User Manual

for S32K14X MCAL Sample Application

Document Number: UMSAASR4.2R1.0.0
Rev. 1.0

Contents

Section number **Title** **Page**

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	08/04/2016	Nicolae Dobrostomat	0.9.0
1.1	24/08/2017	Stefan Tataru	1.0.0 Release

Chapter 2
About this Manual

This User Manual describes utilization of the sample application for S32K14X microcontroller with Autosar MCAL 4.2 Rev0001 version RTM 1.0.0.

2.1 Acronyms and Definitions

Table 2-1. Acronyms and Definitions

Abbreviation / Acronym	Description
DIO	Digital Input Output Driver
PORT	Port Driver
BSW	Basic Software
ADC	Analog Digital Converter
FEE	Flash EEPROM Emulation
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ECU	Electronic Control Unit
ISR	Interrupt Service Routine
OS	Operating System
GUI	Graphical User Interface
API	Application Programming Interface
EcuM	ECU state Manager
WDG	Watchdog Driver
PLL	Phase Lock Loop
LED	Light Emitting Diode
PB Variant	Post Build Variant
LT Variant	Link Time Variant
PC Variant	Pre Compile Variant

2.2 Reference List

Table 2-2. Reference List

#	Items	Version
1	S32K144 Microcontroller Reference Manual	Reference Manual, Rev.4 DraftA, 06/2017

Chapter 3
Installation Steps

3.1 Hardware Installation

The hardware installation describes setup of the S32K144 EVB (SCH-29248 REV B) Freedom Board.

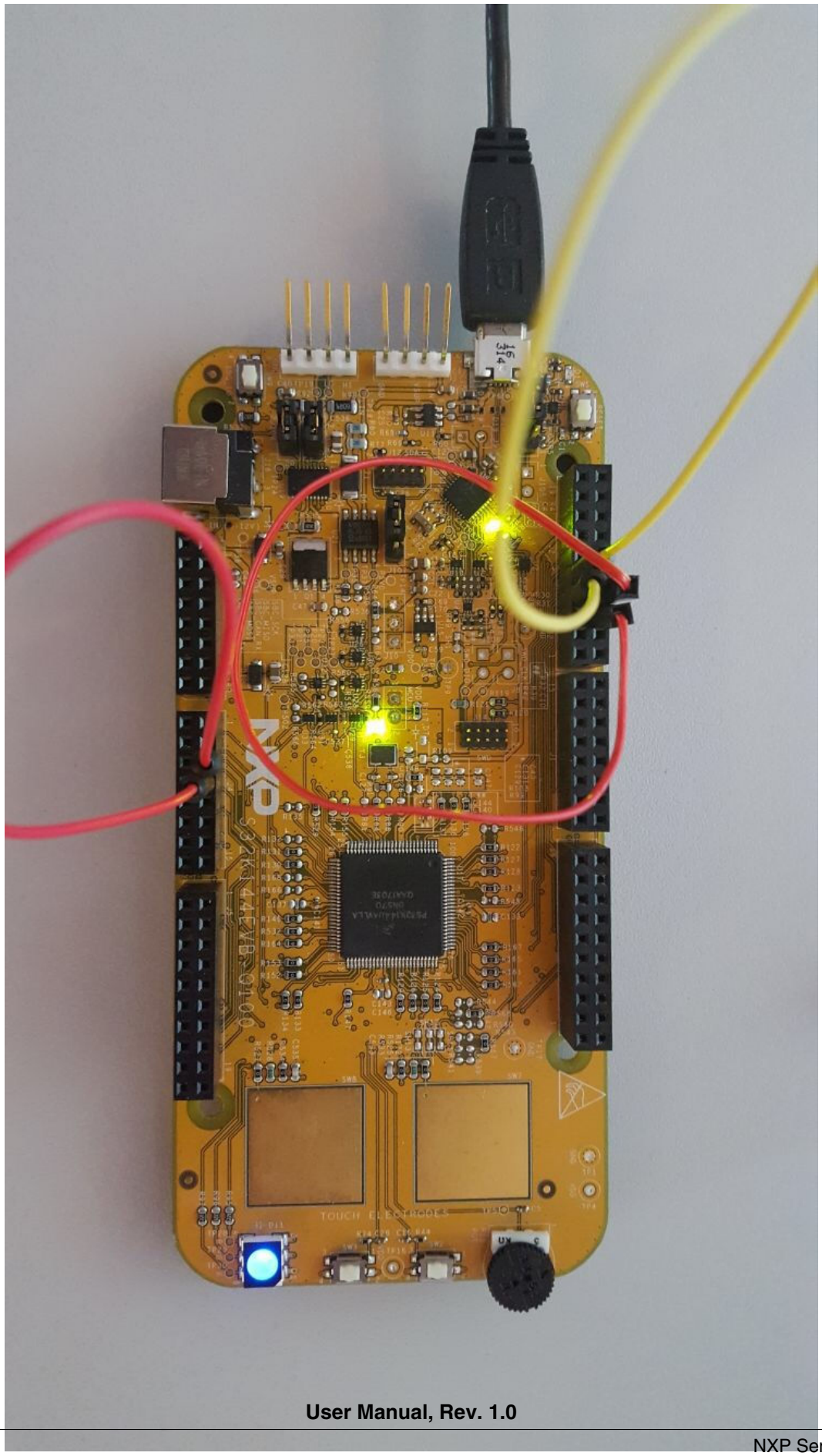

Figure 3-1. S32K144 EVB (SCH-29248 REV B) Freedom Board

To powerup the board use the onboard microusb port.

To use other debugger than the onbard OpenSDA, use the J14 SWD connector.

Check all jumper configurations as shown in Figure 4-1. S32K144 EVB (SCH-29248 REV B) Evaluation Board and Figure 4-2. Buttons and LEDs Jumper Configuration

- J2 6(D0)-8(E6) closed – needed for PWM Blue Led and Icu input connection
- J2 7(B4)-9(B3) closed – needed for SPI loopback connection
- JJ4 5(B0)-7(B1) closed – needed for LIN loopback connection

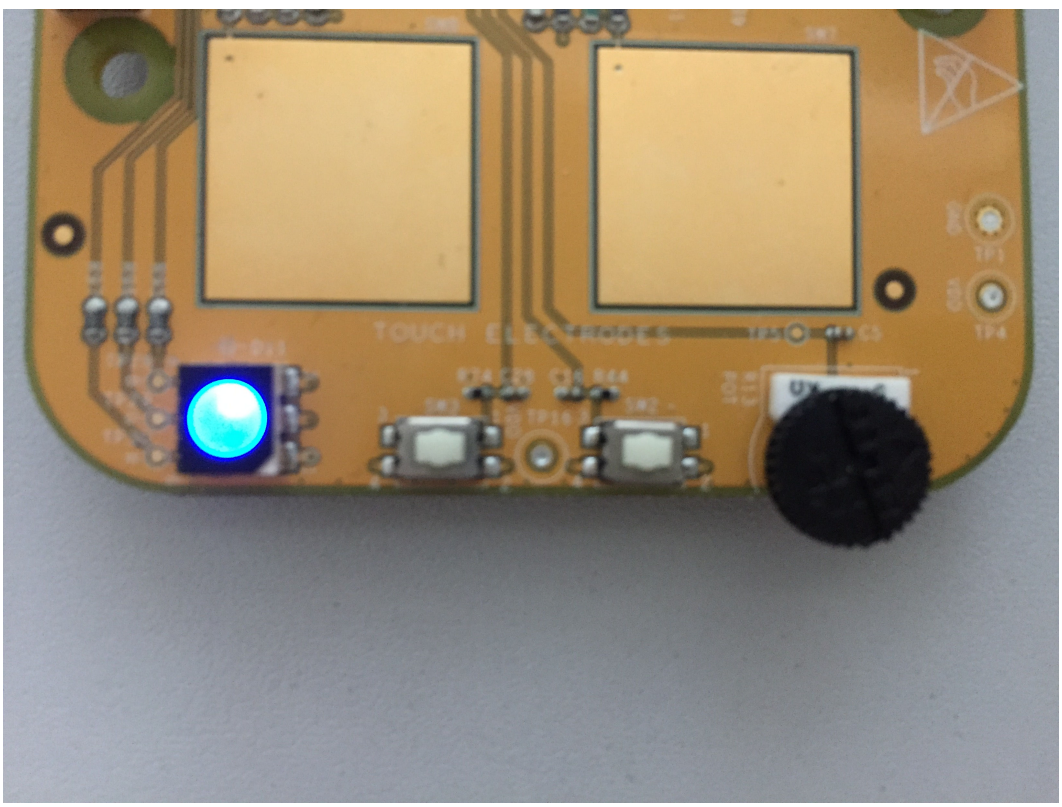

Figure 3-2. Buttons and LEDs Configuration

3.2 Software Installation

Please install the MCAL package on your computer. The package includes the MCAL Sample Application example with the following folder structure:

Folder or file	Description
- bin folder	generated object files and linker output files are stored into this folder
- cfg folder	contains configuration files generated by Tresos tool

Table continues on the next page...

Folder or file	Description
- include subfolder	contains files with pre-compile configurations
- src subfolder	contains files with post-build and link-time configurations
- doc folder	contains documentation
- include folder	contains header files for types definition
- Iar folder	contains worksapce and project files for Iar Embedded Workbench IDE
- make folder	makefiles used for building the application
- src folder	contains the application source code file
- toolchains folder	files needed to build with various toolchains (startup, linker command files)
- makefile file	the MCAL sample application makefile
- Modules file	specifies which modules are compiled and linked
- make.bat file	launches the make command
- launch.bat file	contains path to the Tresos Studio installation and launches the make.bat file
- Tresos folder/workspace	contains the Tresos project with the application configuration

3.2.1 Tresos Project Installation

The following procedure requires that the user has EB Tresos Studio installed.

Procedure:

1. Make sure that all MCAL plugins are already installed in the Tresos Studio plugins directory

2. Open Tresos Studio

3. Import Sample application project

a. Click on "File" and select "Import"

b. Select "Existing Projects into Workspace" and click on "Next" button as shown in Figure 4-3. Import Window - the First View

c. Next steps are depicted in Figure 4-4. Import Window - the Second View

- Select "Select root directory" and click on "Browse"
- Select the location of the [project] folder in the installed Sample application package folder (Tresos/workspace/[project])
- Select "Copy projects into workspace"
- Click on "Finish" button

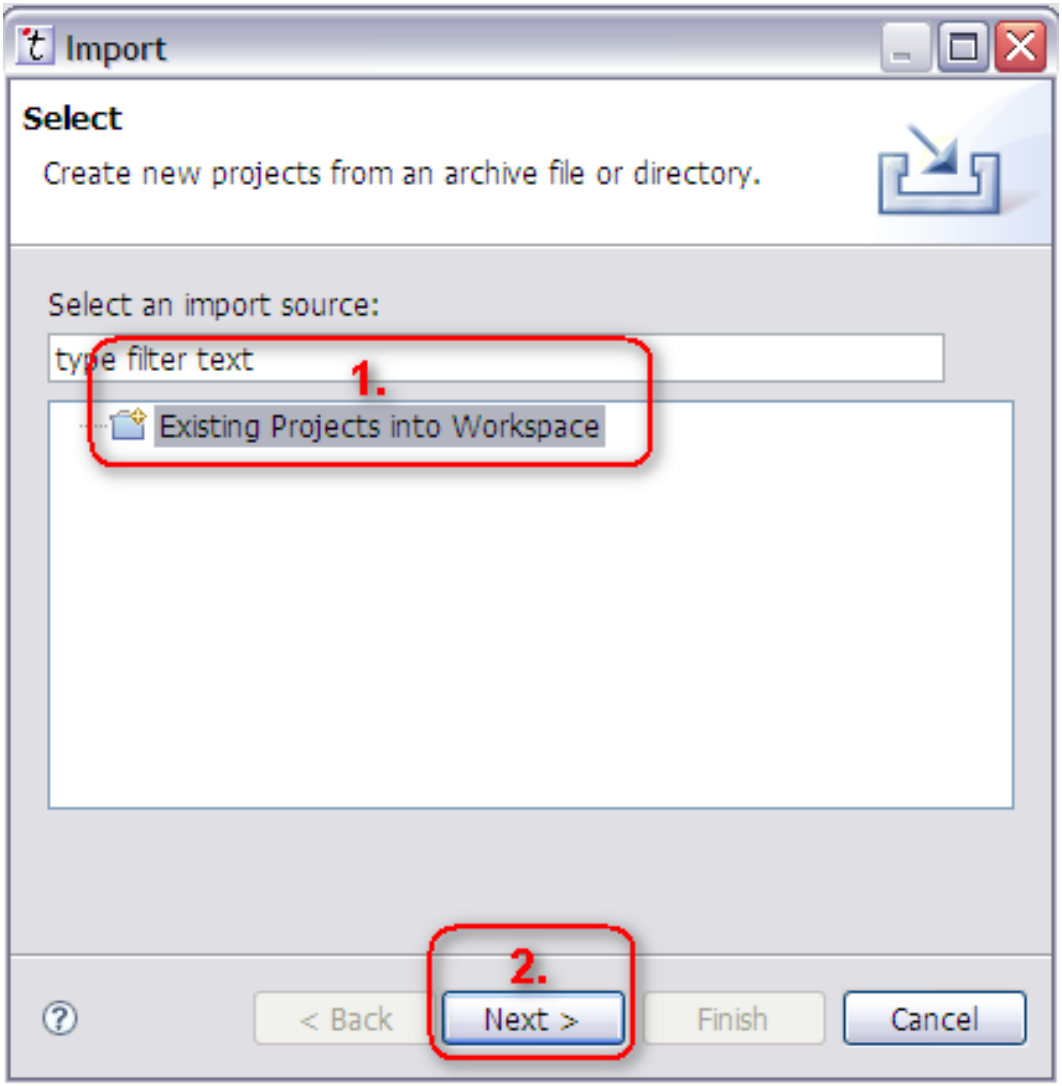

Figure 3-3. Import Window - the First View

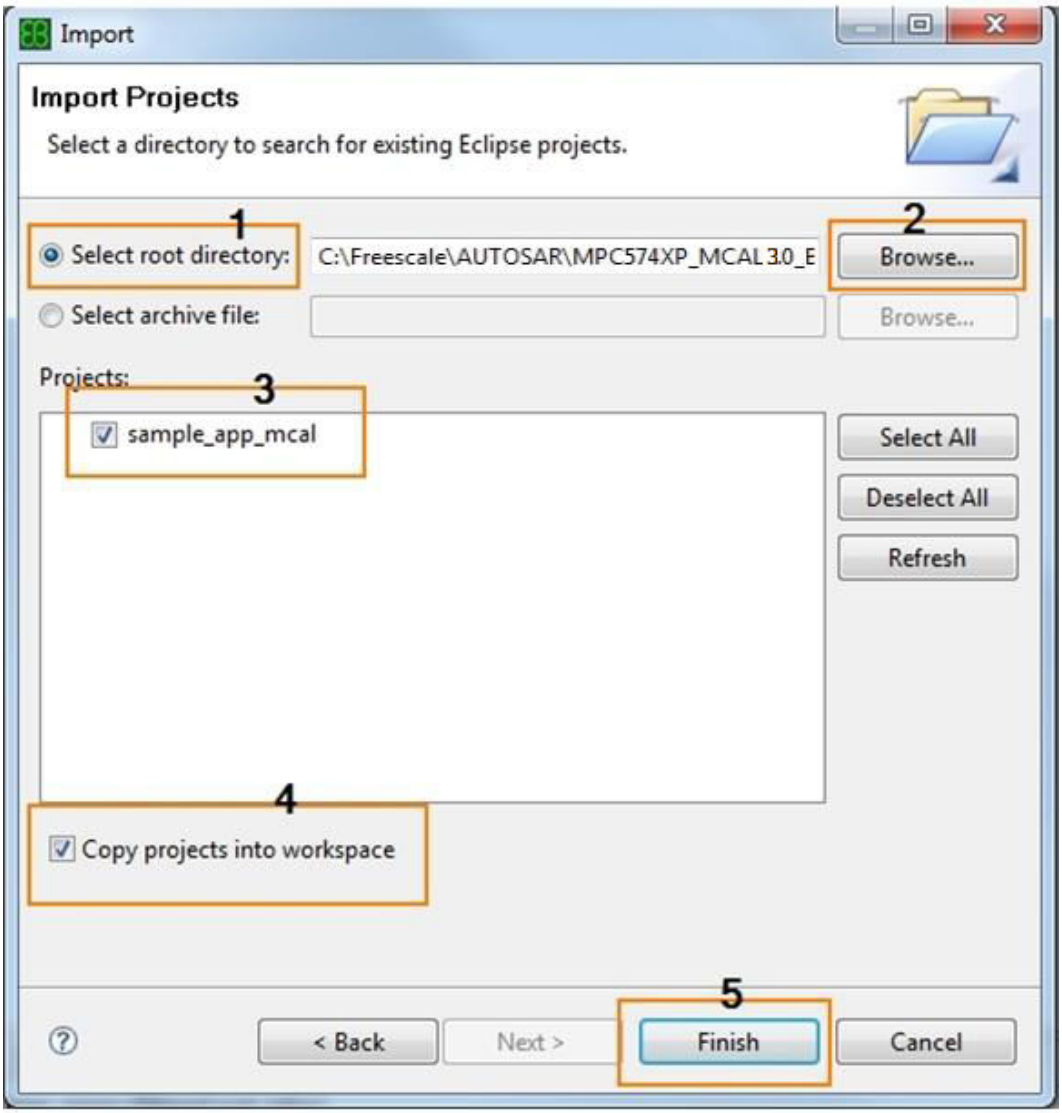

Figure 3-4. Import Window - the Second View

3.2.2 MCAL Application Configuration

The following procedure requires that the user has EB Tresos Studio installed and the toolchains versions specified in the MCAL Release Notes.

The toolchain that will be used needs to be installed for correct operation and the path to the installation location shall be added into the system environment variable(s):

- GHS_DIR for GreenHills Multi

o Ex: SET GHS_DIR=C:/ghs_arm/comp_201716

- IAR_DIR for IAR Embedded Workbench

o Ex: SET IAR_DIR=C:/iar_arm/ARM_8.11.2

- GCC_DIR for GNU Tools for ARM Embedded Processors

o Ex: SET GCC_DIR=C:/Progra~2/GNUTOO~1/62017-~1

Note

The path to the toolchain must not contain spaces. In case the compiler is installed into a path with spaces, the variable must be set with the "short" folder name (8.3 version of the file name that can be displayed with dir /X in command prompt)

Procedure:

1. Open launch.bat file in a text editor and specify the EB Tresos Studio location in the TRESOS_DIR parameter as shown in Figure 4-5. Configuration of the Tresos Studio Location

2. Make sure that installation location of the compiler is added in the system environment variable (GHS_DIR or DIAB_DIR)

3. Setup the plugins folder location if the plugins are not installed in the Tresos plugins folder

```
:: uncomment line below if you do not set TRESOS_DIR over environment
:: SET TRESOS_DIR=
:: SET GHS_DIR=
:: SET IAR_DIR=
:: SET LINARO_DIR=
:: SET PLUGINS_DIR=
```

Figure 3-5. Configuration of the Tresos Studio Location and the plugins folder location

Chapter 4
Sample Application Example Description

This application demonstrates an example of usage for the MCAL modules. It is not part of the production code deliverables

4.1 The application software functionality

Initializes WDG driver and configures it to Slow MODE

Initializes MCU module

- Initializes PLL and configures it to 8MHz

Checks whether PLL is locked

Activates the PLL clock to the MCU clock distribution

Initializes PORT module. Pins configuration is show in section PORT and DIO Modules - Pin Configuration and DioChannel Assignment for keys and leds, and in PORT Configuration excluding Leds and Keys

Initializes the CAN driver and Sample application specific data for this driver

Initializes the SPI driver and Sample application specific data for this driver

Initializes the PWM driver and Sample application specific data for this driver

Initialize the ADC driver and Sample application specific data for this driver

Initialize DIO driver. The DIO driver is used to toggle the Green and Red Leds and retrieve the value of SW2.

Initialize the GPT driver. On GPT notification the Green Led is toggled

Performs while loop

- The WDG is triggered

- If SW2 is pressed the Red Led is set ON and when the timeout expires, the Red Led is set OFF. If the timeout expires the Watchdog will reset the board.
- ADC retrieves the trimmer value.
- The value got from the ADC conversion is used to set up the PWM duty cycle. The output of PWM is on the Blue Led. Moving the trimmer has as effect modifying the intensity of Blue Led.
- Performs a loopback transmission (exploiting CAN's hardware loopback capabilities) on the FlexCAN unit 0
- DSPI unit 0 performs a simple SPI transmission in loopback mode. In this case the SOUT and SIN pins must be tied together in order to have the electrical loopback

Various messages are sent over the UART.

Note

The onboard OpenSDA uart to usb doongle is used to send messages. The serial terminal should have the following settings: 115200, 8, N, 1. Reconnect the USB cable before opening the OpenSDA UART port in the terminal software.

4.2 Description of the LEDs and Buttons functionality

The detailed description of the LEDs and Buttons functionality is depicted in the following table:

Table 4-1. PORT and DIO Modules - Pin Configuration and DioChannel Assignment

PortPin Name	Pin ID (PCR ID)	Pin Mode	Pin Direction	Pin Level	Connected HW	Channel Assignment
PortPin_PWM_Led1	96	FTM2_CH0	Out	Low	Blue Led	-
PortPin_Led2	111	GPIO	Out	High	Green Led	Dio_Led2
PortPin_Led3	112	GPIO	Out	Low	Red Led	Dio_Led3
PortPin_Key1	76	GPIO	In	-	SW 2	Dio_Key1
PortPin_ADC0_SE12	78	ADC0_SE12_ACMP2_IN5	-	-	Pot R13	-

Table 4-2. LEDs and Buttons Functionality

LEDs and Buttons	Functionality	LED ON	LED OFF
Red Led	When SW2 is pressed the led turns on, until the watchdog timeout	When SW2 pressed and watchdog timeout did not expired	If SW2 is not pressed or the watchdog timeout expired
Green Led	Is toggled at each GPT notification	-	-
SW2	Disables triggering the watchdog	-	-

Table 4-3. PORT Configuration excluding Leds and Keys

PortPin Name	Port	Pin ID (PCR ID)	Pin Mode
PortPin_LPSPI0_SCLK	PTB[2]	34	LPSPI0_SCK
PortPin_LPSPI0_SOUT	PTB[4]	36	LPSPI0_SOUT
PortPin_LPSPI0_SIN	PTB[3]	35	LPSPI0_SIN
PortPin_UART1_TX	PTC[7]	71	LPUART1_TX
PortPin_UART1_RX	PTC[8]	72	LPUART1_RX

Chapter 5
Building the Sample Application Example

This section describes the build procedure.

5.1 Building the Sample Application example

Procedure:

1. Open the Windows command prompt window

2. Change the current directory to the sample application folder

3. To build the sample, execute the following command to run launch.bat: launch.bat

4. The object files and linker output file (sample_app_mcal.elf) shall be generated in the /bin subdirectory

5. To execute the sample application load the executable file placed in the /bin subdirectory to the evaluation board using the Lauterbach debugger and run.cmm or run_ram.cmm script.

Note

The launch.bat file calls the make.bat file and then the GNU make utility is called from the Tresos Studio bin directory.

5.2 Building with different compilers

To build the sample application with a different compiler, use the following parameter for the launch command:

launch.bat TOOLCHAIN=[toolchain]

where [toolchain] can have the values:

* ghs - use the GreenHills Multi compiler

* iar - default - use the Windriver DIAB compiler

5.3 Building for different run-modes

To build the sample application for a different run-mode, use the following parameter for the launch command:

launch.bat MODE=[toolchain]

where [toolchain] can have the values:

* SUPR - default - run in Supervisor mode

* USER - un in User mode

Note

In order to run in USER mode, all drivers that need to be executed in this mode should have the "Enable User Mode Support" parameter set to 'true' and their configuration files regenerated from Tresos.

Note

In order to run in USER mode, AUTOSAR OS should not be used since it does not allow other run-modes except Supervisor mode.

5.4 Clean Object and Linker Output Files

To clean the object and linker output files from the folder /bin, execute the following steps

Procedure:

1. Open the Windows command prompt window

2. Change the current directory to sample application folder

3. Execute the following command launch.bat clean

4. The object files and linker output files shall be cleared from the /bin and from the sample application root folders.

5.5 Modifying the Configuration in Tresos Studio

Users may change the application configuration according to their needs.

Procedure:

1. Open the EB Tresos Studio GUI

2. Open previously imported Sample Application project

3. Use the Tresos Studio GUI to modify configuration parameter values and save the changes. The value of the External Crystal Frequency parameter can be changed as depicted in Figure 5-1: Modifying the External Crystal Frequency

4. Select the Sample Application project and click on "Generate" button to generate the configuration files.

5. Copy the generated configuration files from workspace/[project]/output/include directory into the Sample Application folder /cfg/include.

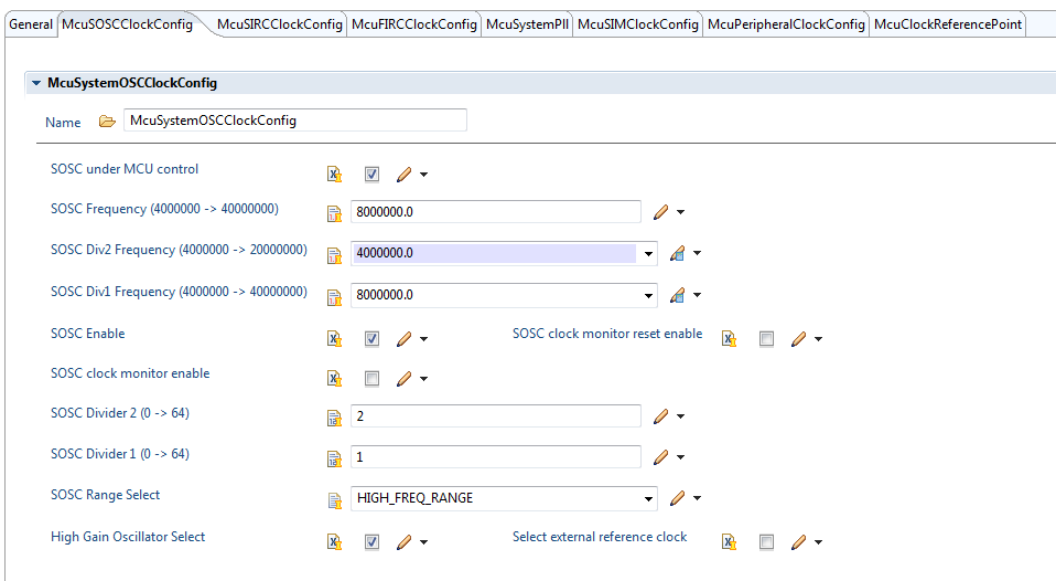

Figure 5-1. Modifying the External Crystal Frequency

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number UMSAASR4.2R1.0.0
Revision 1.0

AUTOSAR