User Manual

for MPC563xM MCAL Integration Application

Rev. 1.1

Contents

Section Number **Title** **Page**

Chapter 1
Revision history

Table 1-1. Revision history

Revision	Date	Author	Description
1.0	14/02/2011	Nicolae Dobrostomat	Initial Version
1.1	23/12/2011	Alex Gauggel	Updated for MPC563xM RTM 2.0.0 Release

Chapter 2
About this Document

This User Manual describes utilization of the sample application for MPC563xM microcontroller with Autosar MCAL 3.0 version RTM 2.0.0.

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

2.1 References

Table 2-2. References

Item	Version
MPC5634M Microcontroller Reference Manual	Rev. 6, 4 October 2011

Chapter 3
Sample application description

Sample application demonstrates the usage of API calls for all the MCAL modules.

The application software functionality:

- Initializes MCU module
- Initializes the PLL
    - PLL is configured to 60MHz
    - External oscillator frequency is 8MHz
- Checks whether PLL's are locked
- Activates the PLL clock to the MCU clock distribution
- Initializes PORT module. Pins configuration is show in section Table 3-1. PORT and DIO Modules - Pin Configuration and DioChannel Assignment
- Initializes the CAN driver and sample application specific data for this driver
- Initializes the LIN driver and sample application specific data for this driver
- Initializes the SPI driver and sample application specific data for this driver
- Initialize the ADC driver and sample application specific data for this driver
- Initialize the PWM driver and sample application specific data for this driver
- Initialize the ICU driver and sample application specific data for this driver
- Initialize DIO driver. The DIO driver is used to toggle LED3, LED2 and retrieve the value of KEY1.
- Initialize the GPT driver. On GPT notification the LED3 is toggled
- Initializes WDG driver and configures it to SLOW MODE
- Initialize FEE driver and sample application specific data for this driver. The FLASH driver is used indirectly by FEE driver. After initialization:
    - Erases memory block of 32 bytes
    - Writes 32 bytes pattern
    - Reads the memory block
    - Verifies the retrieved values if matches the values that were previous written

- Performs while loop
  - The WDG is triggered
  - If KEY1 is pressed the LED 2 is set ON and when the timeout expires, the LED2 is set OFF. If the timeout expires the Watchdog is configured again in SLOW MODE.
  - 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 LED1. Moving the trimmer has as effect modifying the intensity of LED1.
  - ICU is connected to the output of PWM. It is used to measure the PWM duty cycle, and check the value against the value read from ADC
  - If the value of the duty cycle retrieved by ICU matches the initial value read using ADC, the cycle restarts with the retrieving the ADC value
  - Performs a loopback transmission (exploiting CAN's hardware loopback capabilities) on the FlexCAN unit 0
  - LinFlex unit 0 issue a message on the LIN bus. The message itself can be monitored on the bus using various monitoring devices (oscilloscope, LIN analyzer)
  - 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

Connect the serial terminal to DB4 connector on the motherboard. The serial terminal should have the following settings: 115200, 8, N, 1

The detailed description of the LEDs and Buttons functionality is depicted in Table 3-1 and Table 3-2.

Table 3-1. 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
Led1	188	eMIOS_9	Out	Low	LED1	-
Led2	189	GPIO	Out	Low	LED2	Dio_Led2
Led3	190	GPIO	Out	Low	LED3	Dio_Led3
Led4	191	GPIO	Out	High	LED4	-
Key1	179	GPIO	In	Low	KEY1	Dio_Key1

Table 3-2. LEDs and Buttons Functionality

LEDs and Buttons	Functionality		
		LED ON	LED OFF
Led1	Indicates the output for PWM - the intensity varies depending on the duty cycle of PWM. The duty cycles is modified using the trimmer.	-	-
Led2	When KEY1 is pressed the led turns on, until the watchdog timeout	When KEY1 pressed and watchdog timeout did not expired	If KEY1 is not pressed or the watchdog timeout expired
Led3	Is toggled at each GPT notification	-	-
Led4	Set up using port to be OFF	-	-
Key1	Disables triggering the watchdog	-	-

Chapter 4 Installation Steps

4.1 Hardware Installation

The MPC56xx evaluation board and jumper configuration are shown in Figure 4-1. MPC563xM Evaluation Board. Connected XPC563M_144QFP Mini-Module Board contains socket for MPC5634M in 144QFP footprint.

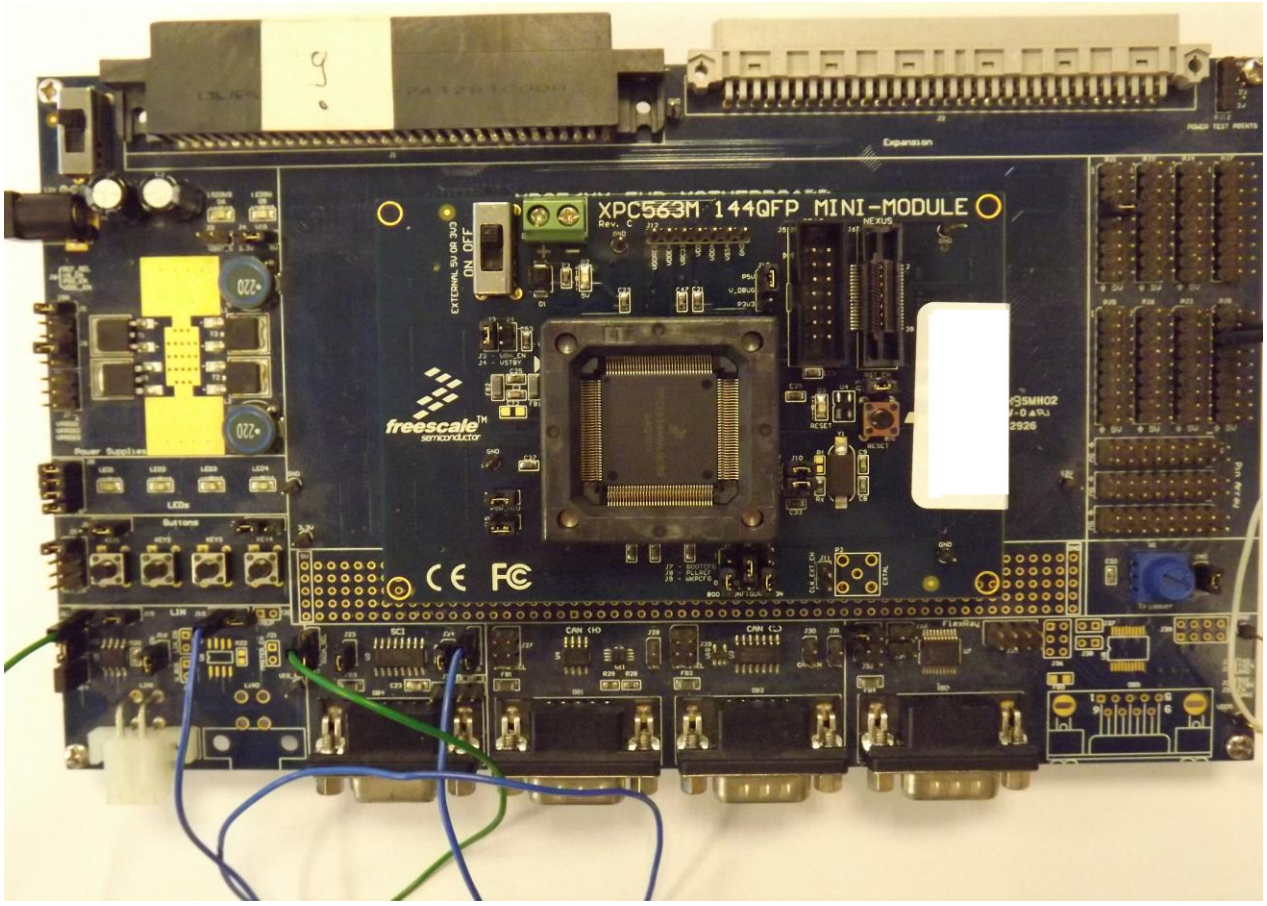

Figure 4-1. MPC563xM Evaluation Board

1. Please check whether the XPC563M_144QFP Mini-Module Board is populated with 8MHz oscillator (crystal Y1). If not, please follow instruction in chapter 5.4.1 Modifying the Configuration in Tresos Studio (especially the Figure 5-5. Modifying the External Crystal Frequency) to adjust the configuration.

2. Check all jumper configurations as shown in Figure 4-1. MPC563xM Evaluation Board and Figure 4-2. Buttons and LEDs Jumper Configuration
    a. J40 - on
    b. J5 - closed
    c. J4 - 5V
    d. J6 (5-6) closed
    e. J8 all closed
    f. J9 1-2 closed
    g. J9 3-4, 5-6, 7-8 open
    h. J10 – 0
    i. J11 – 1
    j. J13, J14, J15, J16 – closed
    k. J23, J24 closed
    l. J22 2-3 closed, J25 2-3 closed

3. Check all connections to be as follows:
    a. PJ1 -7 with PJ1 -8 –needed for SPI loopback
    b. PJ8 - 4 with PJ8-5 – needed for connection between ICU and PWM
    c. J12 –2 with J22-1 and J19-2 with J25 - 1 – needed for UART console support

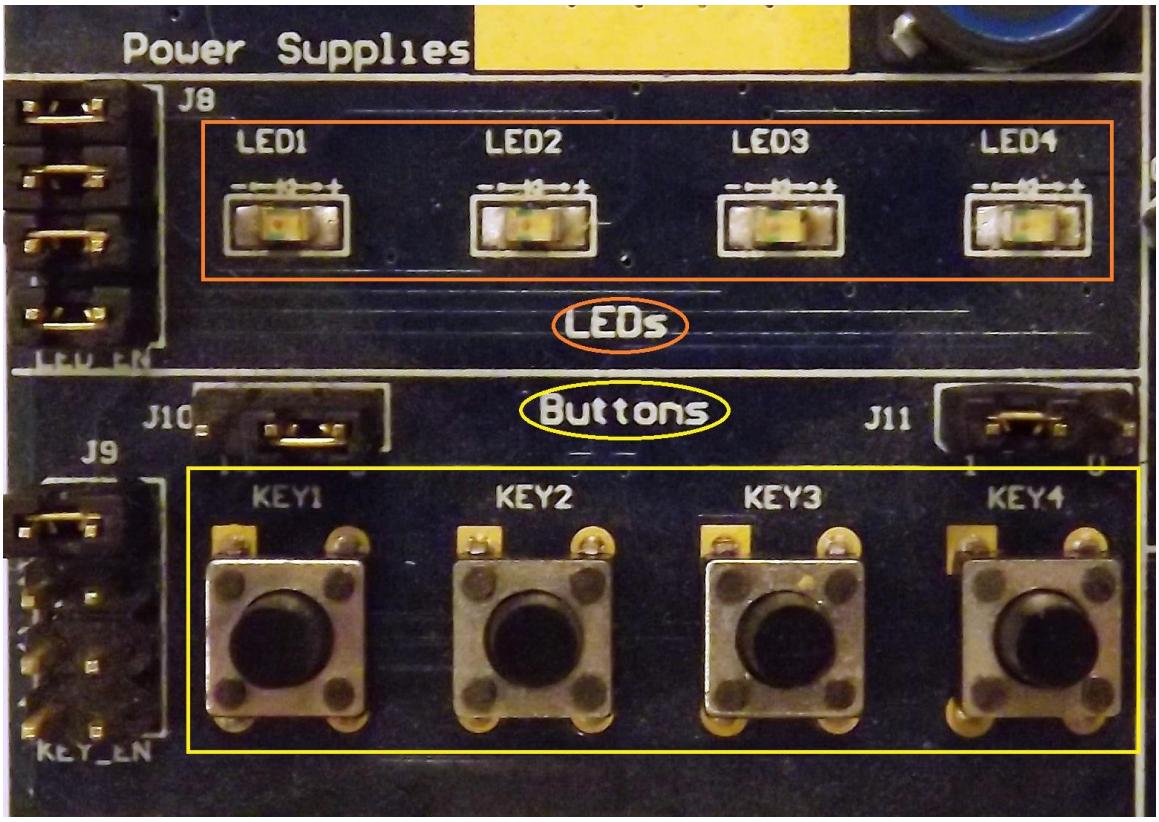

Figure 4-2. Buttons and LEDs Jumper Configuration

4.2 Software Installation

Please install the package on your computer. By installing the package, a folder tree as shown in Figure 4-3. Package Tree shall be created.

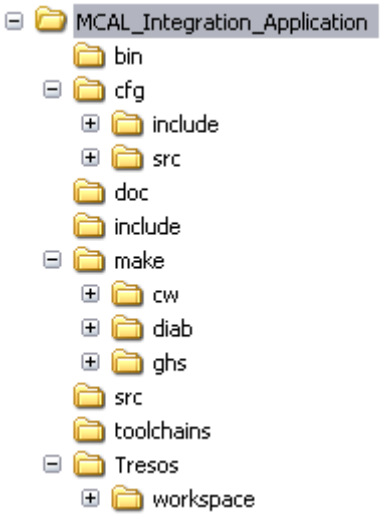

Figure 4-3. Package Tree

The package contains the following files and folders:

- *MCAL_Integration_Application* folder – contains the sample application
  - *bin* folder – generated object files and linker output files are stored into this folder
  - *cfg* folder – contains configuration files generated by Tresos tool
    - *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 MPC563xM device and types definition
- *make* folder – makefiles used for building the application
- *src* folder – contains main.c – the application source code file
- *toolchains* folder – files needed to build with various toolchains (startup, linker command files)
- *Makefile* file – 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 m*ake.bat* file

- *Tresos* folder – contains Tresos project with application configuration
  - *workspace/sample_app* folder - contains Tresos project
    - *output* folder – contains configuration files generated by Tresos tool
      - *include* subfolder – contains generated files with pre-compile configurations
      - *src* subfolder – contains generated files with post-build and link-time configurations.

4.2.1 Tresos Project Installation

The following procedure requires that the user has Tresos Studio at his disposal.

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 4-4. Import Window – the First View
   c. Next steps are depicted in Figure 4-5. Import Window - the Second View
      i. Select “Select root directory” and click on “Browse”
      ii. Select the location of the *sample_app* folder in installed sample application package folder
      iii. Select “Copy projects into workspace”

iv. Click on “Finish” button

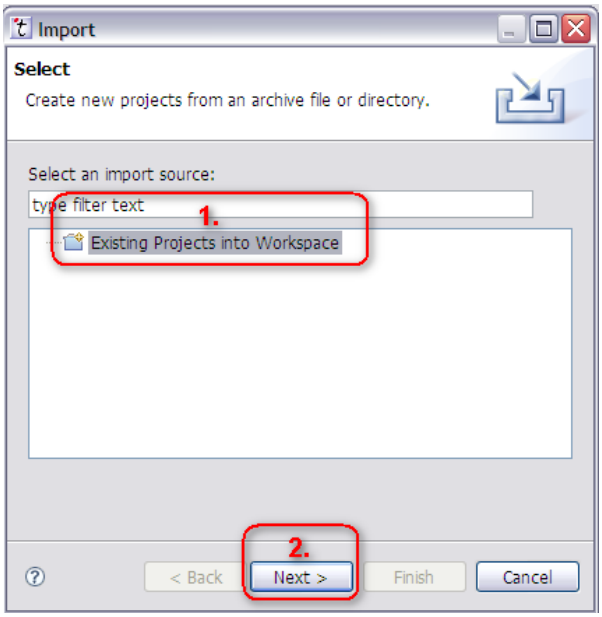

Figure 4-4. Import Window – the First View

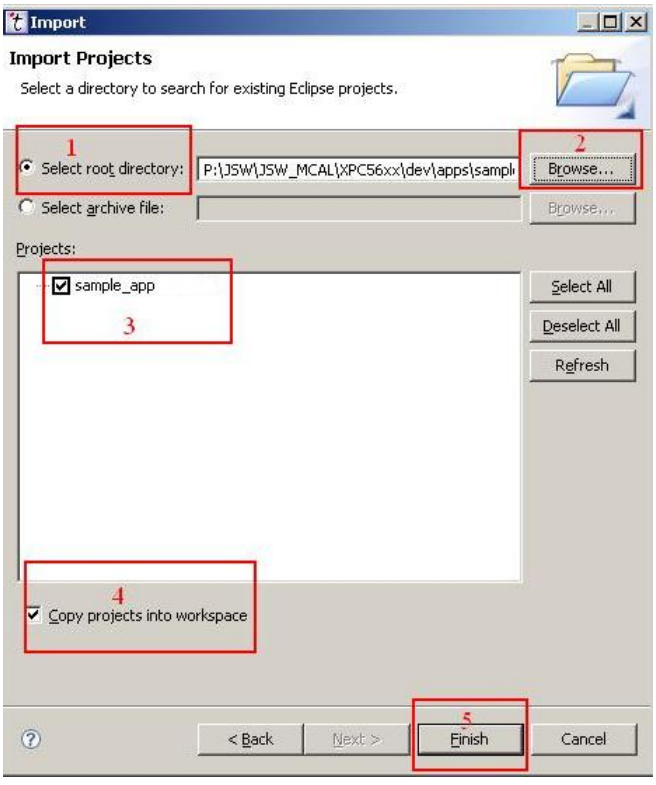

Figure 4-5. Import Window - the Second View

4.2.2 MCAL Application Configuration

The following procedure requires that the user has Tresos Studio at his disposal. Sample application can be compiled using GHS Multi version 5.2.4, CodeWarrior for MPC5xxx V2.8 (compiler version 4.3 build 209) or Windriver DIAB 5.8.0.0 (with the cumulative patch 5.8.0.0-2_WIND00198363).

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

- GHS_DIR for GreenHills Multi
  - Ex: SET GHS_DIR=C:\ghs\multi524
- CW_DIR for CodeWarrior
  - Ex: SET CW_DIR=C:\progra~1\Freesc~1\codewa~2.8
- DIAB_DIR for Windriver DIAB
  - Ex: SET DIAB_DIR=C:\WindRiver\diab\5.8.0.0-2

NOTE

The path to the tool chain 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 Tresos Studio location in the *TRESOS_DIR* parameter as shown in Figure 4-6. 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 CW_DIR or DIAB_DIR)

```
@echo off

:: uncomment line below if you do not set TRESOS_DIR over environment
SET TRESOS_DIR=E:\EB\tresos_20100409-release2010a-sr4

::OS Path
SET SSC_ROOT=E:\Freescale\AUTOSAR\os_freescale_mpc56xxAM

::Compiler Path
SET CW_DIR=E:\TOOLS\cw\CW_2.8_b209
SET DIAB_DIR=E:\TOOLS\WindRiver\DIAB_5_8_0_0-2_wind00198363_20100511_123238\diab\5.8.0.0
SET GHS_DIR=E:\TOOLS\ghs\multi524


if not defined TRESOS_DIR GOTO missing_path_names_TRESOS_DIR
if not exist %TRESOS_DIR% GOTO missing_path_names_TRESOS_DIR

cmd /c "make.bat %1 %2 %3 %4 %5 %6"

GOTO end

:: The variable TRESOS_DIR was not specified
:missing_path_names_TRESOS_DIR
ECHO.
ECHO    The variable TRESOS_DIR was not set correctly.
ECHO    Please specify this variable manually, by editing
ECHO    the batch file 'launch.bat'.
ECHO.
pause
GOTO end

:end
```

Figure 4-6. Configuration of the Tresos Studio Location

Chapter 5
Building Sample application

This section describes the build procedure.

5.1 Building Sample

To build the sample application, execute the following steps

Procedure:

1. Open the Windows command prompt window
2. Change the current directory to *MCAL_Integration_Application* folder
3. To build the sample, execute the following command to run *launch.bat* file as shown in Figure 5-1. Building the Sample - Launch Command

   ```
   launch.bat
   ```

4. The object files and linker output file (*sample_app_mcal.elf*) shall be generated into the *MCAL_Integration_Application/bin* subdirectory
5. To execute the sample application load the executable file placed in *MCAL_Integration_Application/bin* subdirectory to the evaluation board using Lauterbach debugger and run.cmm or run_ram.cmm script or MPC5500 Flash Utility.

NOTE

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

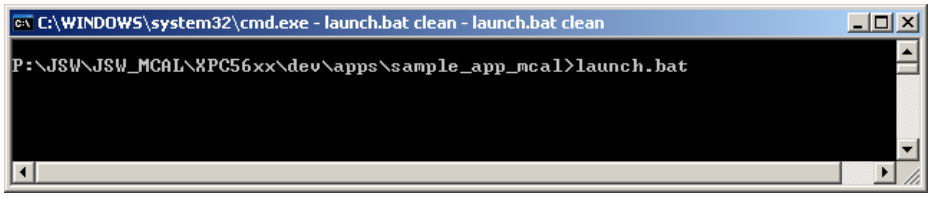

Figure 5-1. Building the Sample - Launch Command

5.2 Building with different compilers

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

TOOLCHAIN=tool_chain

Where tool_chain can have the values:

- ghs – default – use the GreenHills Multi compiler
- cw – use the Freescale CodeWarrior compiler
- diab – use the Windriver DIAB compiler

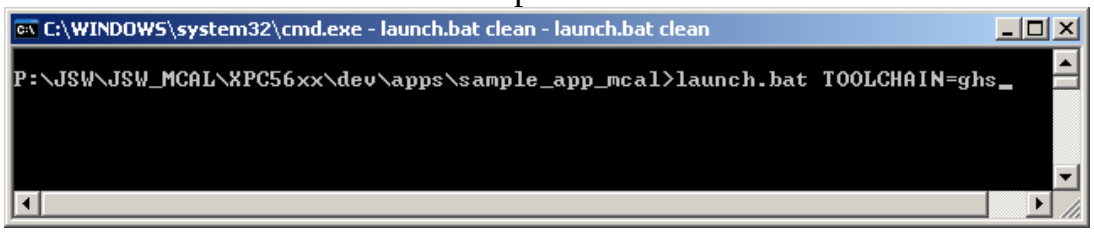

Figure 5-2. Building with different compilers - selecting GreenHills

5.3 Clean Object and Linker Output Files

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

Procedure:

1. Open the Windows command prompt window
2. Change the current directory to *MCAL_Integration_Application* folder
3. To clean the object and linker output files, execute the following command to run *launch.bat* file as shown in Figure 5-3. Cleaning the Sample - Launch Clean Command

   ```
   launch clean
   ```

4. The object files and linker output files shall be cleared from the *MCAL_Integration_Application/bin* and from the *MCAL_Integration_Application* folders.

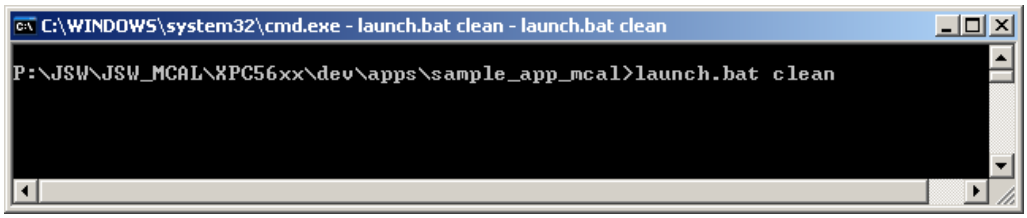

Figure 5-3. Cleaning the Sample - Launch Clean Command

5.4 Modifying the Application

Users may change the application according to their requirements.

5.4.1 Modifying the Configuration in Tresos Studio

Procedure:

1. Open the Tresos Studio GUI
2. Open previously imported *sample_app* 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 5-5. Modifying the External Crystal Frequency
4. Select the *sample_app* project and click on “Generate” button to generate the configuration files as shown in Figure 5-4. Configuration Generation
5. Copy generated pre-compile configuration files which are used in application from *workspace/sample_app/output/include* directory into the sample application folder *MCAL_Integration_Application/cfg/include.*
6. Copy generated post-build and link-time configuration files which are used in application from
   workspace/sample_app /output/src directory into the sample application folder *MCAL_Integration_Application/cfg/src.*

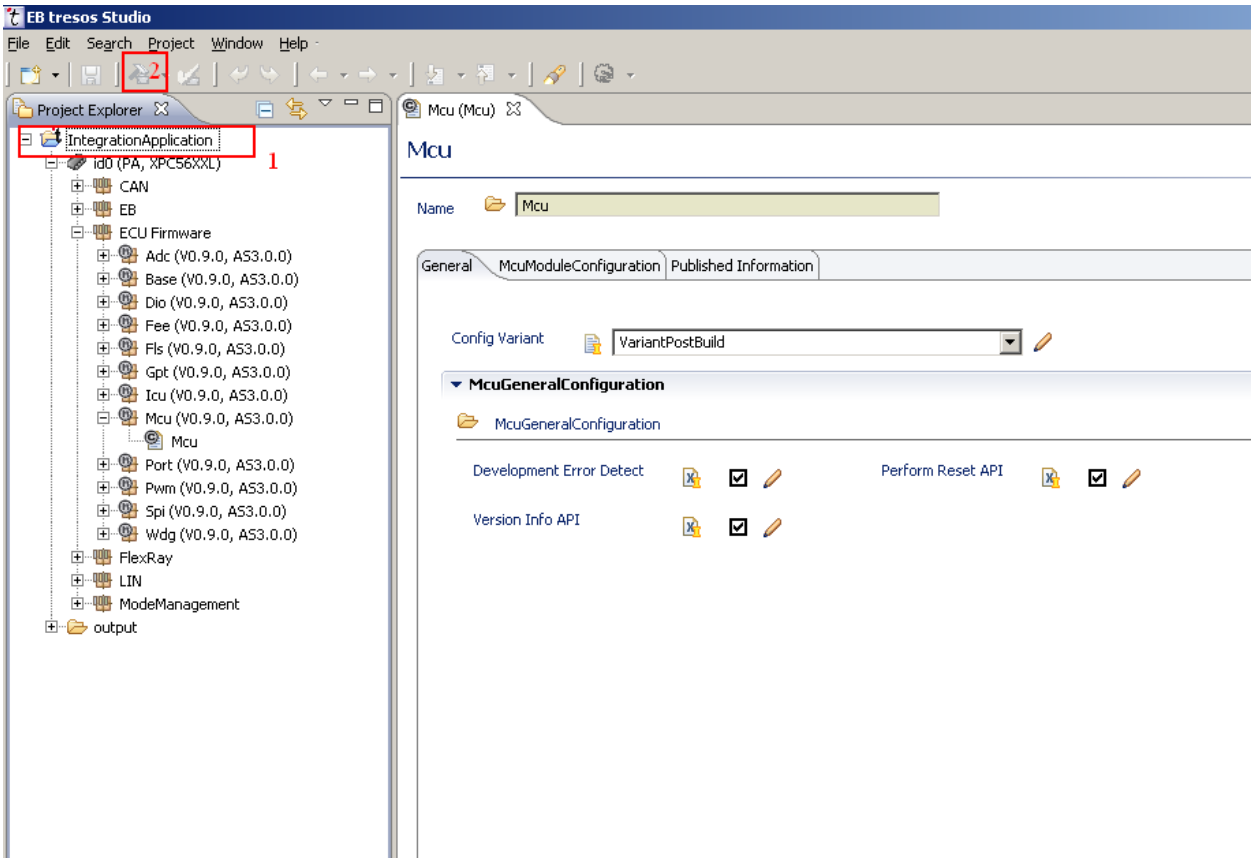

Figure 5-4. Configuration Generation

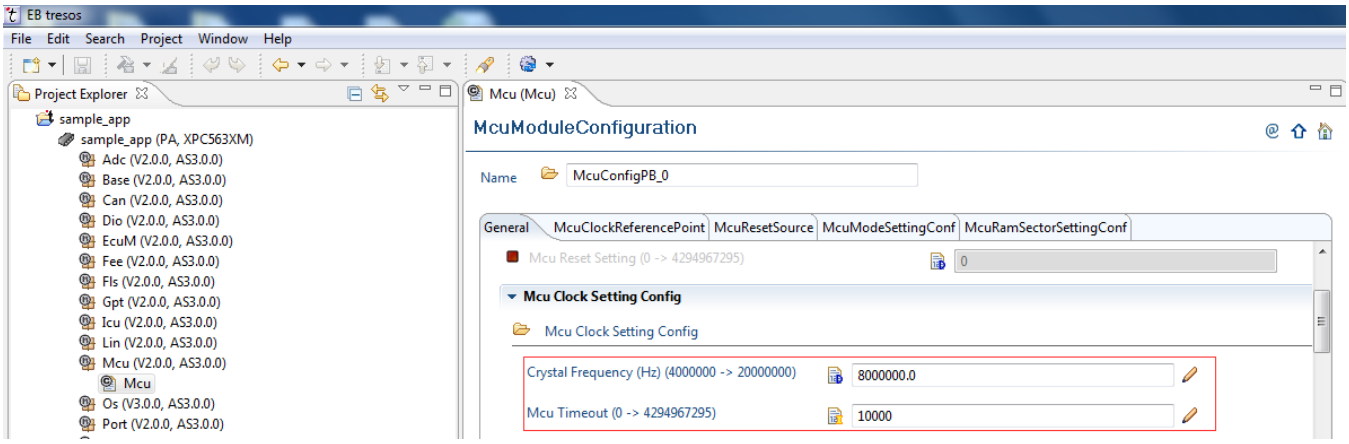

Figure 5-5. Modifying the External Crystal Frequency

NOTE

The value of the *Loops TimeOut* parameter needs to be configured appropriately, otherwise the MCU is not able to start-up. Recommended value for crystal with frequency 8MHz is 10000.

How to Reach Us:

Home Page:

www.freescale.com

Web Support:

http://www.freescale.com/support

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

AUTOSAR and AUTOSAR logo are registered trademarks of AUTOSAR GbR (www.autosar.org)

Rev. 1.1

freescale™ semiconductor