AN12560

Migration Guide from MKW36A512xxx4 to MKW38A512xxx4

Rev. 2 — 30 June 2021 Application Note

Contents

1 Introduction

This document describes how to migrate from Kinetis MKW36A512xxx4 to MKW38A512xxx4 MCUs with emphasis on the connectivity software. In this document, the MKW36A512xxx4 and MKW38A512xxx4 devices are referred to as MKW36 and MKW38, respectively. The document is intended for software engineers, software testers, software integrators, and customers designing their own hardware.

2 Hardware considerations

The MKW36 wireless MCUs in 48-pin HVQFN packages are pin-to-pin compatible with MKW38, and almost all peripherals are the same on both devices. The main difference between the MKW36 and MKW38 is related to the radio.

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

Table 1. KW36/38 comparison

	KW36A	KW38A
Core	48-MHz Cortex-M0+	48-MHz Cortex-M0+
Memory (Flash/RAM)	512 KB with ECC/ 64 Kb	512 KB with ECC/ 64 Kb
Supply voltage (DCDC)	2.1 to 3.6 V	2.1 to 3.6 V
Radios	BT 5 8x Connections (1 Mbit/s) GFSK (250 k/500 k/1 Mbit/s)	BT 5 **LR (coded PHY), HS(2 Mbit/s),** 8x Connections(**125 k/500 k**/1M bit/s**/2 Mbit/s**) GFSK (250 k/500 k/1 Mbit/s**/2 Mbit/s**)
Radio Tx Power	+3.5 dBm at antenna connector (+5 dBm capable)	**+5 dBm** at antenna connector
Radio Sensitivity (Bluetooth LE Uncoded)	-95 dBm(1 Mbit/s) w/balun	**-98 dBm(1 Mbit/s) / -95.5 dBm(2 Mbit/s)** w/balun
Radio Sensitivity (Bluetooth LE Coded or Long-Range)	**NA**	**-105 dBm(125 Kbit/s) / -101 dBm(500 Kbit/s)**
Radio Sensitivity (GFSK, 250 Kbit/s-BT=0.5, H=0.5)	-99 dBm	**-101 dBm**
Radio Power (Rx/Tx)	6.3 mA/5.7 mA(0 dBm)	6.3 mA/5.7 mA(0 dBm)
Others	Radio flexibility (access to internal register) important to implement localization function, software support	Radio flexibility (access to internal register) important to implement localization function, software support

Table continues on the next page...

Table 1. KW36/38 comparison (continued)

		Enhanced Localization Support
Automotive Qualification	AEC-Q100 Grade 2	AEC-Q100 Grade 2

2.1 Peripherals instantiation

The KW36 devices in 48-pin HVQFN packages are pin-to-pin compatible with the KW38 devices, but the LPUART0 used for bootloader is not. KW36 uses PTC2(LPUART0_RX) and PTC18(LPUART0_TX), while KW38 uses PTC6(LPUART0_RX and PTC7(LPUART0_TX). For more details, see Chapter 11. "Kinetis Flashloader" in the *MKW36/35/34 Reference Manual* (document MKW36A512RM) and *MKW39/38/37 Reference Manual* (document MKW39A512RM).

The KW36 devices are also available in 40-pin “wettable” HVQFN packages. The bold alternatives are available only for the MKW36 devices and the alternatives in *italics* are only available for the MKW38 devices.

Table 2. MKW36/38 instance comparative

KW36-40 HVQFN	KW36-48 HVQFN	KW38-48 “HVQFN	**Pin Name**	**Default**	**ALT0**	**ALT1**	**ALT2**	**ALT3**	**ALT4**	**ALT5**	**ALT6**	**ALT7**	**ALT8**	**ALT9**
-	4	4	PTA16	DISABLED		PTA16/ LLWU_P4	SPI1_SOUT	LPUART1_ RTS_b		TPM0_CH0				
-	5	5	PTA17	DISABLED		PTA17/ LLWU_P5	SPI1_SIN	LPUART1_ RX	CAN0_TX	TPM_ CLKIN1				
-	6	6	PTA18	DISABLED		PTA18/ LLWU_P6	SPI1_SCK	LPUART1_ TX	CAN0_RX	TPM2_CH0				
-	7	7	PTA19	**DISABLED/ *ADC0_SE5***	ADC0_SE5	PTA19/ LLWU_P7	SPI1_PCS0	LPUART1_ CTS_b		TPM2_CH1				
-	41	41	PTC5	DISABLED		PTC5/ LLWU_P13/ RF_NOT_ALLOWED/		LPTMR0_ ALT2	LPUART0_ RTS_b	TPM1_CH1		**BSM_CLK**		

Table continues on the next page...

Table 2. MKW36/38 instance comparative (continued)

						RF_PRIORITY								
-	42	42	PTC6	DISABLED		PTC6/ LLWU_P14/ RF_ RFOSC_EN		I2C1_SCL	LPUART0_ RX	TPM2_CH0		**BSM_ FRAME**		
-	43	43	PTC7	DISABLED		PTC7/ LLWU_P15	SPI0_PCS2	I2C1_SDA	LPUART0_ TX	TPM2_CH1		**BSM_DATA**		
1	48	48	PTC19	PTC19		PTC19/ LLWU_P3/ RF_EARLY_ WARNING	SPI0_PCS0	I2C0_SCL	LPUART0_ CTS_b	**BSM_CLK**			LPUART1_ CTS_b	
2	1	1	PTA0	SWD_DIO		PTA0/ ***RF_ACTIVE***	SPI0_PCS1			TPM1_CH0		SWD_DIO		
3	2	2	PTA1	SWD_CLK		PTA1/ ***RF_STATUS***	SPI1_PCS0			TPM1_CH1		SWD_CLK		
14	16	16	PTB0	DIABLED		PTB0/ LLWU_P8/ RF_RFOSC_EN		I2C0_SCL/	CMP0_OUT	TMP0-CH1		CLKOUT	CAN0_TX	
15	17	17	PTB1	**DISABLED** ***ADC0_SE1/***	ADC0_SE1/ CMP0_IN5	PTB1/ RF_ PRIORITY	DTM_RX	I2C0_SDA	LPTMR0_ ALT1	TPM0_CH2		CMT_IRO	CAN0_RX	

Table continues on the next page...

Table 2. MKW36/38 instance comparative (continued)

				CMP0_IN5										
16	18	18	PTB2	**DISABLED** ***ADC0_SE3/*** ***CMP0_IN3***	ADC0_SE3/ CMP0_IN3	PTB2/ RF_NOT_ ALLOWED/ ***LLWU_P9***		DTM_TX	***TPM0_CH0***	TPM1_CH0			***TPM2_CH0***	
17	19	19	PTB3	**DISABLED** ***ADC0_SE2/*** ***CMP0_IN4***	ADC0_SE2/ CMP0_IN4	PTB3/ ERCLK32K/ ***RF_ACTIVE***	LPUART1_ RTS_b	***TPM0_CH1***	CLKOUT	TPM1_CH1		RTC_ CLKOUT	***TPM2_CH1***	
19	21	21	PTB16	EXTAL32K	EXTAL32K	PTB16	LPUART1_ RX	I2C1_SCL		TPM2_CH0				
20	22	22	PTB17	XTAL32K	XTAL32K	PTB17	LPUART1_ TX	I2C1_SDA		TPM2_CH1		**BSM_CLK**		
21	23	23	PTB18	NMI_b/ ***ADC0_SE4/*** ***CMP0_IN2***	ADC0_SE4/ CMP0_IN2	PTB18	LPUART1_ CTS_b	I2C1_SCL	TPM_ CLKIN0	TPM0_CH0		NMI_b		
33	37	37	PTC1	DISABLED		PTC1/ RF_EARLY_ WARNING	***ANT_B***	I2C0_SDA	LPUART0_ RTS_b	TPM0_CH2			SPI1_SCK	**BSM_CLK**
34	38	38	PTC2	DISABLED		PTC2/ LLWU_P10	TX_ SWITCH	I2C1_SCL	LPUART0_ RX	CMT_IRO		DTM_RX	SPI1_SOUT	**BSM_FRAME**
35	39	39	PTC3	DISABLED		PTC3/	RX_	I2C1_SDA	LPUART0_	TPM0_CH1		DTM_TX	SPI1_SIN	CAN0_TX

Table continues on the next page...

Table 2. MKW36/38 instance comparative (continued)

						LLWU_P11	SWITCH		TX					
36	40	40	PTC4	DISABLED		PTC4/ LLWU_P12/ **BLE_RF_ACTIVE**	***ANT_A***	EXTRG_IN	LPUART0_ CTS_b	TPM1_CH0		**BSM_DATA** ***I2C0_SCL***	SPI1_PCS0	CAN0_RX
38	45	45	PTC16	DISABLED		PTC16/ LLWU_P0/ RF_ STATUS	SPI0_SCK	I2C0_SDA	LPUART0_ RTS_b	TPM0_CH3			LPUART1_ RTS_b	
39	46	46	PTC17	DISABLED		PTC17/ LLWU_P1/ RF_EXT_ OSC_EN	SPI0_SOUT	I2C1_SCL	LPUART0_ RX	**BSM_ FRAME**		DTM_RX	LPUART1_ RX	
40	47	47	PTC18	DISABLED		PTC18/ LLWU_P2	SPI0_SIN	I2C1_SDA	LPUART0_ TX	**BSM_DATA**		DTM_TX	LPUART1_ TX	
41	49-64	49	Ground	NA										

NOTE

Table 2 is not a full description of the MKW36/38 pinout. For more details, see the *MKW36A/35A/34A DataSheet* and *MKW39/38/37 Data Sheet*. There is a change in the number of available digital pins between the 40-pin and 48-pin packages due to a different number of pins. For example, in the 48-pin package, there is a total of 25 digital pins. In the 40-pin package, there are 18 digital pins.

2.2 System memory map

Both devices contain various memories and memory-mapped peripherals which are located in the 4-GB memory space. Table 3 shows some peripheral locations within the memory map for the KW36 and KW38 devices.

Table 3. KW36/38 differences between system memory map

System 32-bit Address Range	Destination Slave		Access	
	KW36	KW38	KW36	KW38
0x0000_0000–0x07FF_FFFF	Program flash	Program flash	All masters	All masters
0x1400_0000 – 0x1400_1FFF	Programming Acceleration RAM	Programming Acceleration RAM	All masters	All masters
0x1FFF_C000 – 0x1FFF_FFFF	SRAM_L: Lower SRAM	SRAM_L: Lower SRAM	All masters	All masters
0x2000_0000 – 0x2000_BFFF	SRAM_U: Upper SRAM	SRAM_U: Upper SRAM	All masters	All masters
0x2001_8000–0x3FFF_FFFF	Reserved	Reserved	-	-
0x4000_0000–0x4007_FFFF	AIPS Peripherals	AIPS Peripherals	Cortex-M0+ core & DMA	Cortex-M0+ core & DMA
0x4008_0000–0x4008_FFFF	Reserved	**Radio (including BTLL, GFSK except RSIM)**	**-**	**Cortex-M0+ core & DMA**
0x4009_0000–0x400E_FFFF	Reserved	Reserved	-	-
0x400F_F000–0x400F_FFFF	General purpose input/output (GPIO)	General purpose input/output (GPIO)	Cortex-M0+ core & DMA	Cortex-M0+ core & DMA
0xF800_0000–0xFFFF_FFFF	IOPORT: GPIO (single cycle)	IOPORT: GPIO (single cycle)	Cortex-M0+ core	Cortex-M0+ core

NOTE

Table 3 does not contain the entire memory map. For more details, see the *MKW36A/35A/34A Data Sheet* and *MKW39/38/37 Data Sheet*.

2.3 NVIC configuration

NVIC configuration shows the differences between the KW36 and KW38 devices regarding the interrupt vector assignments. The vector number is the value stored in the stack when an interrupt is serviced and the IRQ number is non-core interrupt source count (which is the vector number minus 16).

Table 4. KW36/8 interrupt vector assignments

Address	Vector	IRQ	Source module		Source description	
			KW36	KW38	KW36	KW38
0x0000_0050	20	4	-	**Data stream**	Reserved for future MCM	**FIFO underrun, FIFO overflow, data ready, transfer complete and error**

2.4 Migration from KW35 series non-wettable flank package to KW38 series HVQFN48 wettable flank

If you have an older design based on KW35 series 48-pin non-wettable flank package, refer to this section.

KW35 series 48-pin non-wettable flank package pinout is shown in the figure below.

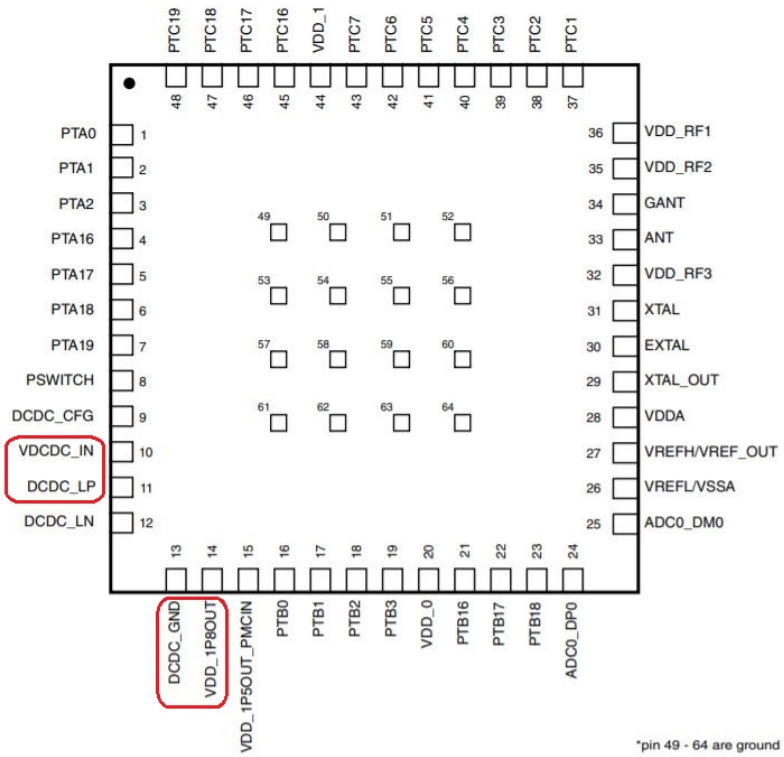

Figure 1. 48-pin non-wettable flank package pinout

KW38 series 48-pin HVQFN48 wettable flank pinout is shown in the figure below.

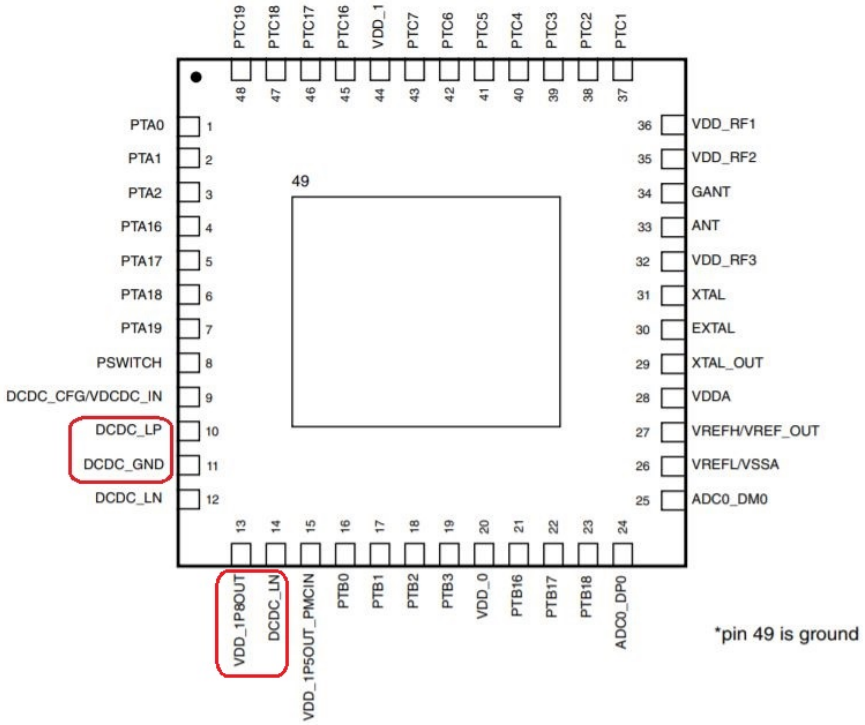

Figure 2. 48-pin HVQFN48 wettable flank pinout

KW35 non-wettable DCDC pins Layout is shown in the figure below.

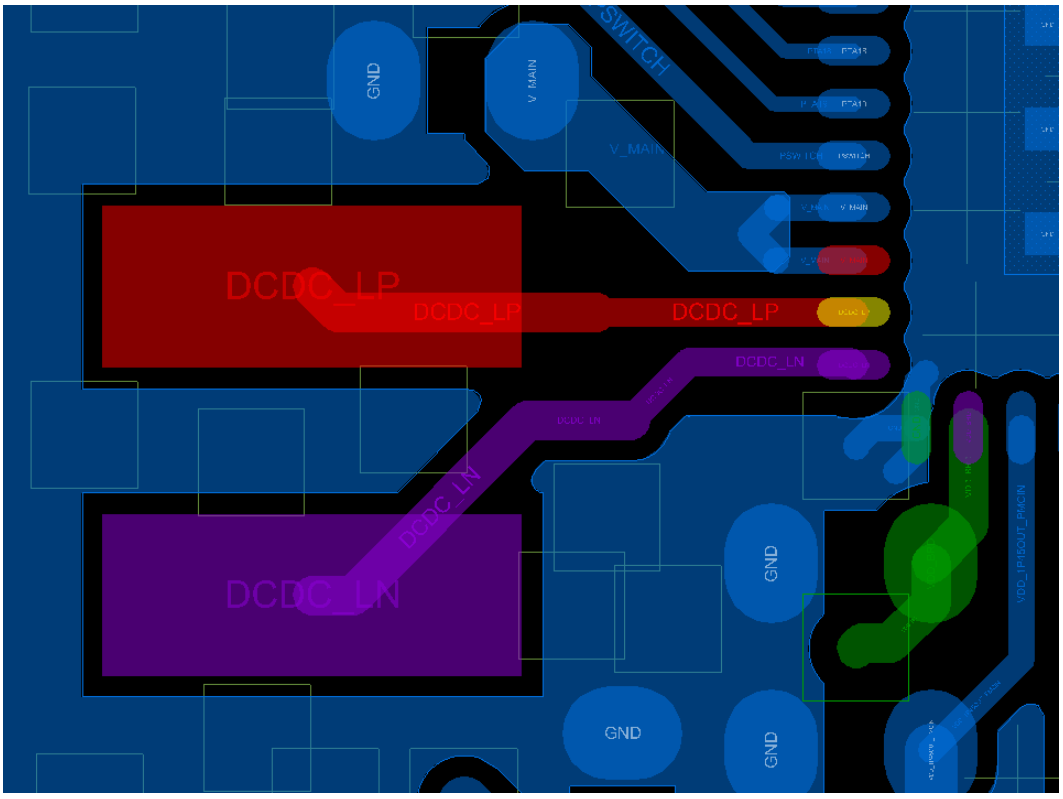

Figure 3. KW35 non-wettable DCDC pins layout

KW38 wettable DCDC pins Layout is shown in the figure below.

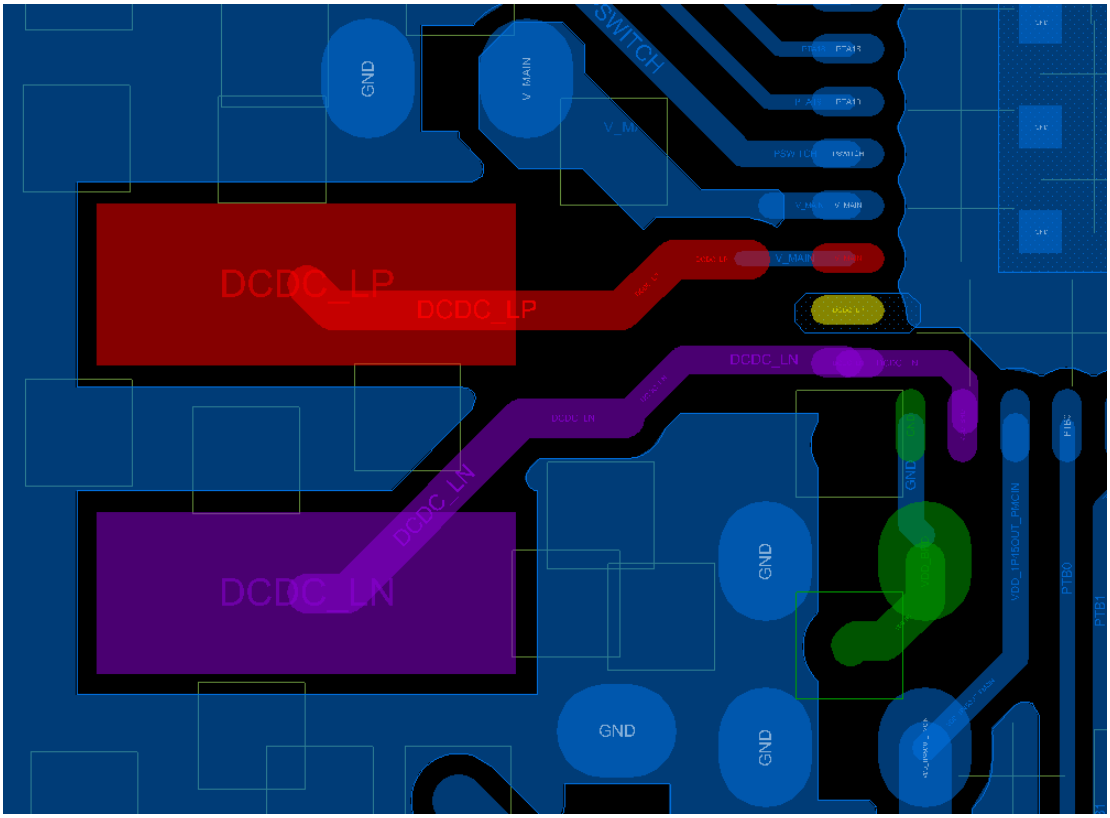

Figure 4. KW38 wettable DCDC pins layout

3 Software development kit download and install

This chapter shows how to download the Software Development Kit (SDK) for the MKW36A512xxx4 and MKW38A512xxx4 devices. The steps to download the SDK package for the KW36 devices are as follows:

1. Go to the MCUXpresso web page (mcuxpresso.nxp.com).
2. Log in with your registered account.
3. Search for the FRDM-KW36 device. Click "FRDM-KW36" in the "Boards" tab and then click “Build MCUXpresso SDK”.

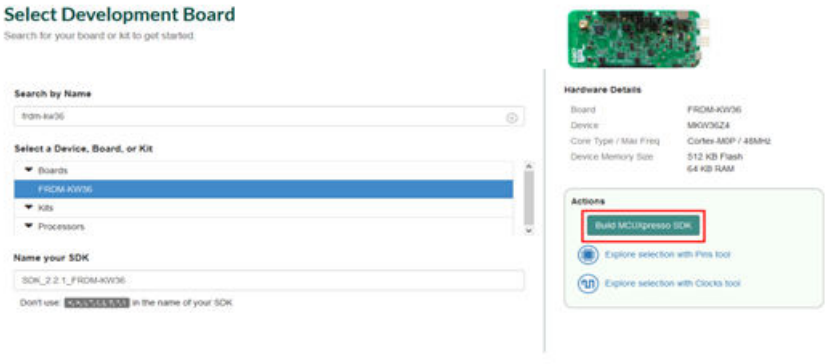

Figure 5. Building MKW36 SDK package

4. The next page is displayed. Select “All toolchains” in the “Toolchain / IDE” box and provide a name to identify the package.

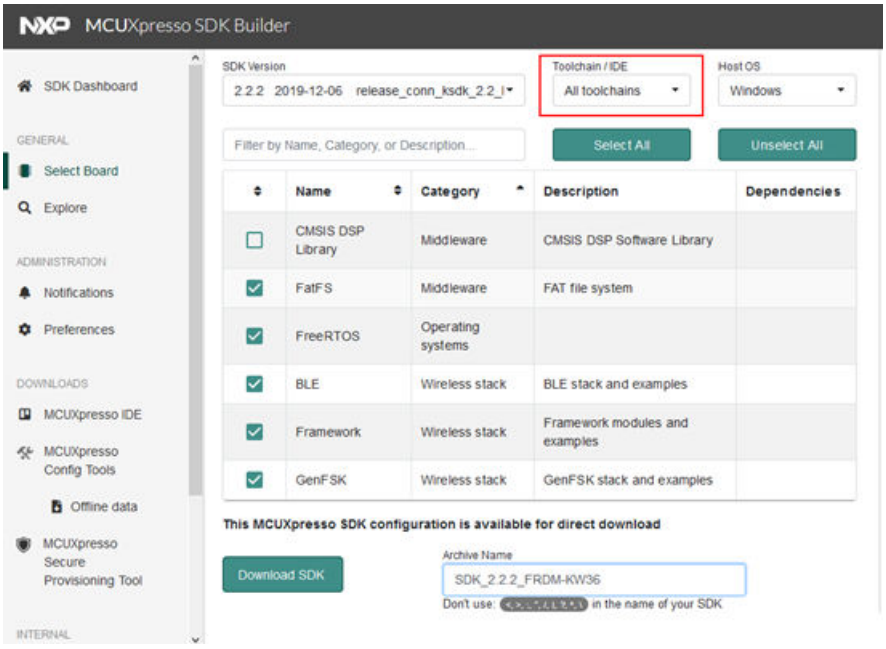

Figure 6. Selecting version of SDK

5. Click the “Download SDK” button. This starts the building process of the desired SDK. It takes a few minutes until the system gets the package into your profile at MCUXpresso web page.
6. When the SDK is ready to be downloaded, the “Software Terms and Conditions” are displayed. Accept them and the download process starts automatically.
7. If the download does not start automatically, click “Download SDK Archive”.

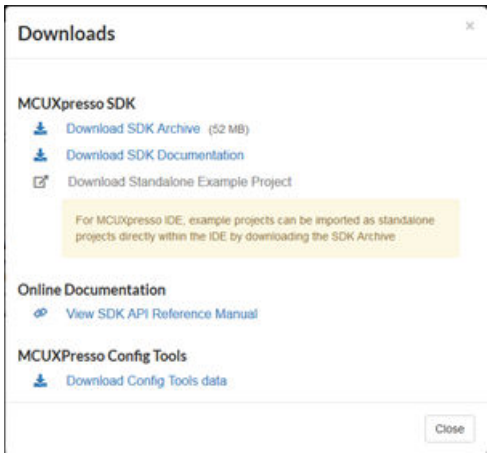

Figure 7. MKW36 SDK displayed in Dashboard

8. If the above picture is not displayed, click the “Download SDK archive and documentation” button in the “MCUXpresso SDK Dashboard”.

Figure 8. MKW36 SDK displayed in Dashboard

Now you have downloaded the SDK package for MKW36A devices. To download the SDK for MKW38A devices, repeat all the above steps substituting “FRDM-KW36” with “FRDM-KW38”.

Now both SDKs are downloaded.

NOTE

The following steps are applicable only if the KW36 Bluetooth LE stack version number and Framework version number are at least equal to 1.3.5. and 5.4.5, respectively.

4 Software migration in IAR Embedded Workbench IDE

This chapter shows how to migrate MKW36 example code to the MKW38 devices in the IAR Embedded Workbench IDE. The Heart Rate Sensor project is used as a base in this document, because it is an easy-to-understand example and involves the Bluetooth LE connectivity software stack (included in the SDK).

4.1 Changes required in project options and settings

NOTE

In this section, the “bare-metal” version of the project is used. However, the same steps apply for FreeRTOS projects. Some paths related to the “bare-metal” projects may differ when using FreeRTOS versions.

1. Copy the KW36 heart rate sensor project located at *<KW36 SDK_root>/boards/frdmkw36/wireless_examples/bluetooth/hrs* into the wireless examples folder of the MKW38 SDK *<KW38SDK_root>/boards/frdmkw38/wireless_examples/bluetooth*.

NOTE

Rename the KW36 heart rate project when copying it into the KW38 SDK. There is also a project named *hrs*. In this document, it is renamed to *hrs_migr*. It is used just as an example on how to migrate a KW36 project to KW38.

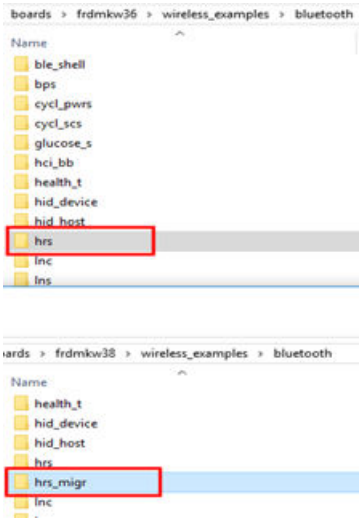

Figure 9. Copy the KW36 project into KW38 SDK

2. After the KW36 project is copied into KW38 SDK examples, open the *hrs_bm.ewp* file in a text editor. It is located in the *<KW38SDK_root>/boards/frdmkw38/wireless_examples/bluetooth/hrs_migr/bm/iar/* folder.
    a. Replace all references to *framework_5.4.x* with *framework*.
    b. Replace all references to *bluetooth_1.3.x* with *bluetooth*.
    c. Save the changes.

    NOTE

    The Framework and Bluetooth versions may differ depending on the KW36 SDK version. To find which versions are used, check the *SW-Content-Register.txt* file in the KW36 SDK folder or see the versions directly in the *hrs_bm.ewp* file.

3. Open the *hrs_bm.eww* project. It is located in the *<KW38SDK_root>/boards/frdmkw38/wireless_examples/bluetooth/hrs_migr/bm/iar/* folder.
    a. Press *ALT + F7* to open the project options.
    b. In the *C/C++ Compiler-> Preprocessor* window, change the references to KW36, as specified in Table 5.

Table 5. Changes

Original version	Changes to made
$PROJ_DIR$/../../../../../../../devices/MKW36Z4/drivers	*$PROJ_DIR$/../../../../../../../devices/MKW38A4/drivers*
$PROJ_DIR$/../../../../../../../middleware/wireless/framework/LowPower/Interface/MKW36Z	*$PROJ_DIR$/../../../../../../../middleware/wireless/framework/LowPower/Interface/MKW38Z4*
$PROJ_DIR$/../../../../../../../middleware/wireless/framework/DCDC/Interface/MKW36Z	*$PROJ_DIR$/../../../../../../../middleware/wireless/framework/DCDC/Interface/MKW38Z4*
$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW36Z4	*$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW38Z4/drv* *$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW38Z4/drv/nb2p4ghz*

Table continues on the next page...

Table 5. Changes (continued)

	$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW38Z4/drv/nb2p4ghz/configs/gen35
$PROJ_DIR$/../../../../../../../devices/MKW36Z4	*$PROJ_DIR$/../../../../../../../devices/MKW38A4*
$PROJ_DIR$/../../../../../../../devices/MKW36Z4/utilities	*$PROJ_DIR$/../../../../../../../devices/MKW38A4/utilities*
$PROJ_DIR$/../../../../../../../middleware/wireless/bluetooth/controller/interface	*$PROJ_DIR$/../../../../../../../middleware/wireless/ble_controller/interface* *$PROJ_DIR$/../../../../../../../middleware/wireless/ble_controller/config*

c. In the *C/C++ Compiler-> Preprocessor* window, change the path to the pre-include file. The new pre-include file should be *$PROJ_DIR$/../../../../../../../boards/frdmkw38/wireless_examples/bluetooth/hrs_migr/bm/app_preinclude.h*.

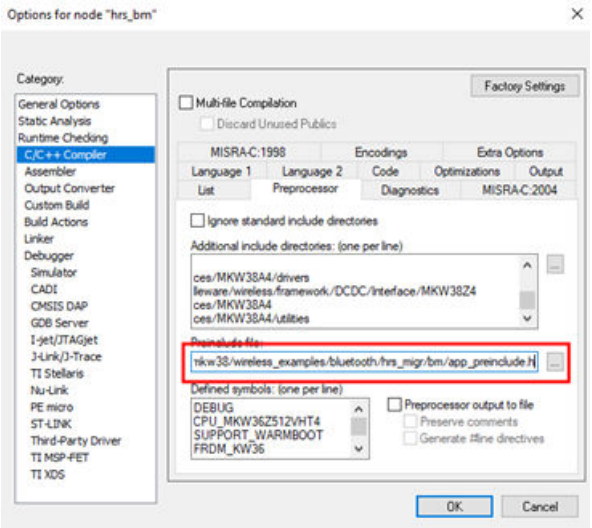

Figure 10. Change path to pre-include file

d. In the *Defined symbols* text box, modify "CPU_MKW36Z512VHT4" to "CPU_MKW38A512VFT4" and "FRDM_KW36" to "FRDM_KW38". Delete "FREEDOM". Add "CR_INTEGER_PRINTF", "ENABLE_RAM_VECTOR_TABLE=1" and "CFG_BLE_PRJ=1".

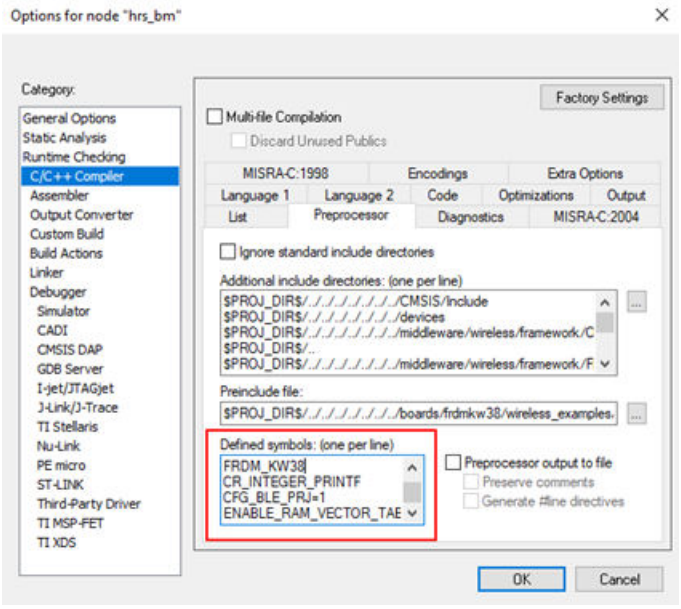

Figure 11. Defined symbols text box

e. Save the workspace ("File -> Save workspace").

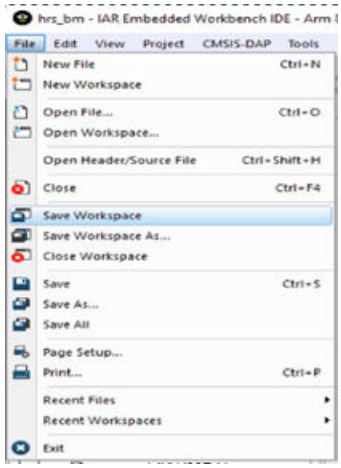

Figure 12. Saving workspace

4. Copy the *MKW38A512xxx4_PD_connectivity.icf* file, which is located at *<KW38SDK_root>/middleware/wireless/framework/Common/devices/MKW38A4/iar*, into the project in the *iar* folder located at *<KW38SDK_root>/boards/frdmkw38/wireless_examples/bluetooth/hrs_migr/bm/iar/*.
5. Delete the *MKW36Z512xxx4_PD_connectivity.icf* file from the folder metioned above.
6. Press *ALT + F7* to open the project options.
7. In the "Linker->Config" window, change the linker configuration file to *$PROJ_DIR$\MKW38A512xxx4_PD_connectivity.icf*.

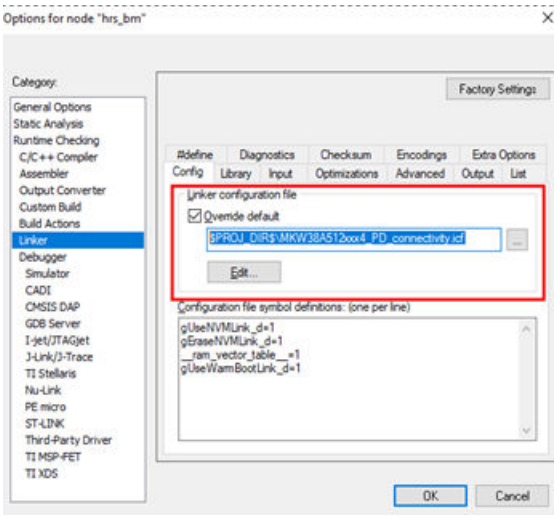

Figure 13. Linker configuration file

8. In "Linker->Library", change the additional library from *$PROJ_DIR$/../../../../../../../middleware/wireless/bluetooth/controller/lib/lib_ble_kw36z_controller_iar.a* to *$PROJ_DIR$/../../../../../../../middleware/wireless/ble_controller/lib/lib_ble_kw38a4_controller.a*.

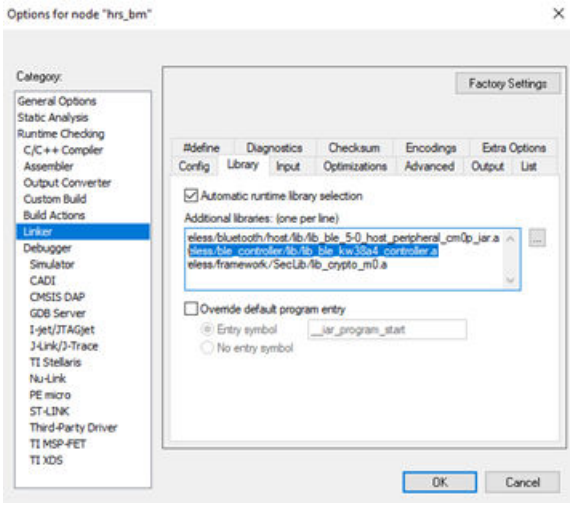

Figure 14. Additional libraries

9. Click "OK" and save the workspace.
10. Press "ALT + F7" to open the options. In the "General Options -> Target" tab, change "Device". Click the icon at the right-hand side of the "Device" textbox and select "NXP -> KinetisKW -> KW3x -> NXP MKW38A512xxx4".

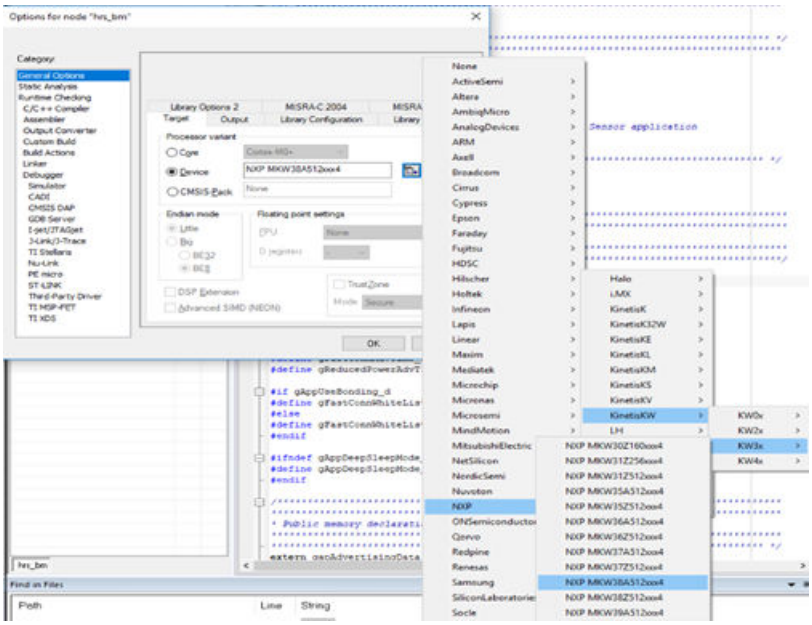

Figure 15. Change device

11. Click "OK" in the "Options" tab and then save the workspace.

NOTE

The above changes are made only for the current configuration. The default configuration is "Debug". To change the "Release" configuration, select "Release" in "Workspace" and repeat steps 3-10.

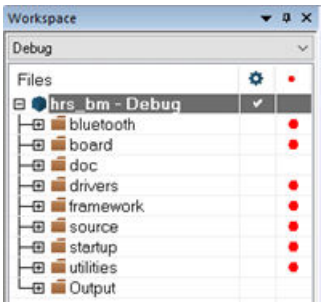

Figure 16. Change configuration

12. Open the *hrs_bm.ewp* file in the *<KW38SDK_root>/boards/frdmkw38/wireless_examples/bluetooth/hrs_migr/bm/iar/project* folder in a text editor.
    a. Replace all references to "devices\MKW36Z4" with "devices\MKW38A4".
    b. Replace all references to "MKW36Z4" with "MKW38Z4".
    c. Replace all references to "MKW36Z" with "MKW38Z4".
13. When you go back to the IAR project, a window warns you that the *hrs_bm.ewp* file was modified and you are asked if you would like to reload the project. Click "Yes to All".
14. Add these groups and files into the heart rate sensor project:
    a. Expand the *bluetooth* folder, select the *controller* folder, right-click, select "Add -> Add Group" and add the *config* group.

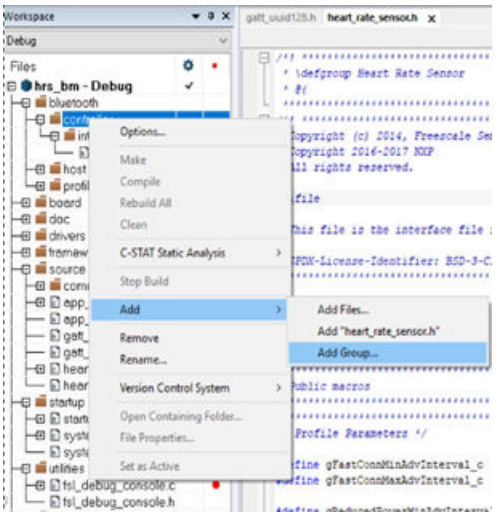

Figure 17. Adding a group

b. Select the *config* folder, right-click, select "Add -> Add Files" and add the *ble_controller_task_config.h*, *ble_ll_globals.h*, and *ble_ll_globals.h*" files at *<KW38SDK_root>/middleware/wireless/ble_controller/config/*.

c. Expand the "interface" group from the *controller* folder. Select *controller_interface.h*, right-click, and click "Remove". Select the *interface* folder and add the following files: *controller_interface.h* from *<KW38SDK_root>/middleware/wireless/ble_controller/interface* and *controller_init.c* from *<KW38SDK_root>/middleware/wireless/ble_controller/src/MKW38/*.

d. Select the *config* folder, right-click, select "Add -> Add Files" and add *ble_controller_task.c* from *<KW38SDK_root>/middleware/wireless/ble_controller/src*.

e. Select the *bluetooth* folder and add the "hci_transport" group. Select the *hci_transport* folder and add the "interface" and "source" groups.

f. Select the *interface* folder and add the *hci_transport.h* file from *<KW38SDK_root>/middleware/wireless/bluetooth/hci_transport/interface/*.

g. Select the *source* folder and add the *hcit_serial_interface.c* file from *<KW38SDK_root>/middleware/wireless/bluetooth/hci_transport/source/*.

h. The *bluetooth* folder should now have the following structure:

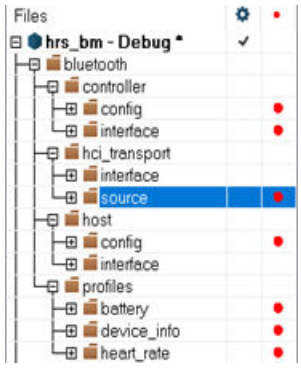

Figure 18. bluetooth folder structure

i. Expand the *drivers* folder and remove the following files: *fsl_i2c.h, fsl_i2c.c*, and *fsl_flash.c*. Add the following files: *fsl_ftfx_controller.c, fsl_ftfx_controller.h, fsl_ftfx_flash.c, fsl_ftfx_flash.h, "sl_ftfx_flexnvm.c*, and *fsl_ftfx_flexnvm.h.* They are located at: *<KW38SDK_root>/devices/MKW38A4/drivers*.

j. Expand the *framework* folder:

- Expand the *SerialManager* folder, expand the *Source* folder, select the *I2C_Adapter.c*, *I2C_Adapter.h*, *UART_Adapter.c*, and *UART_Adapter.h* files, and click *Remove*.
  - Select the *SPI_Adapter* folder, remove the *SPI_Adapter.c* and *SPI_Adapter.h* files, and add *SPI_Serial_Adapter.c*and *SPI_Serial_Adapter.h*, located at *<KW38SDK_root>/middleware/wireless/framework/SerialManager/Source/SPI_Adapter/*.
  - Select the *Source* folder and add the *UART_Serial_Adapter.c* and *UART_Serial_Adapter.h* files located at *<KW38SDK_root>/middleware/wireless/framework/SerialManager/Source/*.
- Expand the *XCVR* folder, then the *MKW38Z4* folder, select all the files inside, and remove them. Then:
  - Select the *MKW38Z4* folder and add the "nb2p4ghz" group. Select the *nb2p4ghz* folder and add the "configs" group.
  - Select the *configs* folder and add all files from the *<KW38SDK_root>/middleware/wireless/framework/XCVR/MKW38Z4/drv/nb2p4ghz/configs/gen35* folder.
  - Select the *nb2p4ghz* folder and add all files from the *<KW38SDK_root>/middleware/wireless/framework/XCVR/MKW38Z4/drv/nb2p4ghz* folder.
  - Select the *MKW38Z4* folder and add all files from the *<KW38SDK_root>/middleware/wireless/framework/XCVR/MKW38Z4/drv/* folder.

k. Expand the *source->common* folder and remove the *ble_controller_task.c* and *ble_controller_task_config.h* files.

l. Expand the *startup* folder, select all the files inside it, and remove them. Add the *startup_MKW38A4.s* file from the *<KW38SDK_root>/devices/MKW38A4/iar/* folder.

m. Select the "hrs_bm" project and add the "device" group. Select the *device* folder and add the *"sl_device_registers.h, MKW38A4_features.h, MKW38A4.h, system_MKW38A4.c,*and *system_MKW38A4.h* files located at *<KW38SDK_root>/devices/MKW38A4*.

n. Select the "hrs_bm" project and add the "components" group. Select the "component" group and add the "lists", "serial_manager", and "uart" groups. Select each folder and add all the files located in the corresponding folders from *<KW38SDK_root>/components/*.

o. Select the *utilities* folder, remove all the files inside it, and add the *fsl_str.c,* and *fsl_str.h* files from *<KW38SDK_root>/devices/MKW38A4/utilities/str/,* the *fsl_assert.c* file from *<KW38SDK_root>/devices/MKW38A4/utilities/*, and the *fsl_debug_console.c, fsl_debug_console.h,* and *fsl_debug_console_conf.h* files from *<KW38SDK_root>/devices/MKW38A4/utilities/debug_console/*.

15. Press "ALT + F7" to open the project options. In the "C/C++ Compiler -> Preprocessor" window and "Additional include directories" textbox, add the following lines:
    - *$PROJ_DIR$/../../../../../../../middleware/wireless/bluetooth/hci_transport/interface*
    - *$PROJ_DIR$/../../../../../../../components/uart*
    - *$PROJ_DIR$/../../../../../../../components/lists*
    - *$PROJ_DIR$/../../../../../../../components/serial_manager*
    - *$PROJ_DIR$/../../../../../../../devices/MKW38A4/utilities/str*
    - *$PROJ_DIR$/../../../../../../../devices/MKW38A4/utilities/debug_console*
16. Save the workspace.

4.2 Changes required at application level

- Open the *board.h* file located in the *board* folder in the workspace:

— Include the *EmbeddedTypes.h* file:

```
 #include "EmbeddedTypes.h"
```

— Change the board name from "FRDM-KW36" to "FRDM-KW38":

```
#define BOARD_NAME "FRDM-KW38"
```

— Add the following debug macros in the "Definitions" section:

```
#ifndef BOARD_DBGINITSET
#define BOARD_DBGINITSET(__x, __y)
#endif

#ifndef BOARD_DBGINITDBGIO
#define BOARD_DBGINITDBGIO()
#endif

#ifndef BOARD_DBGAPPIOSET
#define BOARD_DBGAPPIOSET(__x, __y)
#endif

#ifndef BOARD_DBGTOGGLEDBGIO
#define BOARD_DBGTOGGLEDBGIO()
#endif

#ifndef BOARD_DBGCONFIGINIT
#define BOARD_DBGCONFIGINIT(__x)
#endif

#ifndef DBG_LOG_DUMP
#define DBG_LOG_DUMP(__x)
#endif
```

— Declare the following functions:

```
void BOARD_RTC_Init(void);
void BOARD_RTC_Deinit(void);

extern void BOARD_SetCoreClock48Mhz(void);
extern void BOARD_ResetCoreClock(void);

extern uint8_t BOARD_GetXtal32MhzTrim(bool_t regRead);
extern void BOARD_SetXtal32MHzTrim(uint8_t trimValue, bool_t saveToHwParams);
```

- Open the *board.c* file located in the *board* folder in the workspace:

— Add the next define:

```
#define BOARD_32MHZ_XTAL_TRIM_DEFAULT 0x4BU
```

— Add the following variable definition:

```
static uint8_t Xtal32MhzTrim = BOARD_32MHZ_XTAL_TRIM_DEFAULT;
```

— Remove `static const uint8_t mXtalTrimDefault = 0x36;`.

— In the hardware_init() function, change:

```
if(0xFFFFFFFF == gHardwareParameters.xtalTrim)
{
 gHardwareParameters.xtalTrim = mXtalTrimDefault;
 }
to
if(0xFFFFFFFF != gHardwareParameters.xtalTrim)
{
 Xtal32MhzTrim = (uint8_t)gHardwareParameters.xtalTrim;
}
```

— Add the following function definitions:

```
void BOARD_RTC_Init(void)
{
 SIM->SCGC6 |= SIM_SCGC6_RTC_MASK;

 if ((RTC->CR & RTC_CR_OSCE_MASK) == 0u)
 {
 uint16_t rtcCRMask;
 /* RTC_CR: SC2P=0,SC4P=0,SC8P=0,SC16P=0 */
 rtcCRMask = (uint16_t)~(RTC_CR_SC2P_MASK | RTC_CR_SC4P_MASK | RTC_CR_SC8P_MASK |
RTC_CR_SC16P_MASK);
RTC->CR &= (uint32_t)rtcCRMask;
 /* RTC_CR: OSCE=1 */
 RTC->CR |= RTC_CR_OSCE_MASK;
 }
}

void BOARD_RTC_Deinit(void)
{
if((SIM->SCGC6 & (uint32_t)SIM_SCGC6_RTC_MASK) != 0U)
 {
 /* switch off 32kHz oscillator */
 RTC->CR &= ~RTC_CR_OSCE_MASK;
 }
}

void BOARD_SetCoreClock48Mhz(void)
{
 /* Set core clock to 48Mhz */
 MCG->C4 |= MCG_C4_DRST_DRS(1) | MCG_C4_DMX32(1);
}

void BOARD_ResetCoreClock(void)
{
 /* Set core clock to default clock (20-25MHz) */
 MCG->C4 &= (uint8_t)(~(MCG_C4_DRST_DRS(1) | MCG_C4_DMX32(1)));
}

uint8_t BOARD_GetXtal32MhzTrim(bool_t regRead)
{
 uint8_t retVal;

 if (TRUE == regRead)
 {
 /* get the XTAL trim value from XCVR reg */
 retVal = (uint8_t)((RSIM->ANA_TRIM &
RSIM_ANA_TRIM_BB_XTAL_TRIM_MASK)>>RSIM_ANA_TRIM_BB_XTAL_TRIM_SHIFT);
```

```
 }
 else
 {
 /* get the XTAL trim value from HW params */
 retVal = Xtal32MhzTrim;
 }

 return retVal;
}

void BOARD_SetXtal32MHzTrim(uint8_t trimValue, bool_t saveToHwParams)
{
 uint32_t temp;

 assert((trimValue & 0x80U) == 0U); /* High bit must not be set */
 /* Apply a trim value to the crystal oscillator */
 temp = RSIM->ANA_TRIM;
 temp &= ~(RSIM_ANA_TRIM_BB_XTAL_TRIM_MASK);
 RSIM->ANA_TRIM = temp | RSIM_ANA_TRIM_BB_XTAL_TRIM(trimValue);

 if ((TRUE == saveToHwParams))
 {
 hardwareParameters_t hwParams;

 /* write new XTAL trim value into hardware params structure */
 (void)NV_ReadHWParameters(&hwParams);
 hwParams.xtalTrim = (uint32_t)trimValue;
 (void)NV_WriteHWParameters(&hwParams);

 /* update the local variable that holds the XTAL trim value */
 Xtal32MhzTrim = trimValue;
 }
}
```

- Open the *app_preinclude.h* file located in the *source* folder in the workspace and do the following changes:
  - Delete the definitions of *gXcvrDacTrimValueSorageAddr_d* and *gPreserveXcvrDacTrimValue_d*

```
#define gXcvrDacTrimValueSorageAddr_d ((uint32_t)FREESCALE_PROD_DATA_BASE_ADDR + 1040)
#define gPreserveXcvrDacTrimValue_d 1
```

- Add the *Link Layer pool* configuration to the memory pools. Change:

```
#define PoolsDetails_c \
 AppPoolsDetails_c
to
#define PoolsDetails_c \
 AppPoolsDetails_c \
 LlPoolsDetails_c
```

- Configure the *LlMem pool* by adding the following lines:

```
#ifndef gLlMemPoolId_c
/* If define is not set by application, use a common pool for app/host and LL. */
#define gLlMemPoolId_c 0
#else /* gLlMemPoolId_c */
/* Application set the flag, make sure it is valid. */
#if (gLlMemPoolId_c > 1)
#error Please select pool 0 or pool 1
#endif /* (gLlMemPoolId_c > 1) */
```

```
#endif /* gLlMemPoolId_c */

#if defined(gLlUsePeriodicAdvertising_d)
 /* check compile switch incompatibilities */
 #if defined(gAppExtAdvEnable_d)
 #if ((gAppExtAdvEnable_d == 0) && (gLlUsePeriodicAdvertising_d == 1))
 #error Compile switch incompatibility! gLlUsePeriodicAdvertising_d=1 shall not be used with
gAppExtAdvEnable_d=0
 #endif /* ((gAppExtAdvEnable_d == 0) && (gLlUsePeriodicAdvertising_d == 1)) */
 #else
 /* Periodic advertising support needs extended advertising support. */
 #if (gLlUsePeriodicAdvertising_d == 1)
 #define gAppExtAdvEnable_d 1
 #endif /* (gLlUsePeriodicAdvertising_d == 1) */
 #endif /*defined(gAppExtAdvEnable_d)*/
#endif /* (defined(gLlUsePeriodicAdvertising_d)) */

#if (defined(gLlScanPeriodicAdvertiserListSize_c) && !defined(gLlScanAdvertiserListSize_c))
 #if (gLlScanPeriodicAdvertiserListSize_c != 0)
 #define gLlScanAdvertiserListSize_c (26-gLlScanPeriodicAdvertiserListSize_c)
 #endif /* (gLlScanPeriodicAdvertiserListSize_c != 0) */
#endif /* (defined(gLlScanPeriodicAdvertiserListSize_c) && !defined(gLlScanAdvertiserListSize_c))
*/

#if (!defined(gAppExtAdvEnable_d))
 #define gAppExtAdvEnable_d 0
#endif /* (!defined(gAppExtAdvEnable_d)) */

/* Defines L1Mem pools by block size and number of blocks. Must be aligned to 4 bytes.*/
#if (gAppExtAdvEnable_d == 0)
 /*Large size events (<= 72 bytes).*/
 #define gLlBufferNbrLargeSizeEvent_c (4) //BT_FW_LE_EVENT_TYPE1_BUFFERS
 /*Medium size events (<= 32 bytes).*/
 #define gLlBufferNbrMediumSizeEvent_c (4) //BT_FW_LE_EVENT_TYPE2_BUFFERS
 /*Small size events (<= 12 bytes).*/
 #define gLlBufferNbrSmallSizeEvent_c (6) //BT_FW_LE_EVENT_TYPE3_BUFFERS
 /*Generic events (<= 72 bytes).*/
 #define gLlBufferGenericSizeEvent_c (4) //BT_FW_LE_EVENT_TYPE4_BUFFERS

 #define gLlCmdBuffer80Bytes_c (1)

 /*If extended advertising is not set, use legacy settings for advertising*/
 #ifndef gLlMaxUsedAdvSet_c
 #define gLlMaxUsedAdvSet_c 1
 #endif
 #ifndef gLlMaxExtAdvDataLength_c
 #define gLlMaxExtAdvDataLength_c 31
 #endif
 #ifndef gLlUsePeriodicAdvertising_d
 #define gLlUsePeriodicAdvertising_d 0
 #endif

#if (gLlMemPoolId_c == 1)
#ifndef LlPoolsDetails_c
 #define LlPoolsDetails_c \
 _block_size_ 32 _number_of_blocks_
(gLlBufferNbrSmallSizeEvent_c+gLlBufferNbrMediumSizeEvent_c+((3+4)*gAppMaxConnections_c))
_pool_id_(1) _eol_ \
 _block_size_ 64 _number_of_blocks_ ((2*gAppMaxConnections_c)) _pool_id_(1) _eol_ \
 _block_size_ 80 _number_of_blocks_
```

```
(gLlBufferNbrLargeSizeEvent_c+gLlCmdBuffer80Bytes_c+gLlBufferGenericSizeEvent_c) _pool_id_(1)
_eol_ \
 _block_size_ 268 _number_of_blocks_ (gLlBufferNbrTxAclPkts+gLlBufferNbrRxAclPkts) _pool_id_(1)
_eol_
#endif /* LlPoolsDetails_c */
#else /* (gLlMemPoolId_c == 1) */
#ifdef LlPoolsDetails_c
#error Single pool is used, please do not define LlPoolsDetails_c in app_preinclude.h
#else
 #define LlPoolsDetails_c
#endif /* LlPoolsDetails_c */
#endif /* (gLlMemPoolId_c == 1) */
#else /* (gAppExtAdvEnable_d == 0) */
 /*Large size events (<= 288 bytes).*/
 #define gLlBufferNbrLargeSizeEvent_c (10) //BT_FW_LE_EVENT_TYPE1_BUFFERS
 /*Medium size events (<= 128 bytes).*/
 #define gLlBufferNbrMediumSizeEvent_c (16) //BT_FW_LE_EVENT_TYPE2_BUFFERS
 /*Small size events (<= 64 bytes).*/
 #define gLlBufferNbrSmallSizeEvent_c (12) //BT_FW_LE_EVENT_TYPE3_BUFFERS
 /*Generic events (<= 72 bytes).*/
 #define gLlBufferGenericSizeEvent_c (6) //BT_FW_LE_EVENT_TYPE4_BUFFERS

 #define gLlCmdBuffer288Bytes_c (1)

/*
 * Default configuration for LlPoolsDetails_c
 *
 * The 128 bytes buffer pool has been changed to a 80 bytes buffer pool: 128 byte buffer for
events whose size is > 64 & < 128 is not a good size
 * because there are three events in this range with size = 72 (HCI_READ_LOCAL_SUPPORTED_COMMANDS
& HCI_LE_READ_LOCAL_P256_PUBLIC_KEY) and
 * size = 74 (HCI_VENDOR_DTM_RX_PKT_EVENT)
 * => take 80 bytes buffer size instead
 */
#if (gLlMemPoolId_c == 1)
#ifndef LlPoolsDetails_c
 #define LlPoolsDetails_c \
 _block_size_ 32 _number_of_blocks_ ((3+4)*gAppMaxConnections_c) _pool_id_(1) _eol_ \
 _block_size_ 64 _number_of_blocks_ (gLlBufferNbrSmallSizeEvent_c+(2*gAppMaxConnections_c))
_pool_id_(1) _eol_ \
 _block_size_ 80 _number_of_blocks_ (gLlBufferGenericSizeEvent_c) _pool_id_(1) _eol_ \
 _block_size_ 128 _number_of_blocks_ (gLlBufferNbrMediumSizeEvent_c) _pool_id_(1) _eol_ \
 _block_size_ 268 _number_of_blocks_ (gLlBufferNbrTxAclPkts+gLlBufferNbrRxAclPkts) _pool_id_(1)
_eol_
 _block_size_ 288 _number_of_blocks_ (gLlBufferNbrLargeSizeEvent_c+gLlCmdBuffer288Bytes_c)
_pool_id_(1) _eol_
#endif /* LlPoolsDetails_c */
#else /* (gLlMemPoolId_c == 1) */
#ifdef LlPoolsDetails_c
#error Single pool is used, please do not define LlPoolsDetails_c in app_preinclude.h
#else
 #define LlPoolsDetails_c
#endif /* LlPoolsDetails_c */
#endif /* (gLlMemPoolId_c == 1) */
#endif /* (gAppExtAdvEnable_d == 0) */
```

- If using the KW38 A0 samples, define *gXcvrAddTxOffset_d* to select a proper timing for BLE LL. For the KW38 B0 sample, it is not needed. Because the radio drivers support both Gen 3.5 and Gen 4.0, add the following definition to select the Gen 3.5 radio:

```
#define gXcvrAddTxOffset_d

#define RADIO_IS_GEN_3P5 1
#ifndef RF_OSC_26MHZ
#define RF_OSC_26MHZ 0
#endif
```

- Enable deep sleep modes 1 and 3 and disable deep sleep modes 5 and 8:

```
#define cPWR_EnableDeepSleepMode_1 1 //0
#define cPWR_EnableDeepSleepMode_3 1 //0
#define cPWR_EnableDeepSleepMode_5 0 //1
#define cPWR_EnableDeepSleepMode_8 0 //1
```

- Change the application connection sleep mode and the default deep sleep mode:

```
#define gAppDeepSleepMode_c 1 // 8
#define cPWR_DeepSleepMode 3 //5
```

5 Build and run Bluetooth LE connectivity stack examples

All the examples referenced in the *Bluetooth LE Demo Applications User's Guide* are compatible with the MKW38 devices after the modifications described in this document. The changes required at the application level may be different, depending on the application.

6 Revision history

This table summarizes the changes done to this document since the initial release.

Table 6. Revision history

Revision number	Date	Substantive changes
2	06/2021	Added a new section Migration from KW35 series non-wettable flank package to KW38 series HVQFN48 wettable flank.
1	07/2020	Modified Peripherals instantiation .
0	04/2020	Initial release

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Limited warranty and liability — Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

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.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. 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.

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

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 30 June 2021

Document identifier: AN12560

arm