AN12559

Migration Guide from MKW38A512xxx4 to MKW39A512xxx4

Rev. 0 — 04/2020 Application Note

Contents

1 Introduction

This document describes how to migrate from Kinetis MKW38A512xxx4 MCUs to Kinetis MKW39A512xxx4 MCUs with emphasis on connectivity software. In this document, the MKW38A512xxx4 and MKW39A512xxx4 are referred to as MKW38 and MKW39, respectively. The document is intended for software engineers, software testers, software integrators, and customers designing their own hardware.

2 Hardware considerations

The MKW39 wireless MCU is pin-to-pin compatible with MKW38 and almost all peripherals are the same in both devices. The main difference between the MKW38 and MKW39 MCUs is the number of communication interfaces. The MKW38 MCU presents a second instance of LPUART and a FlexCAN modules.

Table 1. Differences between MKW38 and MKW39

Device	Second instance of LPUART	FlexCAN module
MKW38	X	X
MKW39		

2.1 Peripherals instantiation

The KW38/39 devices in the 48-pin HVQFN package are pin-to-pin compatible. The MKW39 MCU does not support LPUART1 and FlexCAN. Therefore, the signals of the last instances are not available for multiplexing in this device. The **bold** alternatives are only included for the MKW38 MCU.

Table 2. MKW38/39 instance comparative

Pin Name	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8	ALT9
PTA16		PTA16/ LLWU_P4	SPI1_SOUT	**LPUART1_ RTS_b**		TPM0_CH0				
PTA17		PTA17/ LLWU_P5	SPI1_SIN	**LPUART1_ RX**	**CAN0_TX**	TPM_ CLKIN1				

Table continues on the next page...

Table 2. MKW38/39 instance comparative (continued)

PTA18		PTA18/ LLWU_P6	SPI1_SCK	**LPUART1_ TX**	**CAN0_RX**	TPM2_CH0				
PTA19	ADC0_SE5	PTA19/ LLWU_P7	SPI1_PCS0	**LPUART1_ CTS_b**		TPM2_CH1				
PTB0		PTB0/ LLWU_P8/ RF_ RFOSC_ EN		I2C0_SCL	CMP0_OUT	TPM0_CH1		CLKOUT	**CAN0_TX**	
PTB1	ADC0_SE1/ CMP0_IN5	PTB1/ RF_ PRIORITY	DTM_RX	I2C0_SDA	LPTMR0_ ALT1	TPM0_CH2		CMT_IRO	**CAN0_RX**	
PTB3	ADC0_SE2/ CMP0_IN4	PTB3/ ERCLK32K/ RF_ACTIVE	**LPUART1_ RTS_b**	TPM0_CH1	CLKOUT	TPM1_CH1		RTC_ CLKOUT	TPM2_CH1	
PTB16	EXTAL32K	PTB16	**LPUART1_ RX**	I2C1_SCL		TPM2_CH0				
PTB17	XTAL32K	PTB17	**LPUART1_ TX**	I2C1_SDA		TPM2_CH1				
PTB18	ADC0_SE4/ CMP0_IN2	PTB18	**LPUART1_ CTS_b**	I2C1_SCL	TPM_ CLKIN0	TPM0_CH0		NMI_b		
PTC3		PTC3/ LLWU_P11	RX_ SWITCH	I2C1_SDA	LPUART0_ TX	TPM0_CH1		DTM_TX	SPI1_SIN	**CAN0_TX**

Table continues on the next page...

Table 2. MKW38/39 instance comparative (continued)

PTC4		PTC4/ LLWU_P12/ RF_ACTIVE	ANT_A	EXTRG_IN	LPUART0_ CTS_b	TPM1_CH0		I2C0_SCL	SPI1_PCS0	**CAN0_RX**
PTC16		PTC16/ LLWU_P0/ RF_ STATUS	SPI0_SCK	I2C0_SDA	LPUART0_ RTS_b	TPM0_CH3			**LPUART1_ RTS_b**	
PTC17		PTC17/ LLWU_P1/ RF_EXT_ OSC_EN	SPI0_SOUT	I2C1_SCL	LPUART0_ RX			DTM_RX	**LPUART1_ RX**	
PTC18		PTC18/ LLWU_P2	SPI0_SIN	I2C1_SDA	LPUART0_ TX			DTM_TX	**LPUART1_ TX**	
PTC19		PTC19/ LLWU_P3/ RF_ EARLY_ WARNING	SPI0_PCS0	I2C0_SCL	LPUART0_ CTS_b				**LPUART1_ CTS_b**	

NOTE

Table 2 is not a full description of the MKW38/39 pinout. It can be found in the MKW39/38/37 reference manual.

3 Software development kit download and install

This chapter shows how to download the Software Development Kit (SDK) for MKW39A512xxx4 MCUs, because it is used as a starting point to migrate MKW38 code to the MKW39 MCU. Perform the following steps to download the SDK package:

1. Go to the MCUXpresso web page (mcuxpresso.nxp.com).
2. Log in with your registered account.
3. Search for the “KW39” device. Then click on the suggested processor and click the “Build MCUXpresso SDK” button.

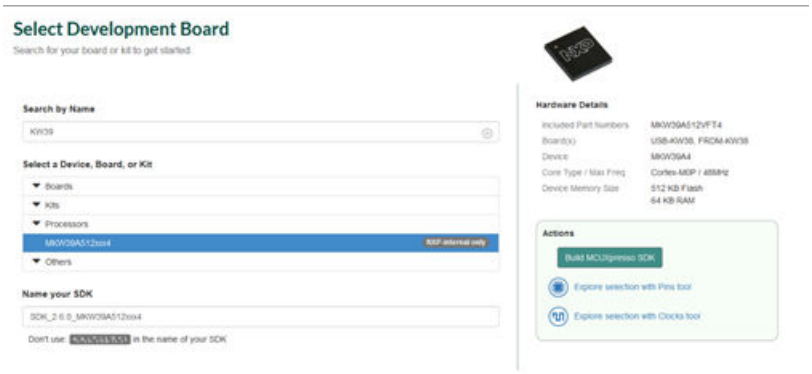

Figure 1. Building MKW39 SDK package

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

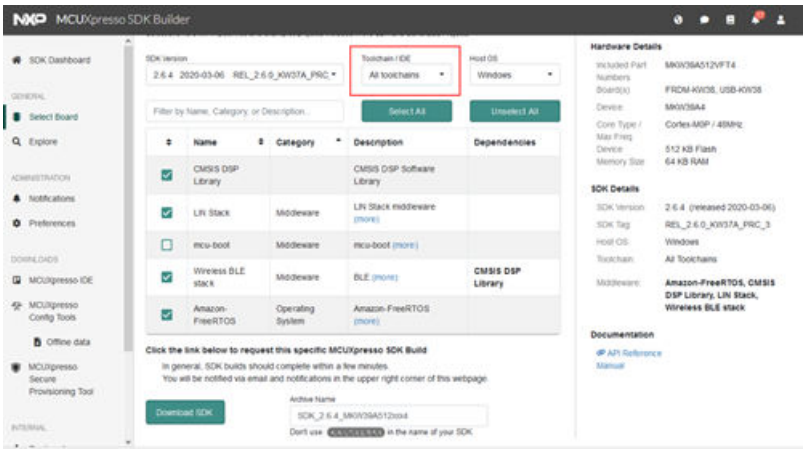

Figure 2. Selecting version of SDK

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 the 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 the “Download SDK Archive” option.

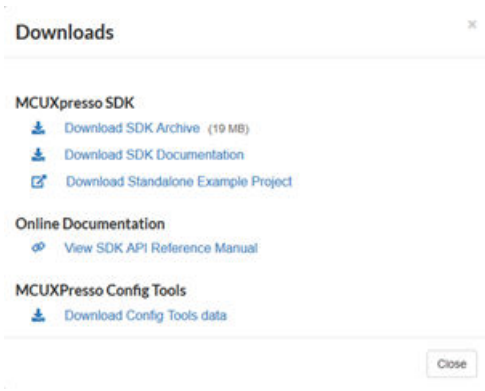

Figure 3. MKW39 SDK displayed in dashboard

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

Figure 4. MKW39 SDK displayed in dashboard

At this point, you have downloaded and installed the SDK package for MKW39A devices.

4 Software migration in IAR Embedded Workbench IDE

This chapter shows how to migrate the MKW38 example code to the MKW39 MCU in the IAR Embedded Workbench IDE. The heart rate sensor project is used as the base in this document, because it has 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, a “bare-metal” version of the project was used. However, the same steps apply for FreeRTOS projects. Some of the paths related to “bare-metal” projects may differ for FreeRTOS versions.

1. Open the heart rate sensor project located at the following path:

   <SDK_root>/boards/frdmkw38/wireless_examples/bluetooth/hrs/bm/ iar/heart_rate_sensor_bm.eww

2. Select the project in the workspace and press Alt + F7 to open project options.

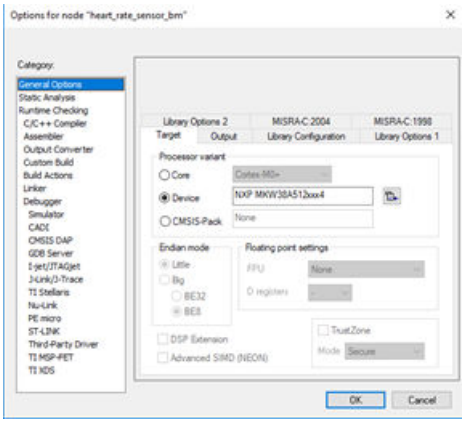

Figure 5. Project options window

3. In the "General Options/Target" window, click the icon next to the device name and select the appropriate device (NXP->KinetisKW->KW3x->NXP MKW37A512xxx4) and click the "OK" button.

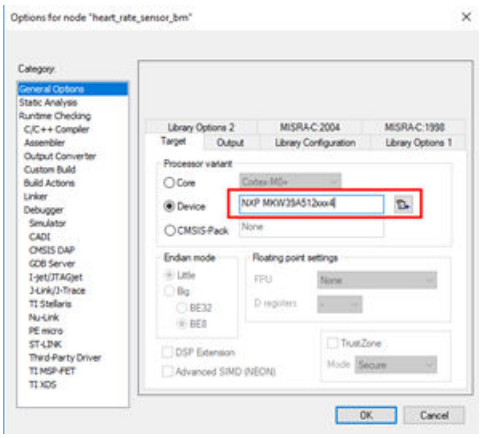

Figure 6. Selecting MKW39 device

4. Create a new folder named *MKW39A4* at following paths:

   <SDK_root>/middleware/wireless/framework/DCDC/Interface

   <SDK_root>/middleware/wireless/framework/DCDC/Source

   <SDK_root>/middleware/wireless/framework/LowPower/Interface

   <SDK_root>/middleware/wireless/framework/LowPower/Source

   <SDK_root>/middleware/wireless/framework/XCVR

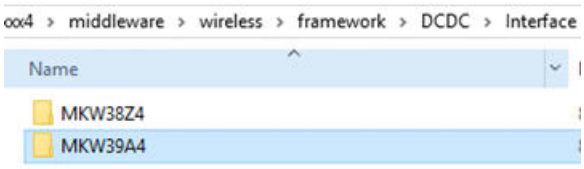

Figure 7. Creating MKW39A folders

5. Copy all files from the *MKW38Z4* folders located at the path mentioned above and paste them into the *MKW39A4* folders.

Figure 8. Creating MKW39 files

6. Select the heart rate sensor application in the workspace and press Alt+F7 to open the "project options" window. In the "C/C++ Compiler/Preprocessor" window, rename the paths related to the *MKW38A/Z4* folders to "MKW39A4" by modifying the name in the "additional include directories" text box. The results must look similar to the following:

   $PROJ_DIR$/../../../../../../../devices/MKW39A4/utilities

   $PROJ_DIR$/../../../../../../../devices/ MKW39A4/drivers

   $PROJ_DIR$/../../../../../../../middleware/wireless/framework/LowPower/Interface/MKW39A4

   $PROJ_DIR$/../../../../../../../devices/MKW39A4/utilities/str

   $PROJ_DIR$/../../../../../../../devices/MKW39A4/utilities/debug_console

$PROJ_DIR$/../../../../../../../middleware/wireless/framework/DCDC/Interface/ MKW39A4

$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW39A4/drv

$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW39A4/drv/nb2p4ghz

$PROJ_DIR$/../../../../../../../middleware/wireless/framework/XCVR/MKW39A4/drv/nb2p4ghz/configs/gen35

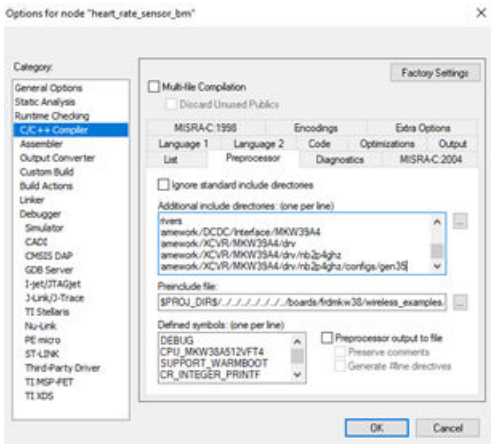

Figure 9. Updating MKW39 files

7. Rename the CPU_MKW38A512VFT4 macro to CPU_MKW39A512VFT4 and delete the FRDM_KW38 macro in the "defined" symbols text box. Click the "OK" button.

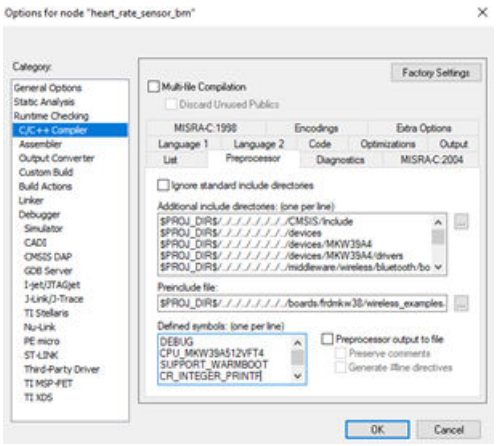

Figure 10. Defining MKW39 macros

8. Expand the startup folder, select all files, right-click and select the “Remove” option. Right-click the folder and select “Add/Add files”. Add the *startup_MKW39A4.s* file located at this path:

   <SDK_root>/devices/MKW39A4/iar/startup_MKW39A4.s

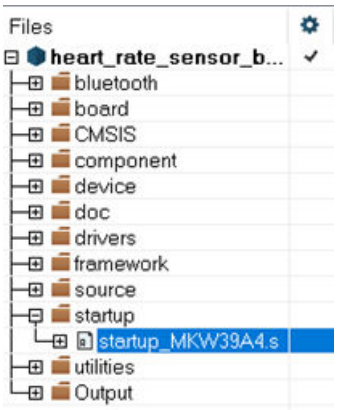

Figure 11. Replacing startup system file

9. Expand the device folder, select all files, right-click, and select the “Remove” option. Right-click the folder and select “Add/Add files”. Add the *fsl_device_registers.h, MKW39A4.h, MKW39A4_features.h, system_MKW39A4.h*, and *system_MKW39A4.c* files at the following path:

   <SDK_root>/devices/MKW39A4

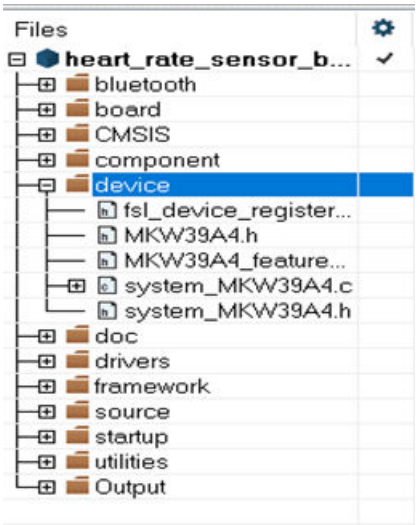

Figure 12. Replacing device files

10. Expand the *drivers* folder, select the *fsl_flexcan.h* and *fsl_flexcan.c* files, right-click, and select the “Remove” option.
11. Some files must be updated. Open the *clock_config.c* file in the *board* folder. Delete the CLOCK_SetLpuart1Clock call to function in the BOARD_BootClockRUN function.

```
CLOCK_SetLpuart0Clock(SIM_LPUART_CLK_SEL_OSCERCLK_CLK);
/* Set LPUART1 clock source. */
// Delete this line: CLOCK_SetLpuart1Clock(SIM_LPUART_CLK_SEL_OSCERCLK_CLK);
/* Set TPM clock source. */
CLOCK_SetTpmClock(SIM_TPM_CLK_SEL_OSCERCLK_CLK);
```

Figure 13. Updating clock_config.c file

12. If necessary, open the *app_preinclude.h* file in the *source* directory in the workspace. This file contains important information about the board, such as the number of switches and LEDs, timers, power consumption settings, and so on. Examine and update this file to fit into your own custom board.

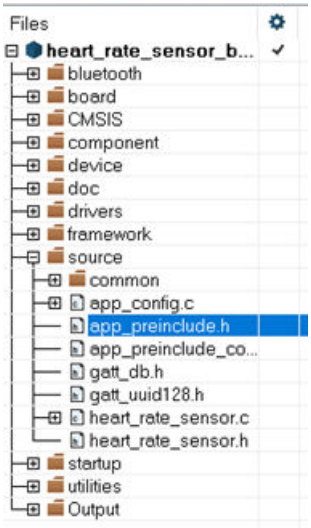

Figure 14. Location of preinclude file

13. If it necessary, open the pin mux files (*pin_mux.c* and *pin_mux.h*) and gpio files (*gpio_pins.c* and *gpio_pins.h*) in the *board* directory in the workspace. These files contain alternatives, options, and multiplexing information of the pins. Examine and update them to fit into your own custom board.

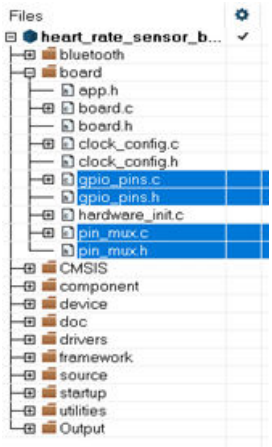

Figure 15. Location of pin mux and gpio files

14. Select the project in the workspace, right-click and press clean.
15. Rebuild the project. At this point, the project is already migrated and you can customize the application for your own goal using the MKW39 device.

5 Software migration in MCUXpresso IDE

This chapter shows how to migrate MKW38 example code to the MKW39 MCU in the MCUXpresso IDE. The heart rate sensor project is used as a base because it has easy-to-understand example and involves the Bluetooth LE connectivity software stack (included in the SDK).

5.1 Changes required in project properties

NOTE

In this section, a “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 in the FreeRTOS versions.

1. Import the KW39 SDK into the MCUXPresso IDE:
   a. Open the MCUXpresso IDE.
   b. Go to the “Installed SDKs’ tab and drag and drop the KW39 SDK.

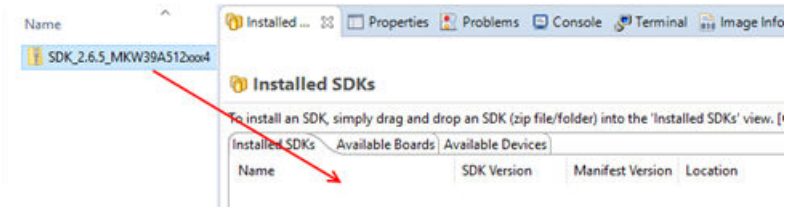

Figure 16. Importing SDK into MCUXpresso IDE

2. Find the "Quickstart Panel" in the lower left-hand corner.

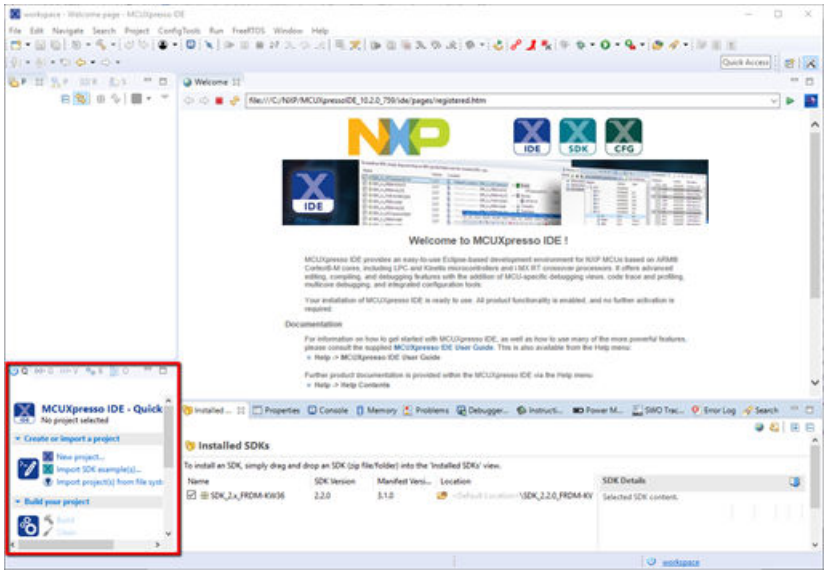

Figure 17. Quickstart Panel perspective in MCUXpresso IDE

3. Click the “Import SDK examples(s)…” option.

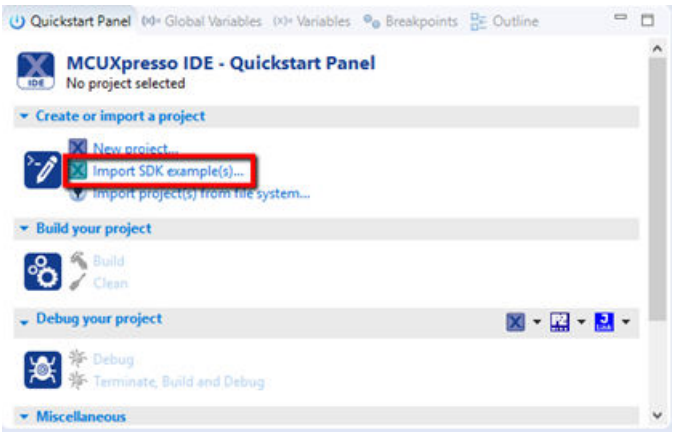

Figure 18. Importing SDK example

4. Expand the *KW3X* folder, select "MKW38A512xxx4", click the "frdmkw38" board, and click "Next".

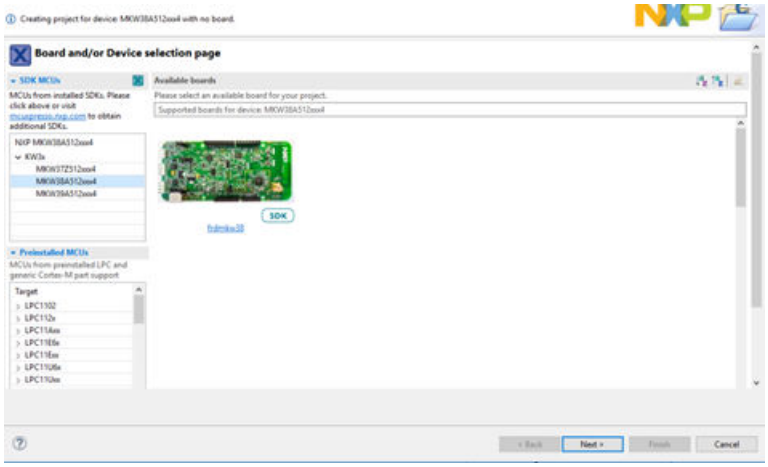

Figure 19. Select FRDM-KW38 board

5. Use the arrow button to expand the list and locate the “hrs” project (wireless_examples -> bluetooth -> hrs). Select the "bm" version of the project. In “SDK Debug Console”, select “UART” and click “Finish”.

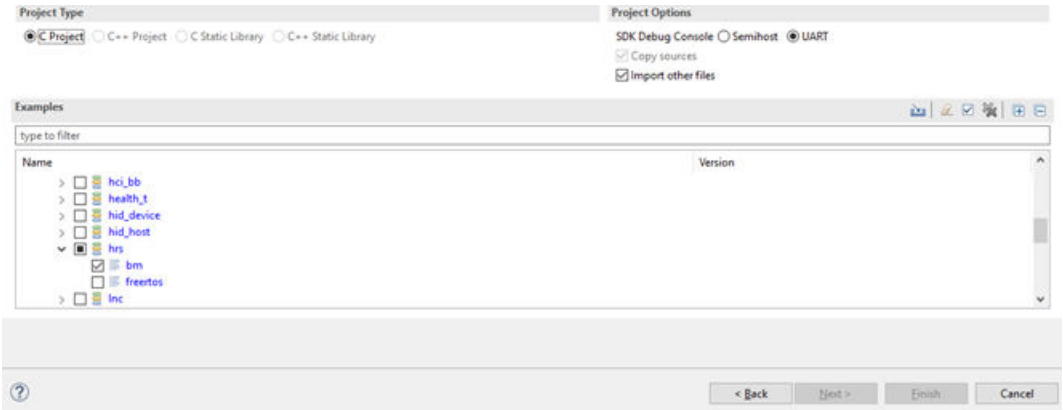

Figure 20. Opening heart rate sensor project

6. Go to "Project/Properties". Expand "C/C++ Build/MCU settings" and select the "MKW39A512xxx4" MCU. Click the "Apply and Close" button to save the configuration.

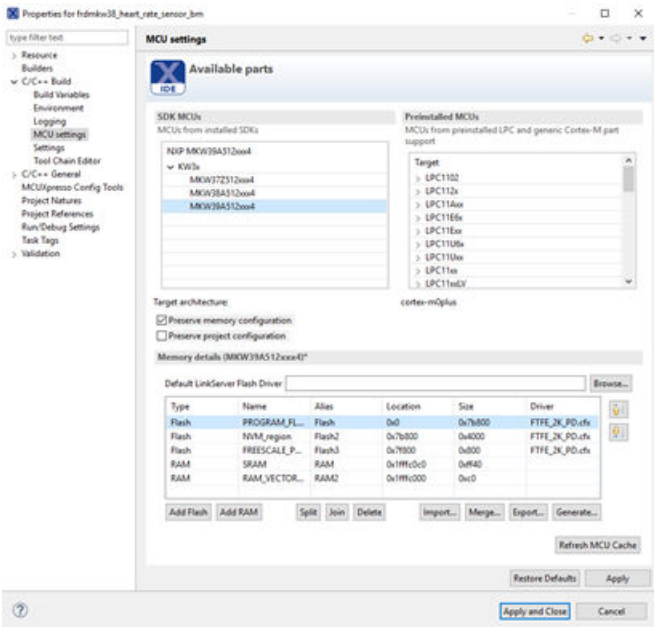

Figure 21. Selecting MKW39 MCU

7. Rename the *MKW38Z4* folder to "MKW39A4" at the following path:

   <MCUXpresso_hrs_project_root>/framework/XCVR

   Default MCUXpresso root in Documents/MCUXpressoIDE<version>/workspace

8. Select the heart rate sensor project in the workspace and press F5 to refresh the modified folders.

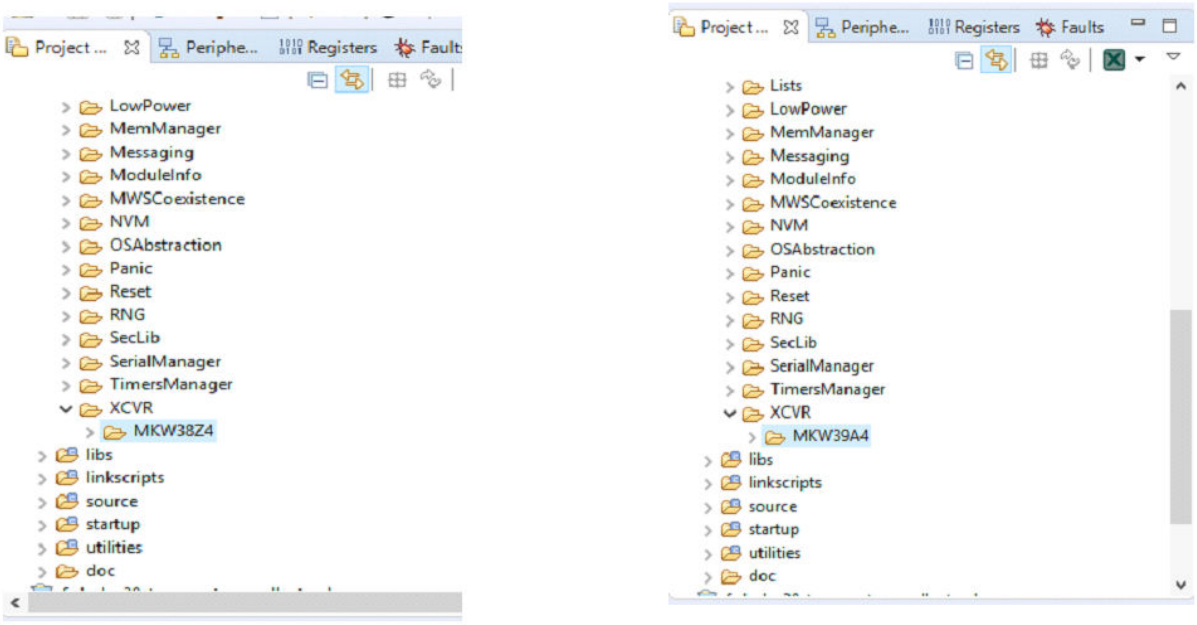

Figure 22. Updating project folders in workspace

9. Open the "Project/Properties" window in the MCUXpresso IDE. Go to "C/C++ Build/Settings" and select the *MCU C Compliler/Includes* folder in the "Tool Settings" window. Edit all paths related to the MKW38 MCU, in accordance with the *MKW39* folders created before. The results must look as follows:

../framework/XCVR/MKW39A4

../framework/XCVR/MKW39A4/nb2p4ghz/configs

../framework/XCVR/MKW39A4/nb2p4ghz

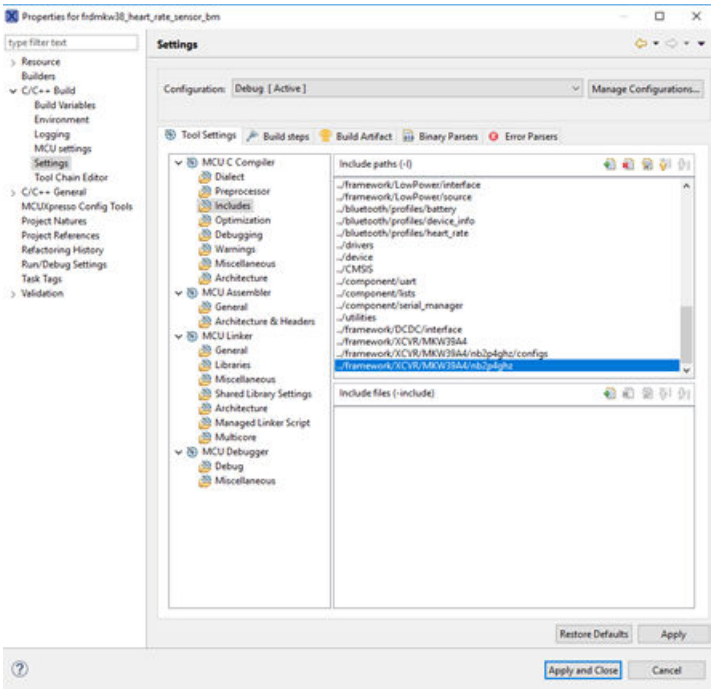

Figure 23. Updating include paths

10. Select the *MCU Assembler/General* folder in "Tool Settings". Edit the paths related to the MKW38 MCU. The results must look as follows:

    ../framework/XCVR/MKW39A4

    ../framework/XCVR/MKW39A4/nb2p4ghz/configs

    ../framework/XCVR/MKW39A4/nb2p4ghz

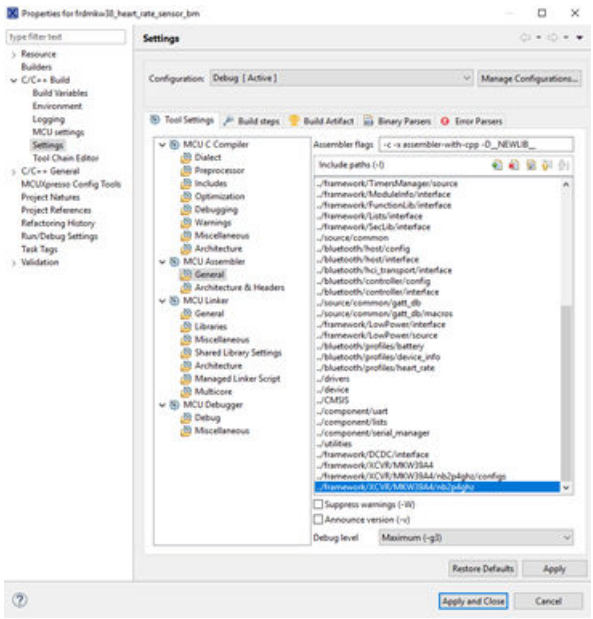

Figure 24. Updating assembler paths

11. Open the "MCU C Compiler/Preprocessor" window and rename the CPU_MKW38A512VFT4 and CPU_MKW38A512VFT4_cm0plus macros to CPU_MKW39A512VFT4 and CPU_MKW39A512VFT4_cm0plus, respectively. Delete the FRDM_KW38 macro. Click the "Apply and Close" button.

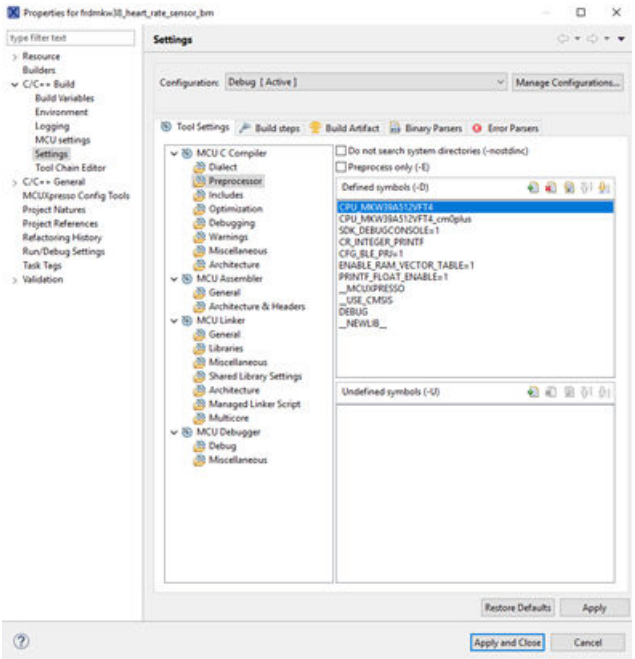

Figure 25. Updating preprocessor macros

12. Delete the *fsl_device_registers.h*, *MKW38A4.h*, *MKW38A4_features.h*, *system_MKW38A4.h*, and *system_MKW38A4c.* files located in the *device* folder in the workspace.
13. Unzip the MKW39A SDK package and search for the *fsl_device_registers.h*, *MKW39A4.h*, *MKW39A4_features.h,system_MKW39A4.h*, and *system_MKW39A4.c* files in this folder at the following paths:

    <SDK_folder_root>/devices/MKW39A4/fsl_device_registers.h

<SDK_folder_root>/devices/MKW39A4/MKW39A4.h

<SDK_folder_root>/devices/MKW39A4/MKW39A4_features.h

<SDK_folder_root>/devices/MKW39A4/system_MKW39A4.h

<SDK_folder_root>/devices/MKW39A4/system_MKW39A4.c

Drag and drop these files into the *device* folder in the workspace.

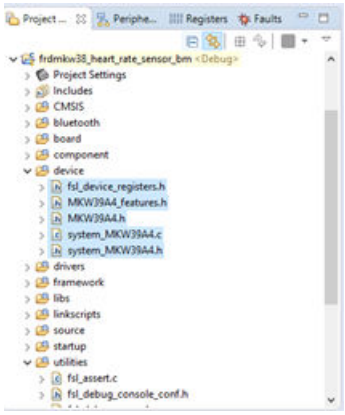

Figure 26. Updating device files

14. Replace the *startup_mkw38a4.c* file located in the *startup* folder in the workspace by the *startup_mkw39a4.c* file. This file is located in the MKW39A SDK at the following path:

    <SDK_folder_root>/devices/MKW39A4/mcuxpresso/startup_MKW39A4.c

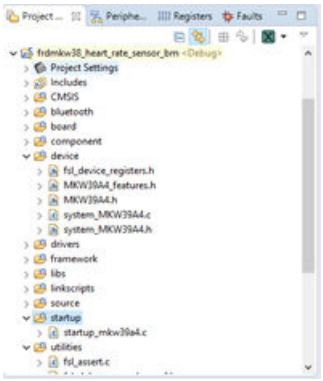

Figure 27. Updating startup file

15. Some files must be updated. Open the *clock_config.c* file in *board* folder. Delete the CLOCK_SetLpuart1Clock call to function in the BOARD_BootClockRUN function.

```
/* Set LPUART0 clock source. */
CLOCK_SetLpuart0Clock(SIM_LPUART_CLK_SEL_OSCERCLK_CLK);
/* Set LPUART1 clock source. */
//Delete this line: CLOCK_SetLpuart1Clock(SIM_LPUART_CLK_SEL_OSCERCLK_CLK);
/* Set TPM clock source. */
CLOCK_SetTpmClock(SIM_TPM_CLK_SEL_OSCERCLK_CLK);
```

Figure 28. Updating clock_config.c file

16. If necessary, open the *app_preinclude.h* file under the *source* directory in the workspace. This file contains important information about the board, such as the number of switches and LEDs, timers, power consumption settings, and so on. Examine and update this file to fit into your own custom board.

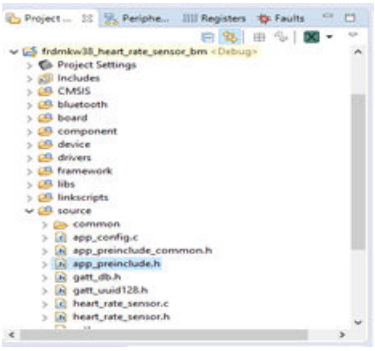

Figure 29. Location of preinclude file

17. If necessary, open the pin mux files (*pin_mux.c* and *pin_mux.h*) and the gpio files (*gpio_pins.c* and *gpio_pins.h*) in the *board* directory in the workspace. These files contain alternatives, options, and multiplexing information of the pins. Examine and update them to fit into your own custom board.

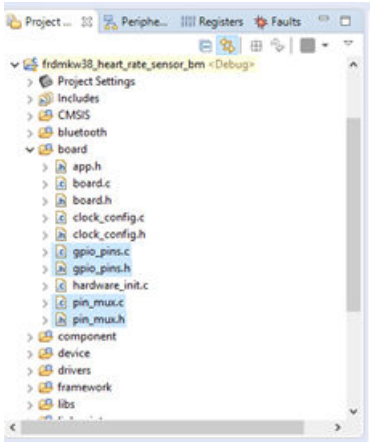

Figure 30. Location of pin mux and gpio files

18. Expand the *drivers* folder, select the *fsl_flexcan.h* and *fsl_flexcan.c* files, right-click, and select the “Delete” option.
19. Build the project. At this point, the project is already migrated and you are ready to run the demo using MKW39.

6 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 MKW39 device after the modifications described in this document.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

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

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

Date of release: 04/2020

Document identifier: AN12559

arm