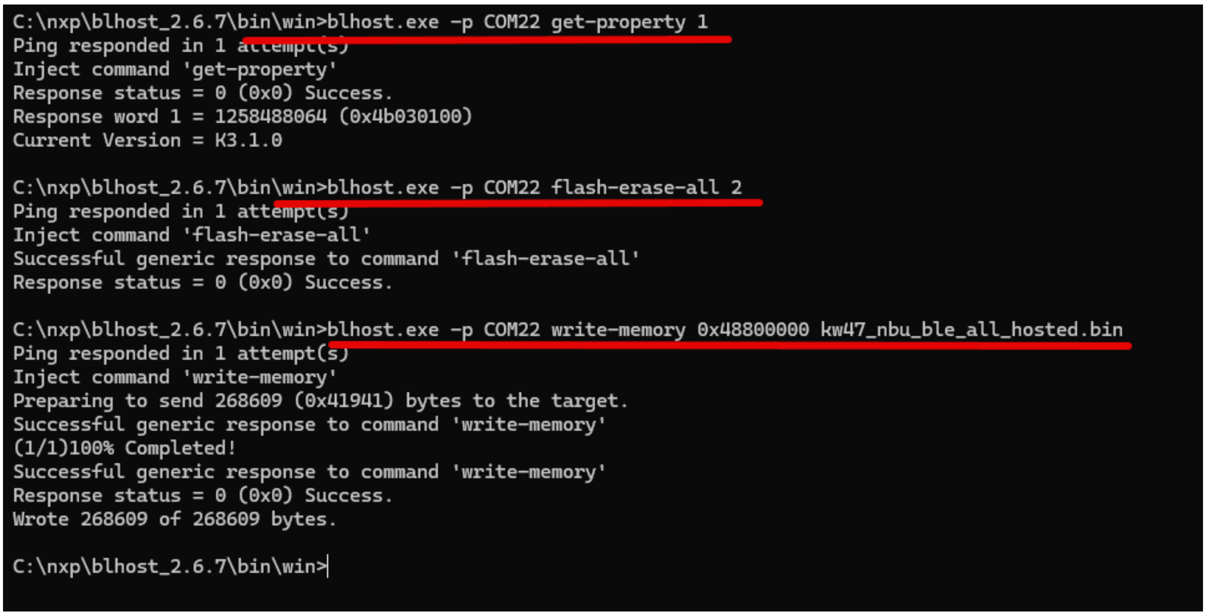


Figure 29. MCU-Link COM port

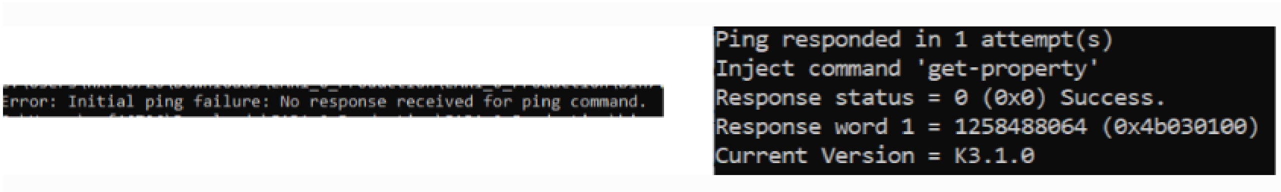
3. Open a command prompt. Change the directory to the location of blhost.exe `(BLHost_root_location)\blhost_2.6.7\bin\win`
   To verify that the communication is working correctly, run the command `blhost.exe -p COMX get-property 1`
   Note: *Ensure to replace the COMX with the COM port assigned to your device.*
4. Update the NBU firmware using the next commands. First, erase the NBU memory:
   `blhost.exe -p COMX flash-erase-all 2`
5. Write the new firmware image. The write-memory command uses 0x4880 0000 as the start address, which corresponds to the NBU memory base. Provide the full path to the binary file, or alternatively, to simplify the command, copy the .bin file into the same directory as blhost.exe:
   `blhost.exe -p COMX write-memory 0x48800000 kw47_nbu_ble_all_hosted.bin`



```
C:\nxp\blhost_2.6.7\bin\win>blhost.exe -p COM22 get-property 1
Ping responded in 1 attempt(s)
Inject command 'get-property'
Response status = 0 (0x0) Success.
Response word 1 = 1258488064 (0x4b030100)
Current Version = K3.1.0

C:\nxp\blhost_2.6.7\bin\win>blhost.exe -p COM22 flash-erase-all 2
Ping responded in 1 attempt(s)
Inject command 'flash-erase-all'
Successful generic response to command 'flash-erase-all'
Response status = 0 (0x0) Success.

C:\nxp\blhost_2.6.7\bin\win>blhost.exe -p COM22 write-memory 0x48800000 kw47_nbu_ble_all_hosted.bin
Ping responded in 1 attempt(s)
Inject command 'write-memory'
Preparing to send 268609 (0x41941) bytes to the target.
Successful generic response to command 'write-memory'
(1/1)100% Completed!
Successful generic response to command 'write-memory'
Response status = 0 (0x0) Success.
Wrote 268609 of 268609 bytes.

C:\nxp\blhost_2.6.7\bin\win>
```

Figure 30. Update the NBU firmware

Note: *You can run the command* `blhost.exe -p COMX get-property 1` *to verify that the board has entered ISP mode properly. If the device does not enter in the ISP mode, it cannot be programmed. Figure 31 show when the device does not enter in ISP (left) and when the device is programmed properly (right).*



```
Error: Initial ping failure: No response received for ping command.
```

```
Ping responded in 1 attempt(s)
Inject command 'get-property'
Response status = 0 (0x0) Success.
Response word 1 = 1258488064 (0x4b030100)
Current Version = K3.1.0
```

Figure 31. Verify ISP mode has been entered

6.2.2 Secure Provisioning Tool (SEC Tool)

The NBU firmware can also be uploaded or changed using the NXP Secure Provisioning Tool (SEC Tool). The GUI-based application can be downloaded from: MCUXPRESSO-SECURE-PROVISIONING.

To upload the SB3 file with NXP keys on the KW47-EVK using SEC Tool, follow these steps:

1. Create or open a workspace for KW47.

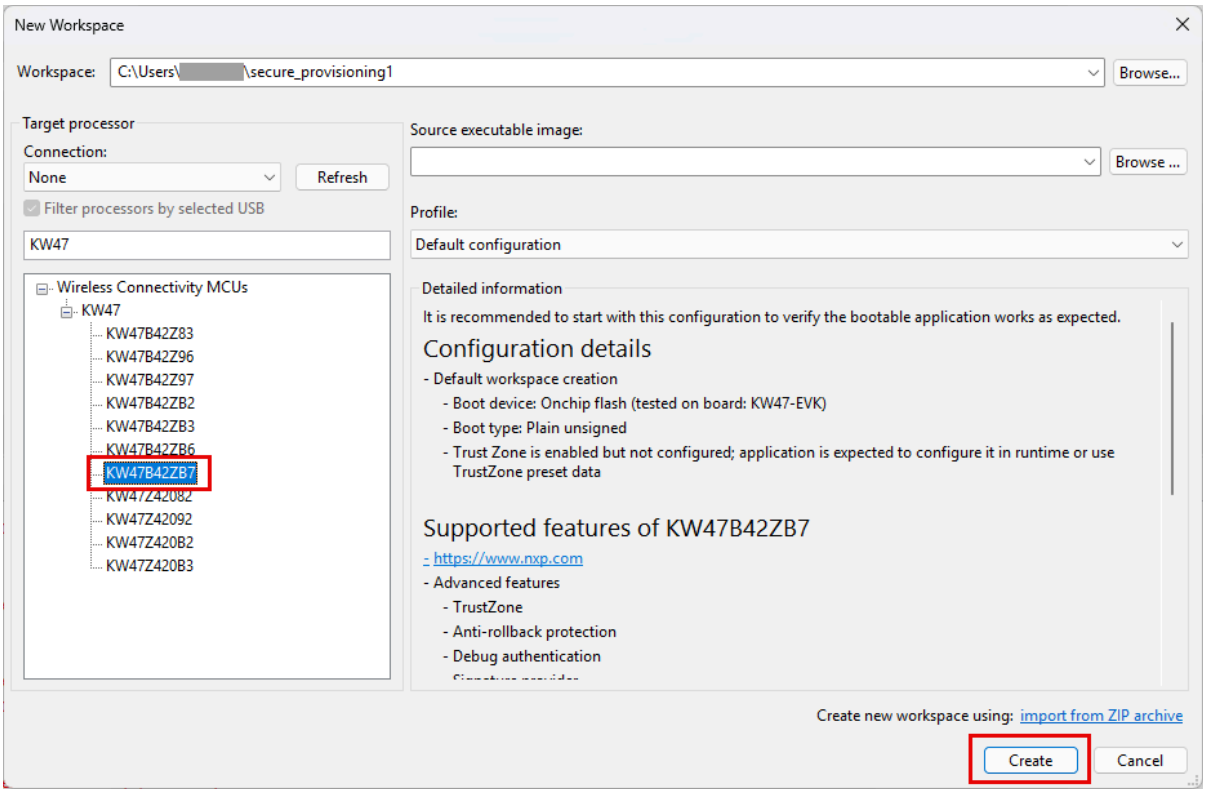


Figure 32. Create a workspace for KW47

2. Ensure that the device is in the ISP mode. Hold the SW4 on the KW47-EVK board and attach the USB connector J14 to the computer. Then, release the SW4 after plugging the USB cable on the computer. To verify that the communication is working correctly, click the **UART** view, fill the COM port and **Test connection**.

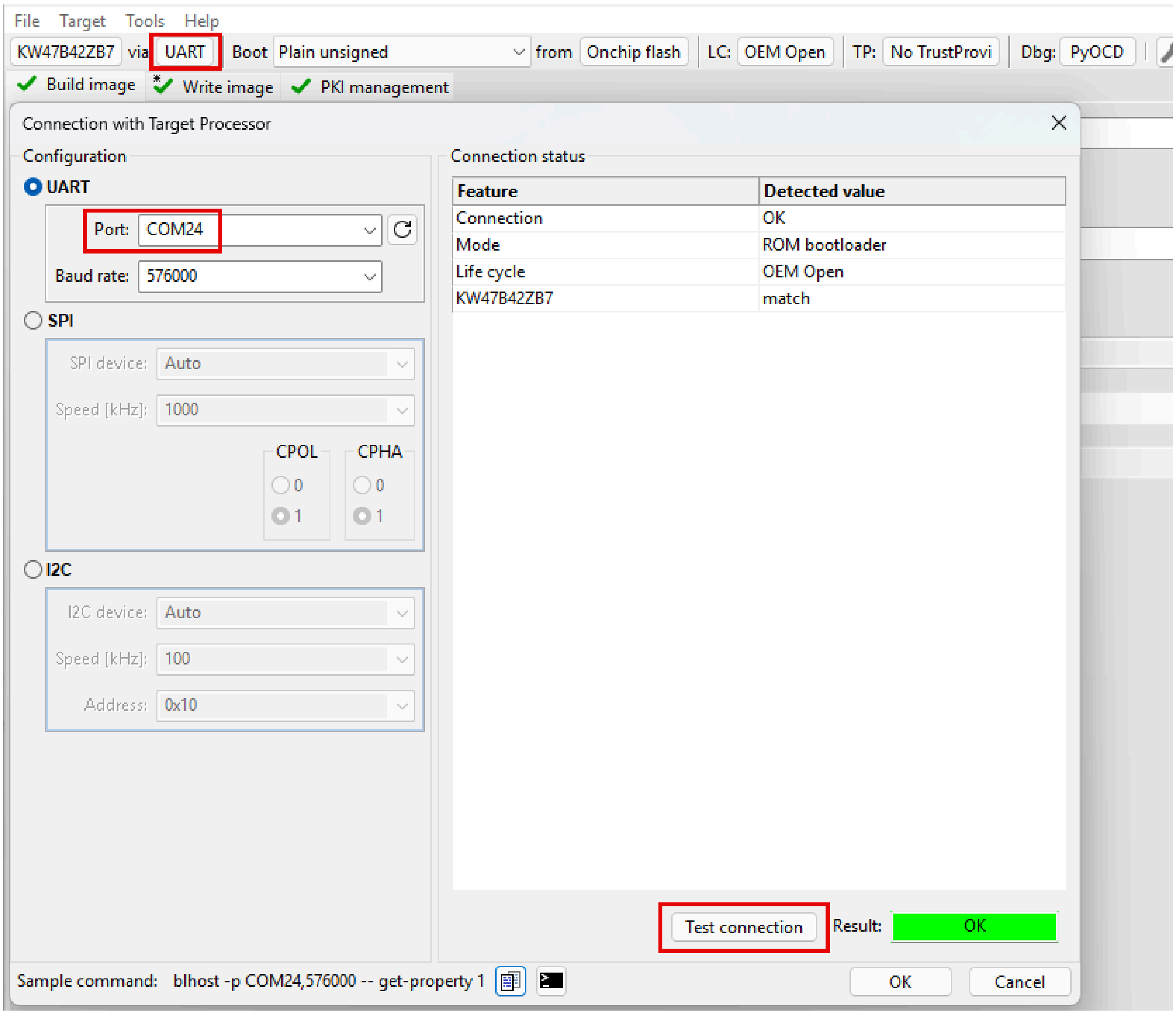


Figure 33. Test a connection

3. Ensure that you have selected the **Plain unsigned** or Plain with CRC boot type in the toolbar.
4. In the **Build image** tab:
   a. Load the binary file in the **Source executable image**.
   b. Write the **Start address** as **0x48800000**, which corresponds to the NBU memory base.
   c. Click **Build image**.

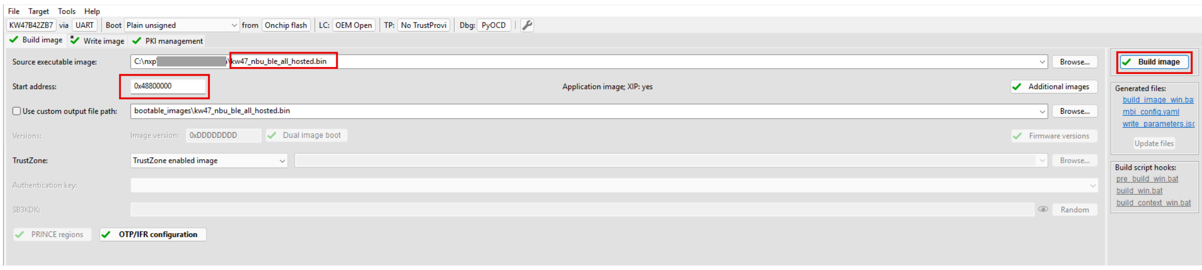


Figure 34. Build the image

5. Move to the **Write image** tab:
   a. Select **Use built image**.

b. Click **Write image**.

Figure 35. Write the image

6. A success message appears when the NBU firmware loading is completed.

Figure 36. NBU firmware loading

6.2.3 LinkFlash

A third way of updating the NBU for KW47-EVK is using the LinkFlash tool from LinkServer. To use this tool, LinkServer debug probe must be available on the board.

To install the LinkServer, download it from LinkServer for Microcontrollers by clicking Downloads, then select the package corresponding to your OS. The user can also use the MCUXpresso installer within the VS Code extension. To download, Select the LinkServer checkbox and click install.

1. Inside the LinkServer folder, go to path MCU-LINK_installer\scripts, where two scripts are available, one for programming CMSIS debug probe and another for J-Link debug probe. For LinkServer, program the CMSIS.

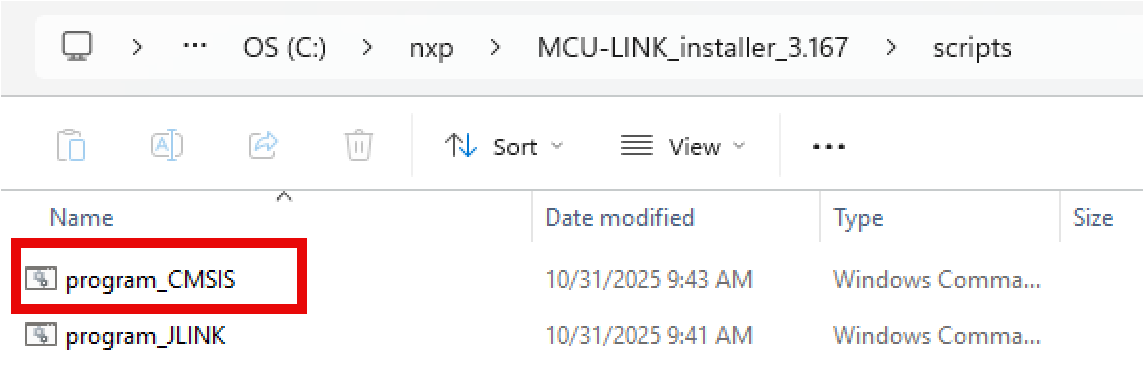


Figure 37. Program the CMSIS

2. Open the **program_CMSIS** script and place the board in the ISP USB mode by placing a jumper on JP20.
3. Go back to the main LinkServer folder and execute the LinkFlash.exe file.
4. Ensure that the device is in the ISP mode. Hold the SW4 on the KW47-EVK board and attach the USB connector J14 to the computer. Then, release the SW4 after you plugged the USB cable on the computer.
5. Once the device is connected, click the “Refresh” button to update and set the board probe target.
   Note: *Ensure to remove the JP20 jumper prior to this step.*

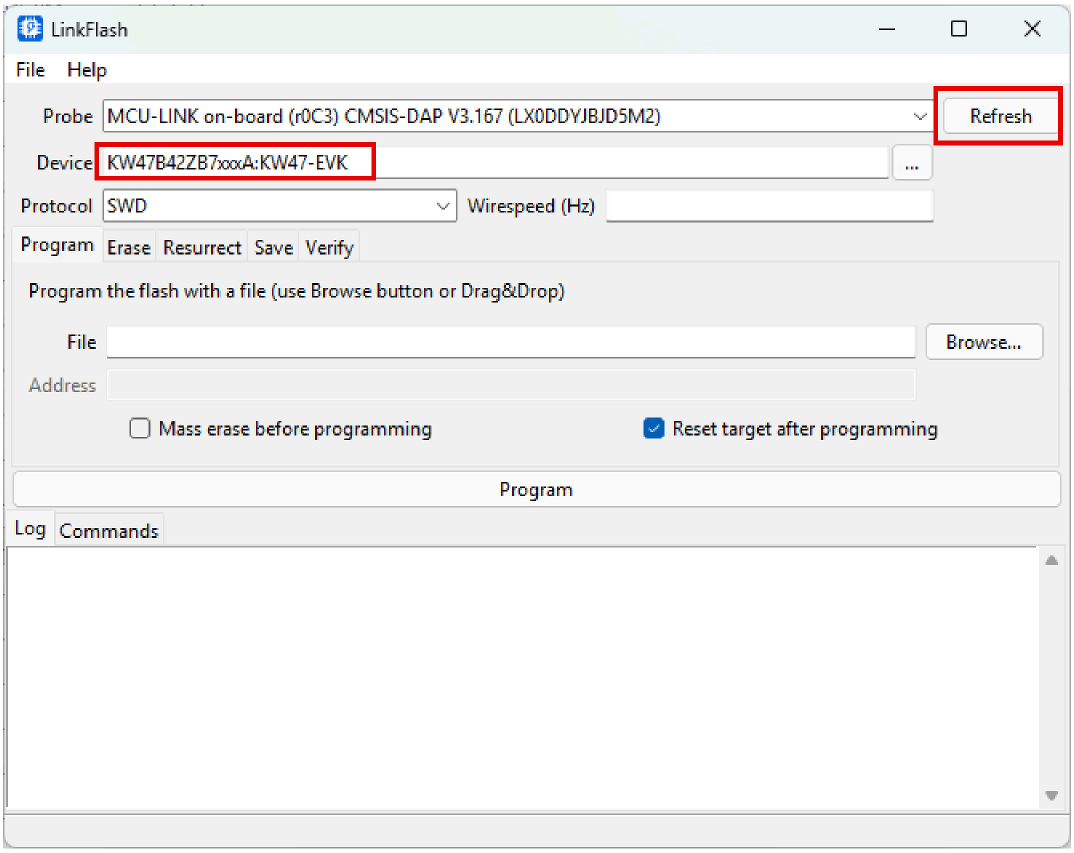


Figure 38. Update and set the board probe target

6. Go to the **Program** tab, click **Browse**, and select the NBU firmware file. Set the **Address** to **0x48800000**.
7. Ensure to select checkboxes **Mass erase before programming** and **Reset target after programming**.

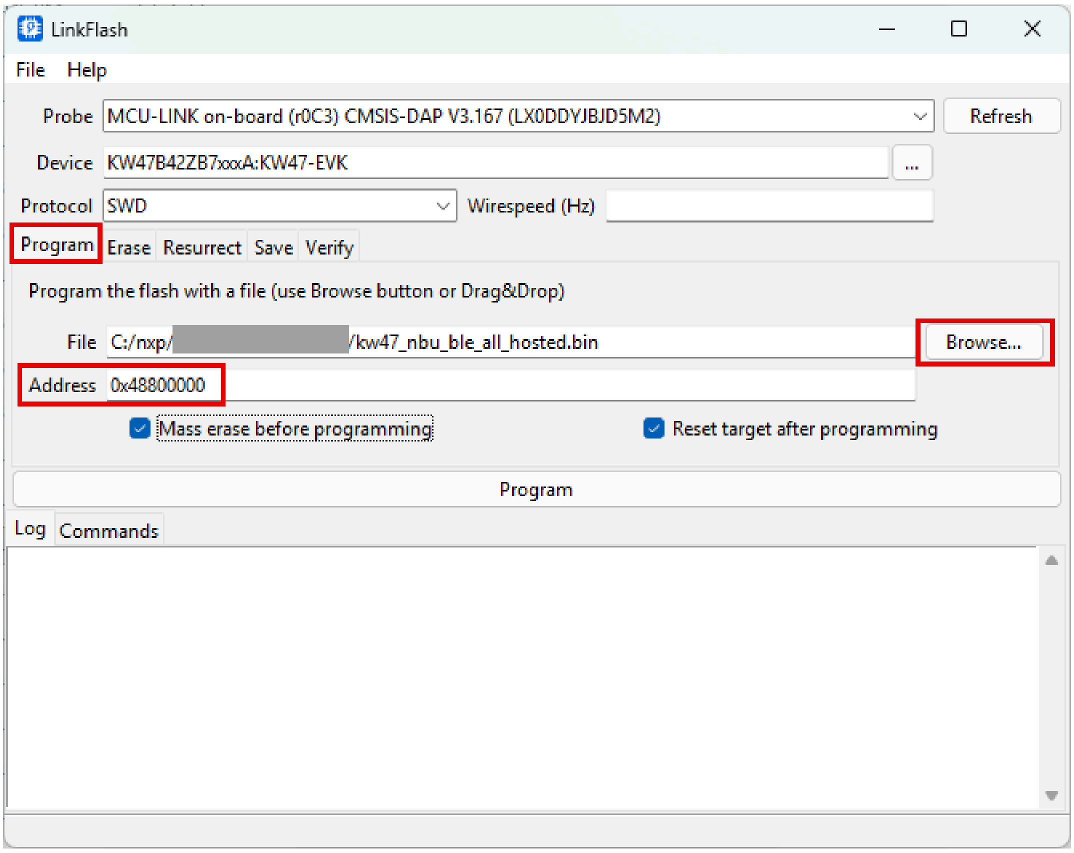


Figure 39. Load the NBU firmware

8. To load the NBU firmware, click **Program**.

6.3 Bluetooth LE migration project

This section outlines the procedure for migrating an existing Bluetooth LE application from KW45 to KW47. The following considerations must be considered.

To migrate an application for KW47, it is recommended to use an example from the SDK as a base for the destined MCU, to ensure proper configuration. The application level can vary depending on the specific use case. SDK examples require application-level modifications implemented by the user to meet specific needs.

To migrate examples that involve a type of connection for peripherals can include more modifications on proper declaration, instance, and functionality in corresponding hardware files. Examples for custom boards require corresponding adjustments for the hardware changes implemented by the final user.

The migration example is based on the SDK example kw45b41zevk_beacon_freertos and references the Community Post Updating Beacon Data for beacon update testing.

1. From the “Quickstart Panel” window, click **Import Example from Repository**.
   a. Select the KW47-EVK repository from the **Repository** list.
   b. Select the KW47-EVK board from the **Board** list.
   c. Select the beacon_freertos example from the **Template** list.
   d. Select the **Freestanding application** from the **App type** list. Modifying SDK files within a freestanding/standalone project is a cleaner way of doing so.
   e. Click **Import**.

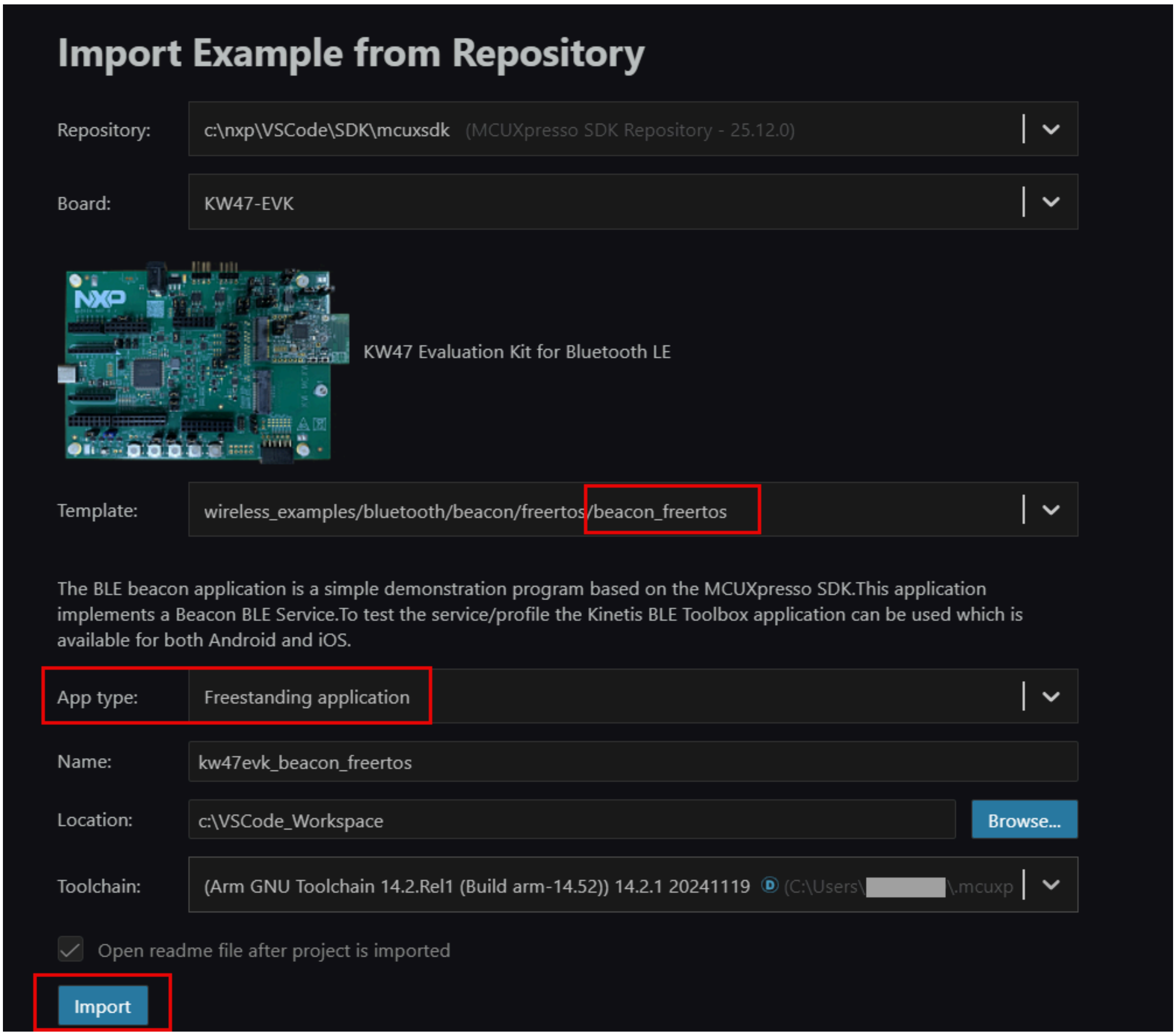


Figure 40. Import example from repository

2. From the pre-existing KW45B41Z-EVK beacon project, select each of the modified files:

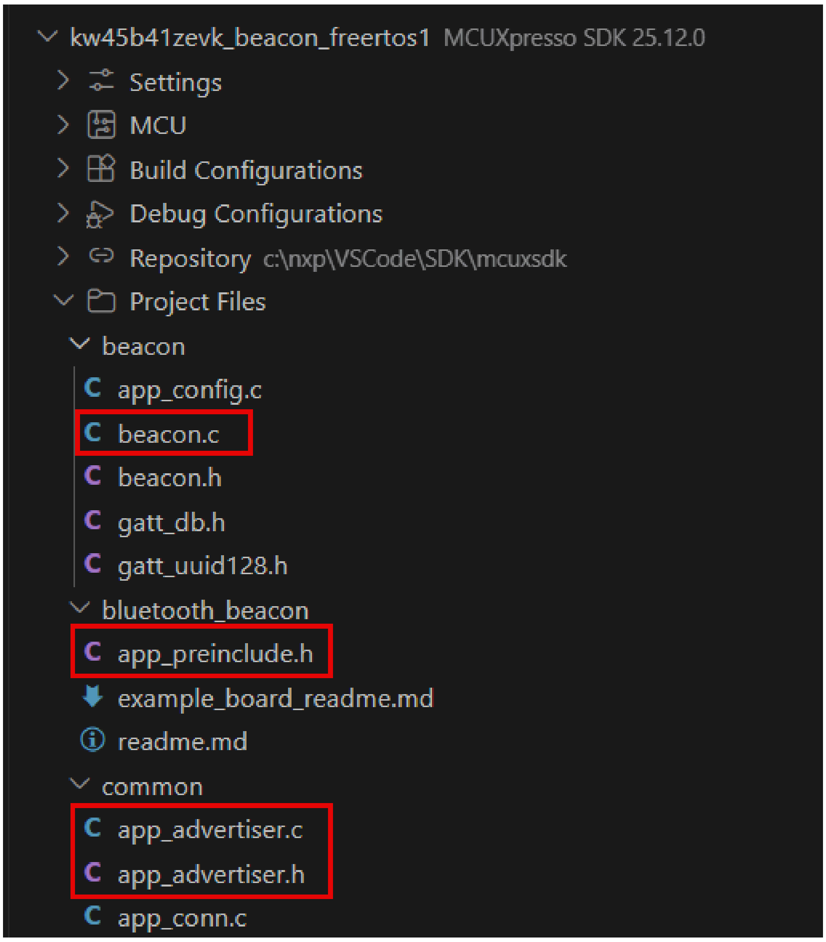


Figure 41. Select the modified files

3. Drag and drop each of the files into their respective folder in the “Project Files” section of the KW47-EVK project. A pop-up window appears, click **Copy files**, and then **Yes** to replace the pre-existing file. Repeat this step for every modified file that you wish to migrate to the new project.

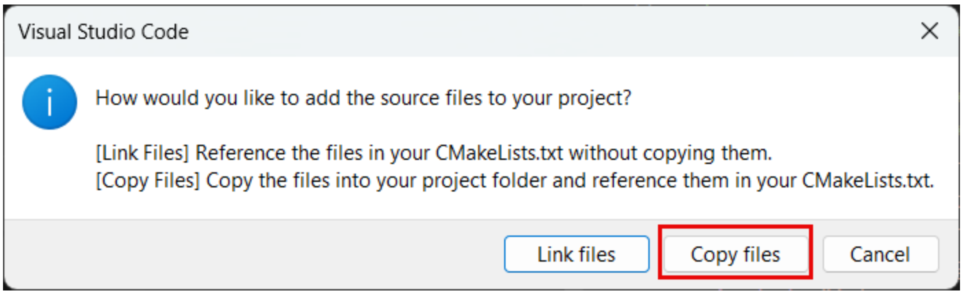


Figure 42. Copy files

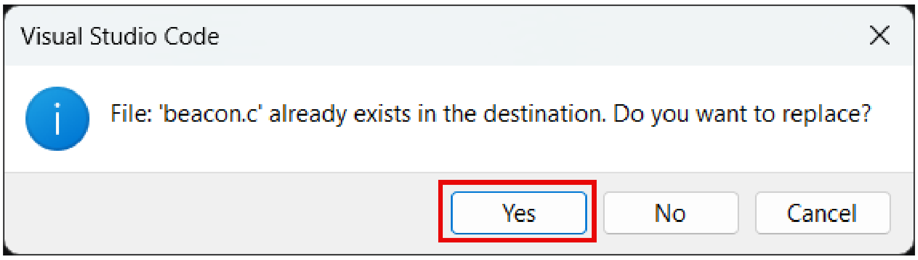


Figure 43. Replace the pre-existing files

4. On the kw47evk_beacon_freertos project, click **Pristine Build/Rebuild Project** and debug.

Figure 44. Perform a Pristine build/rebuild project

5. Open the NXP IoT Toolbox, select **Beacons**, and press Scan.

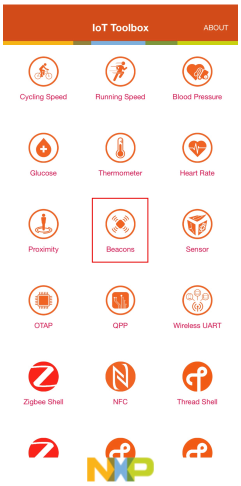


Figure 45. NXP IoT Toolbox

6. On the KW47-EVK board, to start advertising, press the SW2 button.

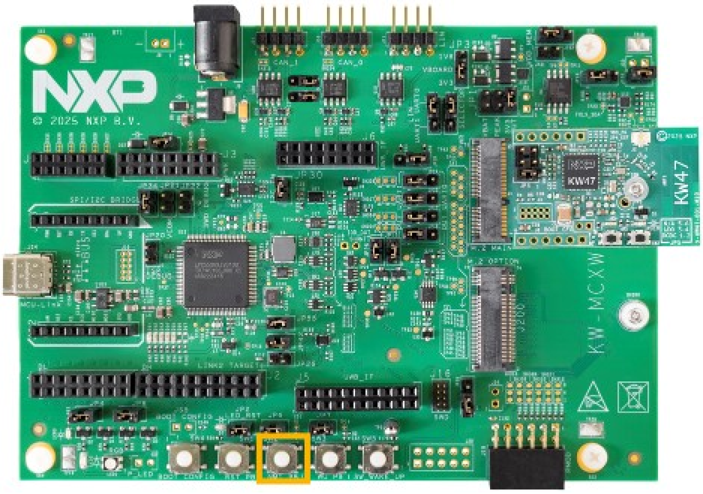


Figure 46. SW2 button

7. In the IoT Toolbox, the user must be able to see the counter increasing its value at every 5 seconds in the field **A**.

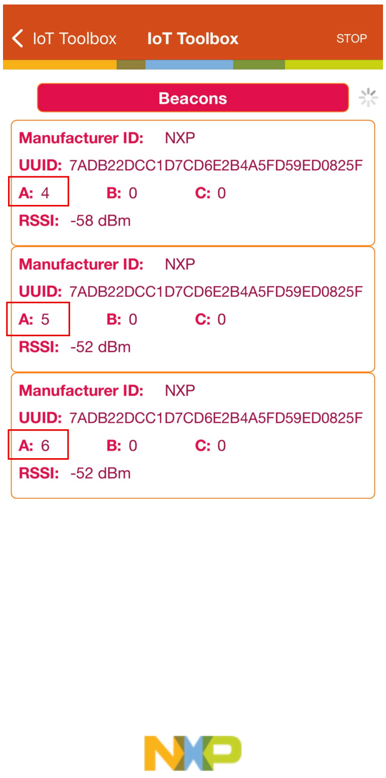


Figure 47. Counter value

For detailed instructions on testing the application and validating its functionality, refer to the KW45 - Updating Beacon Data community post. It provides step-by-step guidance on using the IoT Toolbox and interpreting beacon data.

The examples referenced in the Bluetooth LE Demo Applications user guide are compatible with KW47-EVK devices. Also, the examples can be migrated from KW45B41Z-EVK to KW47-EVK by applying project-specific modifications, such as those outlined in this section.

Note: *The required changes at the application level can vary depending on the specific use case.*

For more details on Bluetooth LE examples and modification, refer to the following guides:

- Bluetooth Low Energy Demo Applications User's Guide (BLEDAUG)
- Bluetooth Low Energy Application Developer's Guide (UG10184)

7 Acronyms

Table 8 lists the acronyms used in this document.

Table 8. Acronyms

Acronym	Description
AEC	Automotive Electronics Council
CAN	Control Area Network
CBC	Cipher Block Chaining
CCM	Continuous Conduction Mode
CMAC	Cipher-Based Message Authentication Code
CTR	Current Transfer Ratio
ECB	Electronic Code Book Mode
ECC	Elliptic-Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
FlexCAN	Flexible Control Area Network
FlexIO	Flexible Input/Output
FRO	Free-Running Oscillator
FSK	Frequency-Shift Keying
GCM	Galois Counter Mode
GFSK	Gaussian Frequency-Shift Keying
GMSK	Gaussian Minimum-Shift Keying
I2C	Inter-Integrated Circuit
ISP	In-System Programming
LDO	Low-Dropout Regulator
LIN	Local Interconnect Network
LPI2C	Low-Power Inter-Integrated Circuit
LPUART	Low-Power Universal Asynchronous Receiver Transmitter
MCU	Microcontroller Unit
MIPI	Mobile Industry Processor Interface
MSK	Minimum-Shift Keying
NBU	Narrowband Unit
OQPSK	Offset Quadrature Phase-Shift Keying
PMIC	Permanent Management-Integrated Circuit
PWM	Pulse Width Modulation

Table 8. Acronyms...*continued*

Acronym	Description
QFN	Quad Flat No Lead
RAM	Random Access Memory
RF	Radio Frequency
ROM	Read-Only Memory
RoTKTH	Root Key Table Hash
SDK	Software Development Kit
SMBus	System Management Bus
SP	Serial Port
SPI	Serial Peripheral Interface
SPSDK	Secure Provisioning SDK
SRAM	Static Random-Access Memory
SWD	Serial Wire Debug
UART	Universal Asynchronous Receiver Transmitter

8 References

Table 9 lists the references used to supplement this document.

Table 9. Related documentation/resources

Document	Link/how to access
KW47 Hardware Design Guide	UG10127
KW47 Reference Manual	KW47RM
KW47-EVK Board User Manual	UM12094
Power Management Hardware for the KW47	AN14709
KW45B41Z-EVK Board User Manual	KW45B41Z-EVKUM
Hardware Design Considerations for KW45B41Z and K32W148 Bluetooth LE Devices	AN13227
KW45/K32W148 - Power Management Hardware	AN13831
KW45 Reference Manual	KW45RM
Programming the KW45 flash for Application and Radio firmware via Serial Wire Debug during mass production	AN14003
MCUXpresso SDK Builder	MCUXpresso SDK Builder
Bluetooth Low Energy Demo Applications User's Guide	BLEDAUG
Bluetooth Low Energy Application Developer's Guide	UG10184
KW47 Knowledge Hub - NXP Community	KW47 Knowledge Hub
The best way to build a PCB first time right with KW47 (Automotive) or MCX W72 (IoT/Industrial) - NXP Community	The best way to build a PCB first time right with KW47 (Automotive) or MCX W72 (IoT/Industrial)
KW45 Knowledge Hub - NXP Community	KW45 Knowledge Hub
The best way to build a PCB first time right with KW45 (Automotive) or K32W1/MCXW71 (IoT/Industrial) - NXP Community	The best way to build a PCB first time right with KW45 (Automotive) or K32W1/MCXW71 (IoT/Industrial)
NXP Bootloader Host Application (blhost)	Bootloader Host Application (blhost)
MCUXpresso Secure Provisioning Tool	MCUXPRESSO-SECURE-PROVISIONING

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 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.

10 Revision history

Table 10 summarizes the revisions to this document.

Table 10. Revision history

Document ID	Release date	Description
AN14796 v.1.0	17 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.

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.

Suitability for use in automotive applications — This NXP product has been qualified for use in automotive applications. If this product is used by customer in the development of, or for incorporation into, products or services (a) used in safety critical applications or (b) in which failure could lead to death, personal injury, or severe physical or environmental damage (such products and services hereinafter referred to as "Critical Applications"), then customer makes the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, safety, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. As such, customer assumes all risk related to use of any products in Critical Applications and NXP and its suppliers shall not be liable for any such use by customer. Accordingly, customer will indemnify and hold NXP harmless from any claims, liabilities, damages and associated costs and expenses (including attorneys' fees) that NXP may incur related to customer's incorporation of any product in a Critical Application.

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.

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.

Amazon Web Services, AWS, the Powered by AWS logo, and FreeRTOS — are trademarks of Amazon.com, Inc. or its affiliates.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, μVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

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.

EdgeLock — is a trademark of NXP B.V.

J-Link — is a trademark of SEGGER Microcontroller GmbH.

Contents

Date of release: 17 April 2026
Document identifier: AN14796