

Processor Expert Software – Microcontrollers Driver Suite

Getting Started Guide

This document introduces Microcontrollers Driver Suite tool. The document describes how to create a new Processor Expert project, create a simple *HelloWorld* application and use the code generated by Processor Expert by a third party C compiler.

Contents

1	Introduction.....	1
2	Creating New Processor Expert Project	2
3	Creating Simple Application.....	8
4	Using Generated Code	15
5	Importing Example Projects.....	18

1 Introduction

Microcontrollers Driver Suite is a rapid application design tool targeted for Freescale Kinetis and ColdFire+ Microcontrollers providing the following key features:

- A Graphical User Interface which allows an application to be specified by the functionality needed.

- An application created from Embedded Components encapsulating initialization and functionality of basic elements of embedded systems.

- An automatic code generator which creates tested and optimized C code which is tuned to your application needs and the selected Freescale device.

A built-in knowledge base, which immediately flags resource conflicts and incorrect settings, so errors are caught early in design cycle allowing you to get to market faster with a higher quality product.

Processor Expert creates a Hardware Abstraction Layer (HAL) which provides a unified application programming interface (API) allowing easy migration between Freescale devices.

The application created in Processor Expert is built from the building blocks called the Embedded Components and the CPU components. The CPU components are special type of Embedded Components. An Embedded Component provides:

- Selection of peripheral module, pins and timing used by the selected component
- Initialization of selected peripheral module according to the current setting of the selected component
- Methods to interface component functionality
- Events to handle hardware or software events related to the component

The Processor Expert tool can generate code for IAR C Compiler, CodeWarrior C Compiler or GNU C Compiler:

The tool can generate code for IAR C Compiler, CodeWarrior C Compiler, Arm Keil or GNU C Compiler.

For the Kinetis devices, the following compilers are supported:

- CodeWarrior for MCU, 10.1 and higher.
- IAR ARM compiler 6.3 and higher
- GNU C Compiler for ARM architecture
- Arm Keil C/C++ compiler 5.4 and higher

For the ColdFire+ devices, the following compilers are supported:

- CodeWarrior for MCU, 10.1 and higher.
- IAR ColdFire C compiler 1.2
- GNU C Compiler for ColdFire architecture

The Microcontrollers Driver Suite is implemented as Eclipse plug-in that can be installed in an existing Eclipse environment (3.6.x or 3.7.x). For installation instructions, see Installation Guide (*PEXDRVSINSTALLUG.pdf*).

2 Creating New Processor Expert Project

The Microcontrollers Driver Suite provides two options for creating a new Processor Expert project. You can either create an empty project directly using the **New Processor Expert Project** Wizard or you can extend an already existing Eclipse project by the Processor Expert project.

2.1 Using New Project Wizard

To create a new Processor Expert project:

1. Run the Eclipse environment.

2. Select **File > New > Processor Expert Project** from the IDE menu bar. The **Create a Processor Expert Project** screen of the **New Processor Expert Project Wizard** appears (Figure 1).
 1).
3. Type a name for the project in the **Project name** text box. For example, *HelloWorld* (Figure 1).

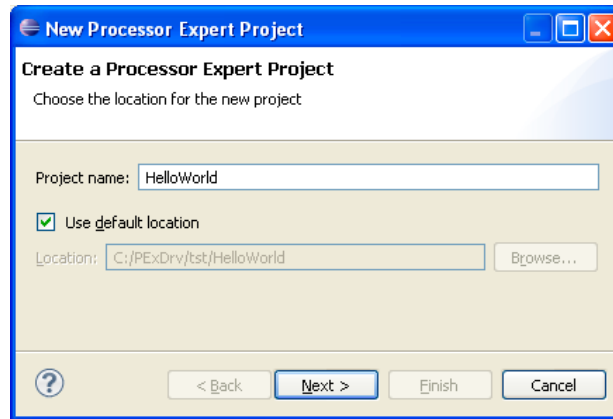


Figure 1. New Processor Expert Project Wizard - Create a Processor Expert Project Screen

4. To specify a different location for the new project, uncheck the **Use default location** checkbox, and click **Browse**. The default setting of the **Use default location** checkbox is checked.
5. Click **Next** to display the **Devices** screen as shown in Figure 2.

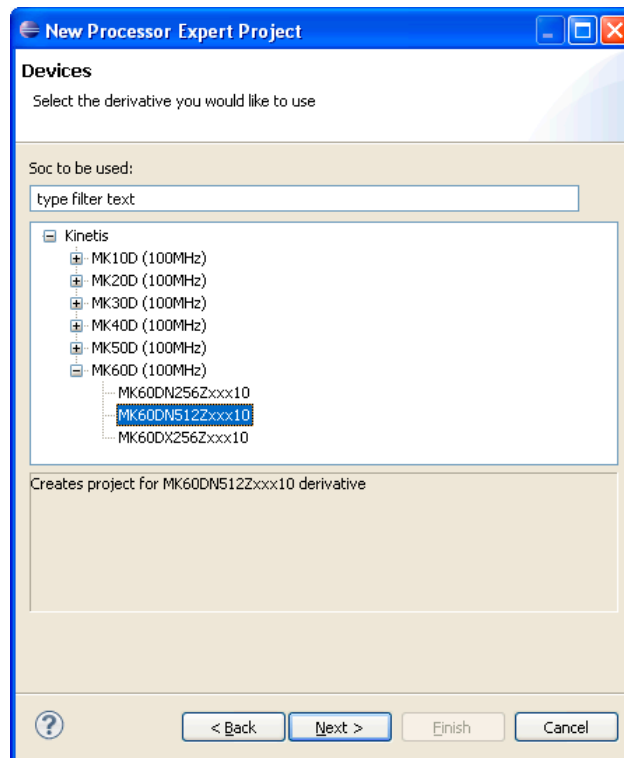


Figure 2. New Processor Expert Project Wizard– Devices Screen

6. Select the target derivative to be used for the project and click **Next** to display the **Rapid Application Development** screen (Figure 3).

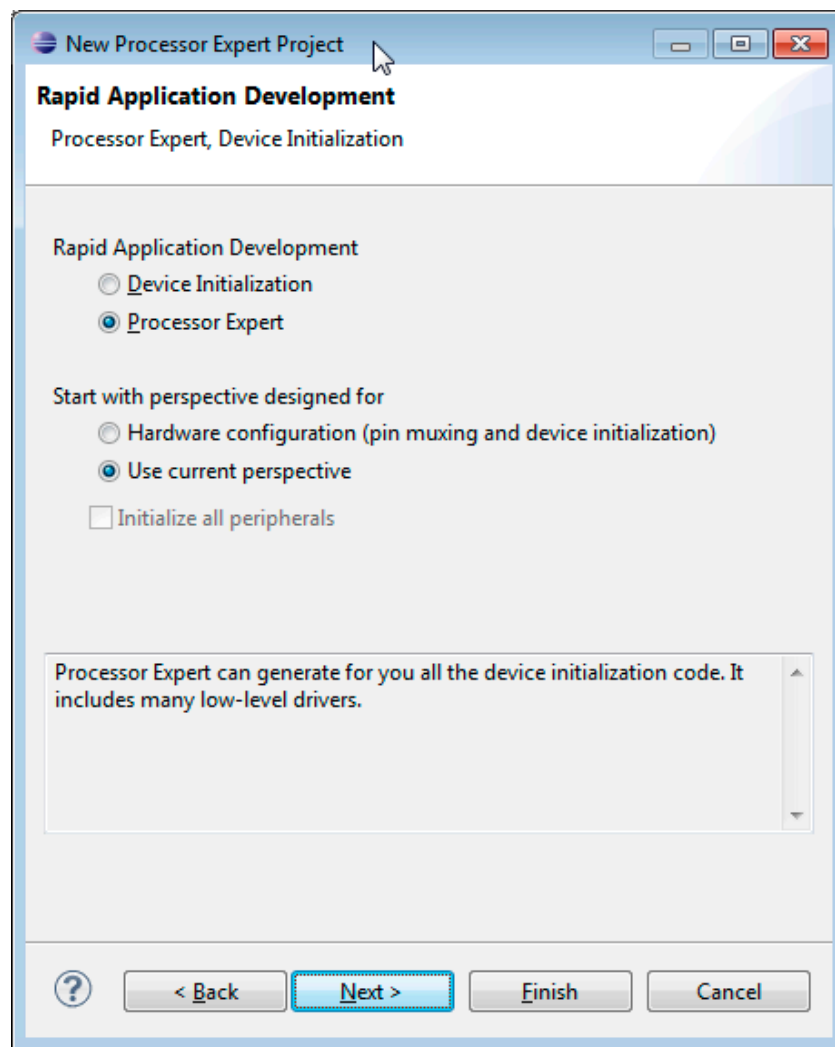


Figure 3. New Processor Expert Project Wizard - Rapid Application Development Screen

7. Select the type of project, **Processor Expert** or **Device Initialization**.
8. Click **Next** to display the **Processor Expert Target Compiler** screen shown below. This screen lets you select the target C compiler for which the project will generate code (the target compiler can be later changed in the 'Build options' tab of the CPU component).

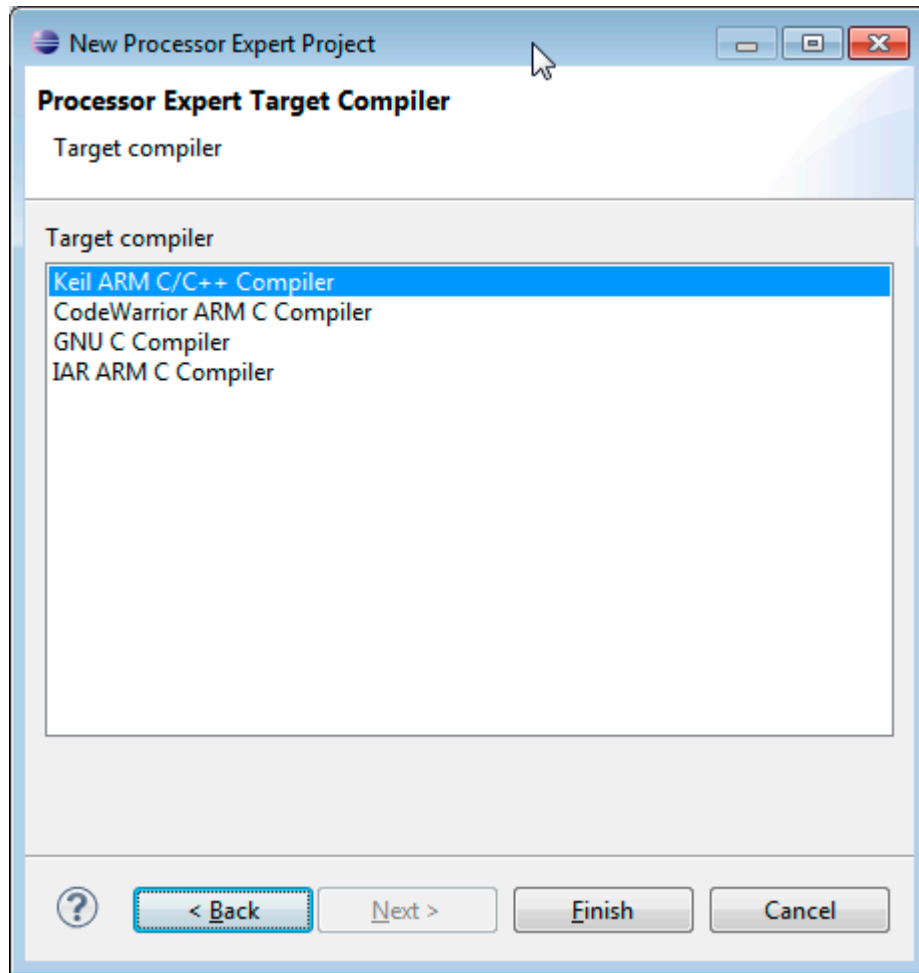


Figure 4. New Processor Expert Project Wizard - Processor Expert Target Compiler Screen

9. Select the required target C compiler and click **Finish** to create the Processor Expert project. The Processor Expert perspective is displayed as shown in figure below. The project *HelloWorld* is created and appears in the **Project Explorer** view. The other views of the Processor Expert perspective are also shown below.

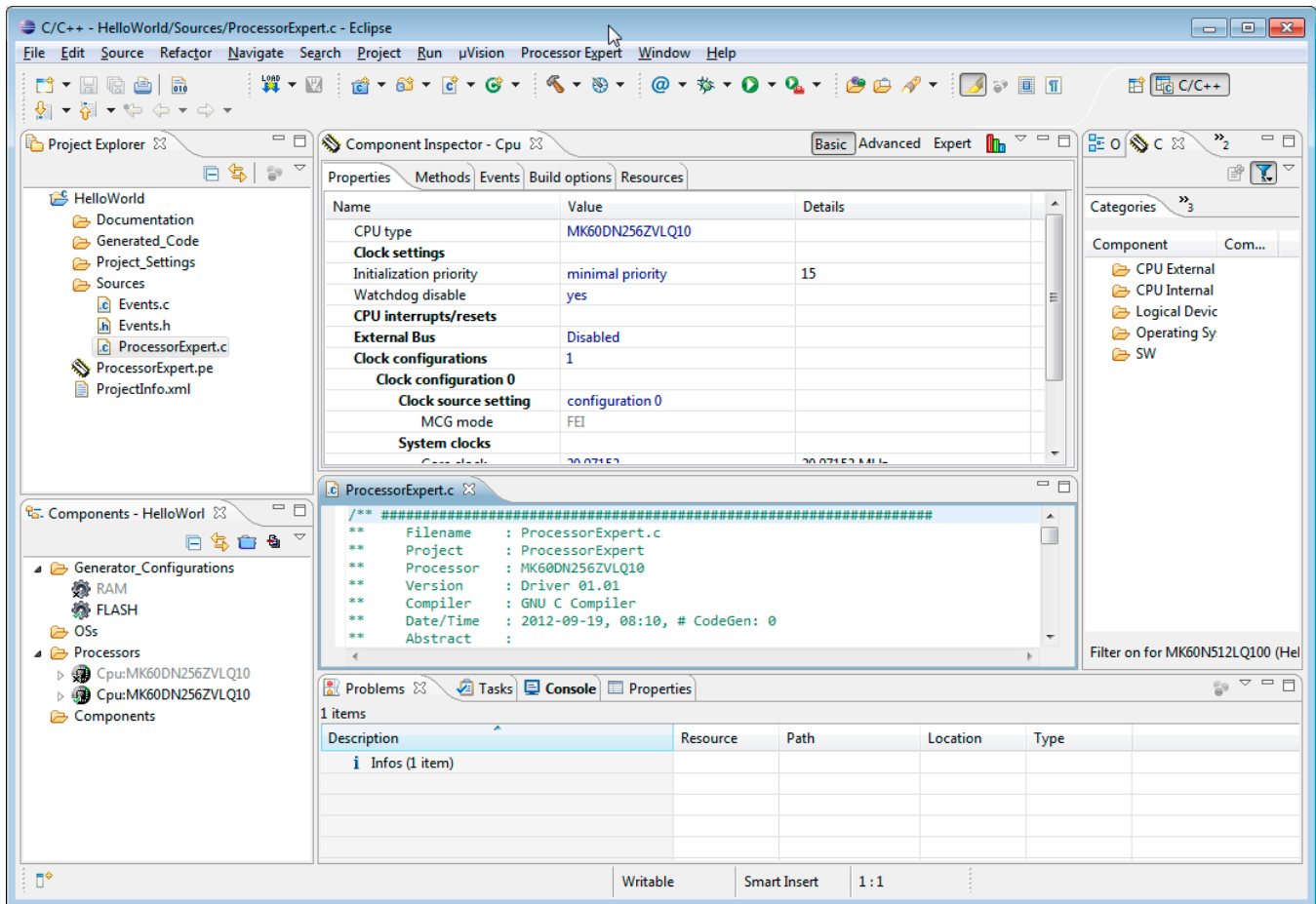


Figure 5. Processor Expert Perspective

The main views in the Processor Expert perspective are:

- Project Explorer
- Components
- Component Inspector
- Components Library
- Problems

For more information on the Processor Expert perspective and its views in detail, refer the *Microcontrollers Driver Suite Processor Expert User Manual*.

2.2 Adding Processor Expert Project to Existing Project

If an Eclipse project is already created in the Eclipse Environment (for example, a project for a third party Eclipse plugin), it is possible to add the Processor Expert support to use generated code in this project.

1. Run Eclipse environment.
2. Open the Eclipse project in which you want to add the Processor Expert support.

3. Select **File > New > Other** from the IDE menu bar. The **Select a Wizard** screen appears.

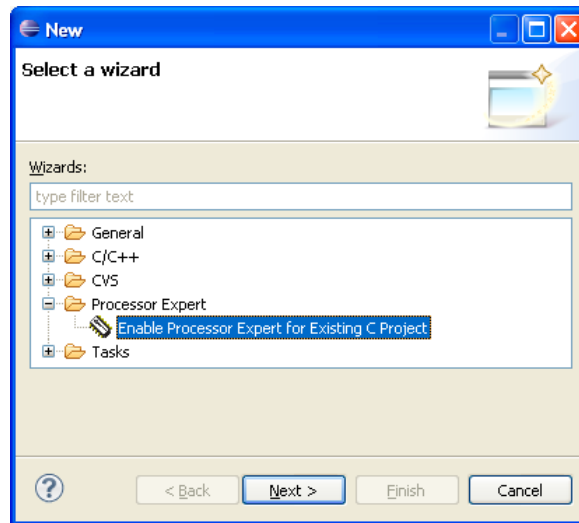


Figure 6. Select a Wizard Screen

4. Click **Next** to open the **Processor Expert Project File** screen as shown in Figure 7.
5. Click **Browse** to select the project to which the Processor Expert project will be added.
6. Select **Project Type** as Processor Expert or Device Initialization.

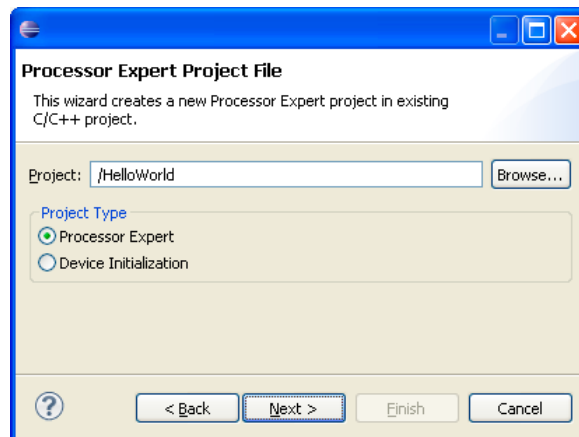


Figure 7. Processor Expert Project File Screen

7. Click **Next** to open the **Target MCU** selection screen as shown below. This screen lets you select the target derivative to be used for the project.

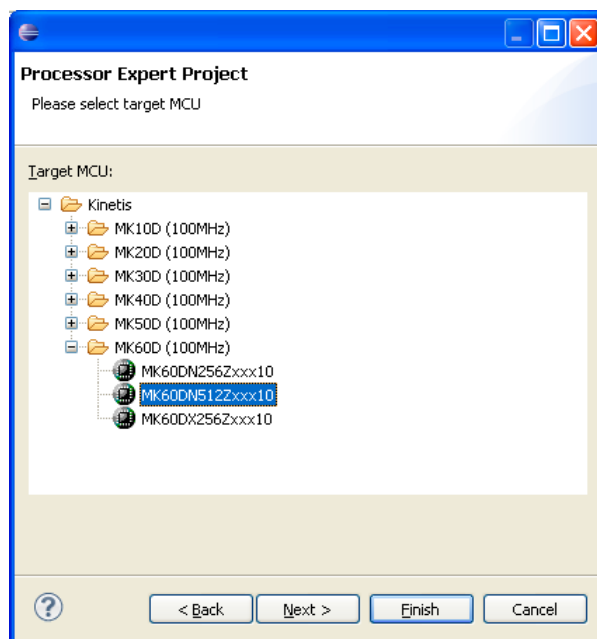


Figure 8. Target MCU Select Screen

8. Click **Next** to display the **Processor Expert Target Compiler** screen shown in Figure 4. This screen lets you select the target C compiler for which the project will generate code (the target compiler can be later changed in the 'Build options' tab of the CPU component).
9. Click **Finish** to create Processor Expert project. The Processor Expert perspective is displayed as shown in Figure 5.

NOTE

The *ProcessorExpert.c* file placed in the **Sources** folder contains definition of the `main()` function. If you are adding Processor Expert to a project which already contains user code, it may be necessary to modify the code. Some parts of the existing user code may also conflict with the code generated by Processor Expert, for example, definition of the interrupt vector table. It is recommended to back up the whole project before the conversion.

3 Creating Simple Application

This section shows creation of a simple application, which blinks a LED and sends *Hello World* text to the serial output.

NOTE

The project is designed to work with MK60DN512ZVMD10 CPU and Tower System Kit TWR-K60N512-KIT. However, it is not necessary to have this hardware; the project can be created without it.

3.1 Adding Components to Project

The *HelloWorld* application consists of four components:

CPU component MK60DN512ZVMD10

GPIO_LDD which will control LED output, connected to PTA10 pin

TimerUnit_LDD which will provide periodical timing.

Serial_LDD which will send *Hello World* message to the serial output

To create the *HelloWorld* application:

1. Create a new project for Kinetis MK60DN512ZVMD10 derivative. The new project will have the CPU component automatically added when the project is created. See [Creating New Processor Expert Project](#).
2. To add components, select the **Alphabetical** tab in the **Components Library** window as shown below.

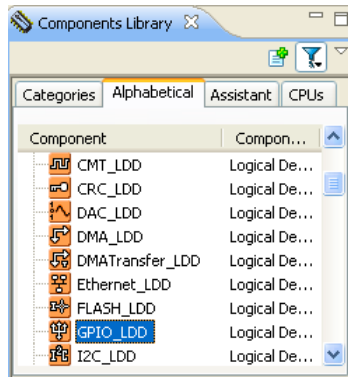


Figure 9. Components Library - Alphabetical Tab

3. Find the GPIO_LDD component and double-click on the component to add it to the project.
4. Find the TimerUnit_LDD component and double-click on the component to add it to the project.
5. Find the Serial_LDD component and double-click on the component to add it to the project. All the components will be visible now in the Component view.

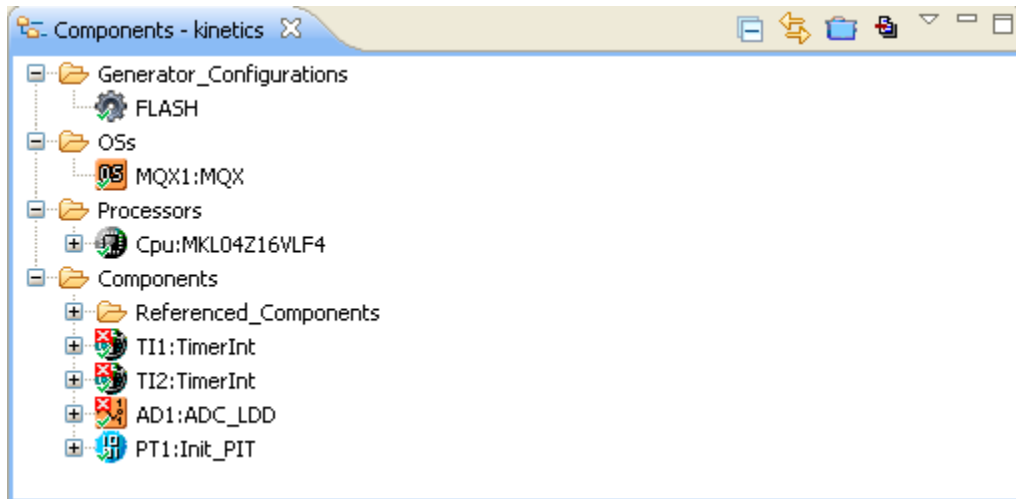


Figure 10. Component View – List of Components

3.2 Configuring Components

The components are configured by defining/changing the values of the properties and enabling/disabling methods and events. The **Component Inspector** view is used for accessing the component properties:

1. To configure **CPU** component, click on this component in the **Component** view to display **Component Inspector**.
2. To match the CPU component settings to the parameters of the TWR-K60N512 board, set the following properties:

System oscillator to *Enabled*

System oscillator > Clock source to *External reference clock*

System oscillator > Clock source > Clock frequency to *50.0*

Clock source settings > Clock source setting 0 > MCG settings > MCG mode to *PEE*

Clock configurations > Clock configuration 0 > System clocks > Core clock to *48*

Clock configurations > Clock configuration 0 > System clocks > Bus clock to *48*

Clock configurations > Clock configuration 0 > System clocks > External bus clock to *24*

Clock configurations > Clock configuration 0 > System clocks > Flash clock to *24*

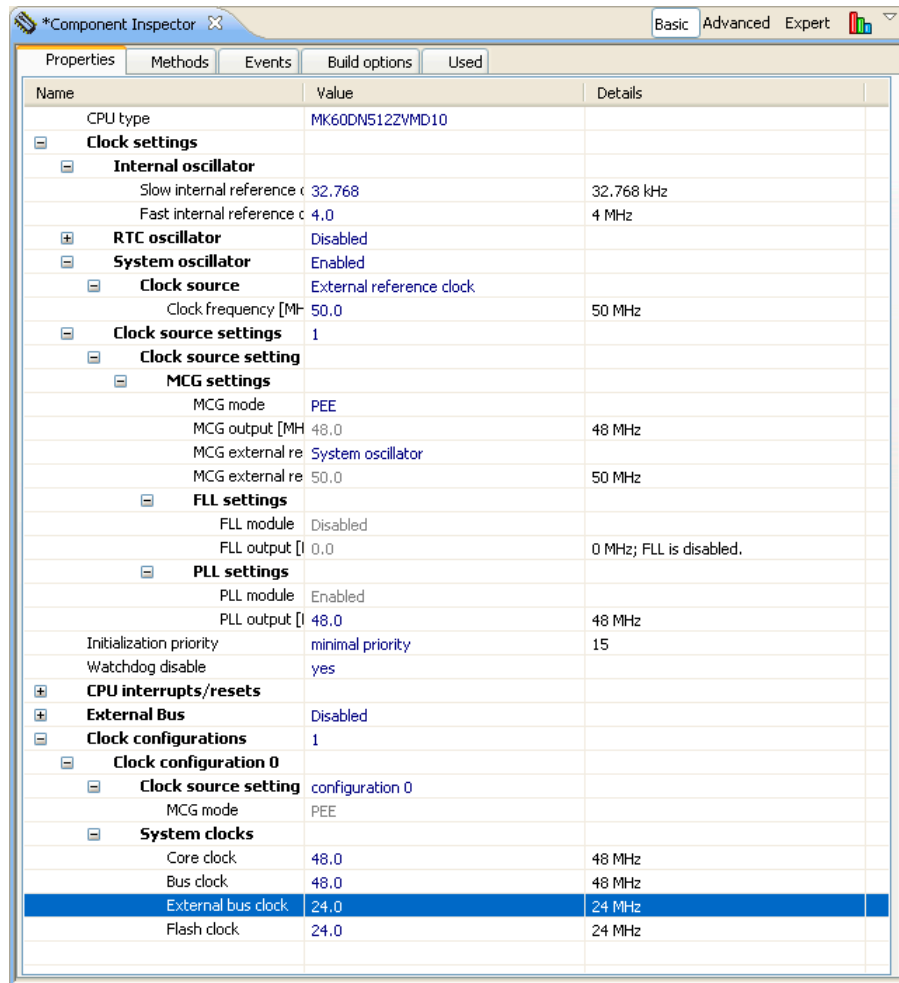


Figure 11. Component Inspector - CPU

- To configure **GPIO1** component, click on this component in the **Component** view to display **Component Inspector**.

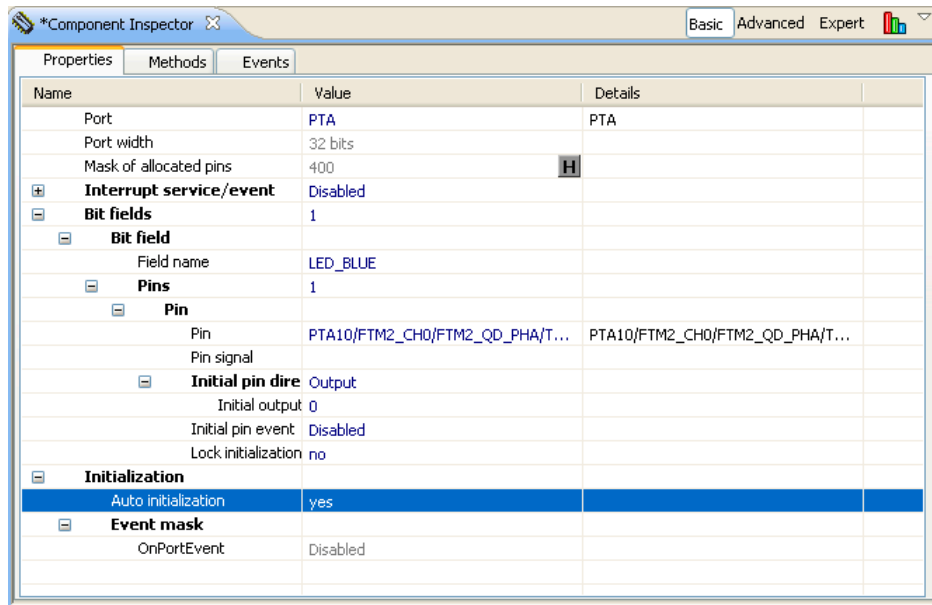


Figure 12. Component Inspector – GPIO1

- Set the following properties:
 - Field name** in the first **Bit field** group to *LED_BLUE*.
 - Pin** to *PTA10*. This pin corresponds to blue LED on the TWR-K60N512 board.
 - Initial pin direction to *Output*.
 - Auto initialization** to *yes* to automatically initialize the component during startup.
- To configure **TU1** component, click on this component in the **Component** view to display the **Component Inspector**.

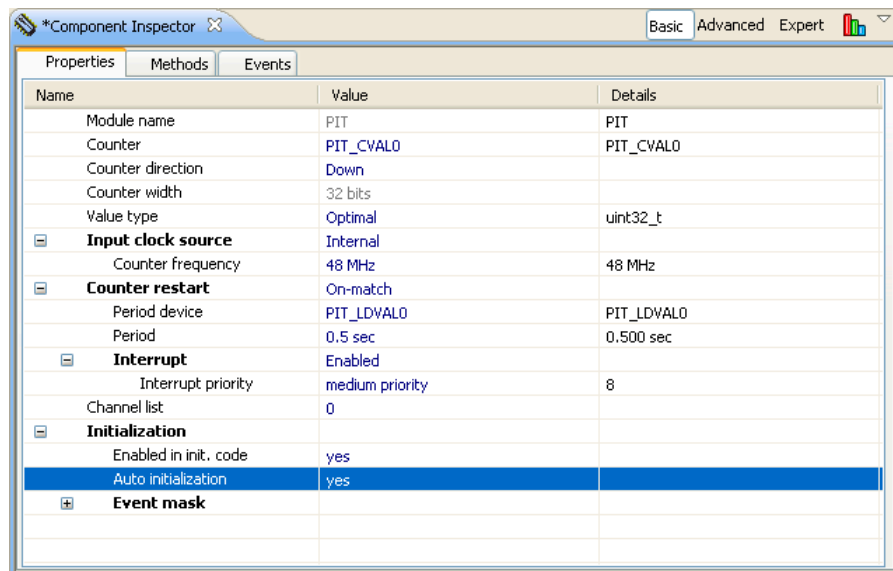


Figure 13. Component Inspector - TU1

- Set the following properties:

Counter to *PIT_CVAL0*

Counter direction to *Down*

Counter restart to *On-match*. This will allow setting desired period of interrupt, otherwise interrupt invocation period is fixed to counter overflow.

Counter frequency - Use the available button and select the only value offered in the right pane: *48 MHz*.

Counter restart > Period - Click on the available button and set *Init. value* to *0.5 sec*

Interrupt to *Enabled*

Auto initialization to *yes*

- To configure **AS1** component, click on this component in the **Component** view to display **Component Inspector**.

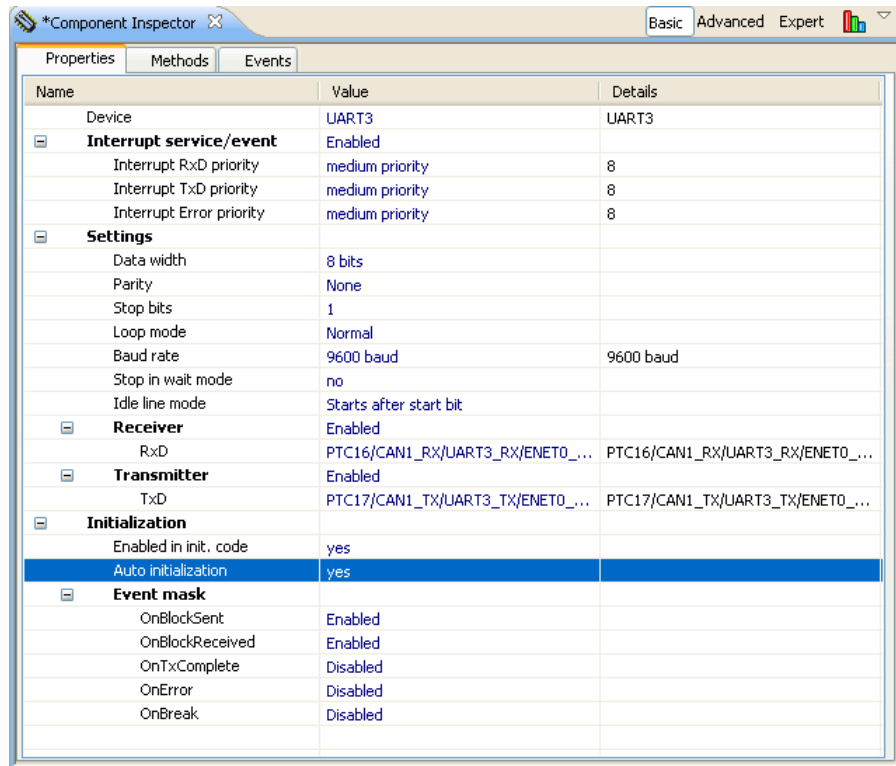


Figure 14. Component Inspector - AS1

- Set the following properties:

Device to *UART3* (the Tower serial board is connected to the UART3 port on the TWR-K60N512 board, using pins PTC16 and PTC17)

Baud rate - Click on the available button and set **Init. value** to *9600 baud*

RxD to *PTC16*

TxD to *PTC17*

Auto initialization to *yes*

3.3 Generating Code

Select **Project > Generate Processor Expert Code** in the IDE menu to generate code. This process generates source files for components to the **Generated_Code** folder in the **Project Panel** view. Other modules (*ProcessorExpert.c* and *Events.c*) can be found in the **Sources** folder.

3.4 Writing Code

The Processor Expert components provide API, which is used by user code. Initially, you can put the code into the *ProcessorExpert.c* file, which is automatically created by Processor Expert:

1. In the **Project Explorer** view, expand the **Sources** folder and double-click on the **ProcessorExpert.c** module to open the file in the editor.

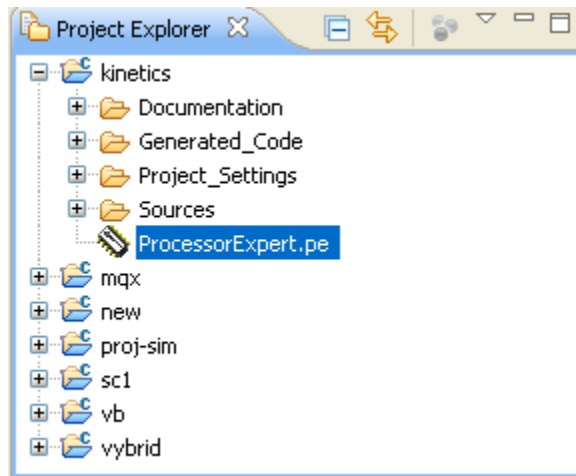


Figure 15. Project Explorer - Main Module

2. In the *ProcessorExpert.c* file, do the following modifications:

Declare global variable with the message:

```
char HelloWorld[] = "Hello world";
```

In the `main()` function after `PE_low_level_init();` call the `SendBlock` function to send the message:

```
AS1_SendBlock(AS1_DeviceData, HelloWorld, sizeof(HelloWorld)-1);
```

```

*ProcessorExpert.c
#include "PE_Const.h"
#include "IO_Map.h"

/* User includes (#include below this line is not maintained by Processor
char HelloWorld[] = "Hello world";

void main(void)
(
/* Write your local variable definition here */

/** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!!
PE_low_level_init();
/** End of Processor Expert internal initialization.

/* Write your code here */
AS1_SendBlock(AS1_DeviceData, HelloWorld, sizeof(HelloWorld)-1);

/** Don't write any code pass this line, or it will be deleted during c
/** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! **
for(;;){}
/** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! **
/** End of main routine. DO NOT MODIFY THIS TEXT!!! **/

```

Figure 16. ProcessorExpert.c File

3. In the **Component** view, expand the list of events and methods of **TU1** component.
4. Right-click on **TU1_OnCounterRestart** and select the **View Code** command from the pop-up menu to open the code of **TU1_OnCounterRestart** in the editor.

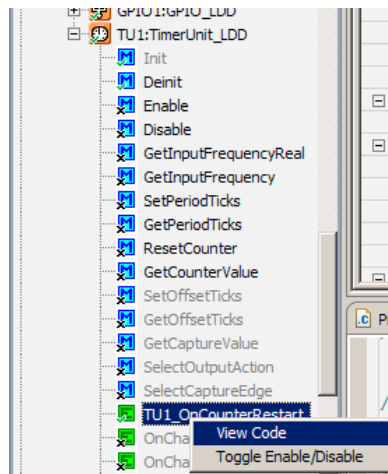


Figure 17. View Code

5. Insert the following lines (toggling LED output) into the TU1_OnCounterRestart event handler function body:

```
GPIO1_ToggleFieldBits(GPIO1_DeviceData, LED_BLUE, 1);
```

The application is complete and can be compiled and run using a third party toolchain.

4 Using Generated Code

Processor Expert uses the following sub-directory structure within the project directory:

Generated_Code – the directory containing all generated source code modules for components.

Documentation – the directory with the project documentation files generated by Processor Expert.

Sources – the directory for main module, event module and other user modules.

Project_Setings\Linker_Files – the directory containing linker file generated by Processor Expert

Project_Setings\Startup_Code – the directory containing the Processor Expert startup code (compiler dependent)

4.1 Project Include Path

Set the following paths in the settings of the target IDE to correctly use the code generated by Processor Expert with a third party C compiler:

```
{ProjectDirectory}\Sources
{ProjectDirectory}\Generated_Code
```

For Kinetis family, set the following paths:

```
{ProcessorExpertDirectory}\lib\Kinetis\iofiles
{ProcessorExpertDirectory}\lib\Kinetis\pdd\inc
```

For ColdFire+ family, set the following paths:

```
{ProcessorExpertDirectory}\lib\ColdFirePlus\iofiles
{ProcessorExpertDirectory}\lib\ColdFirePlus\pdd\inc
```

The default location of the Processor Expert directory depends on the operating system and user rights. By default, the Processor Expert directory is created as a subdirectory of the Eclipse installation:

```
{eclipse}\ProcessorExpert
```

The actual location can be found in the preferences of Processor Expert. Select **Window > Preferences > Processor Expert > Processor Expert Service** to view the actual location.

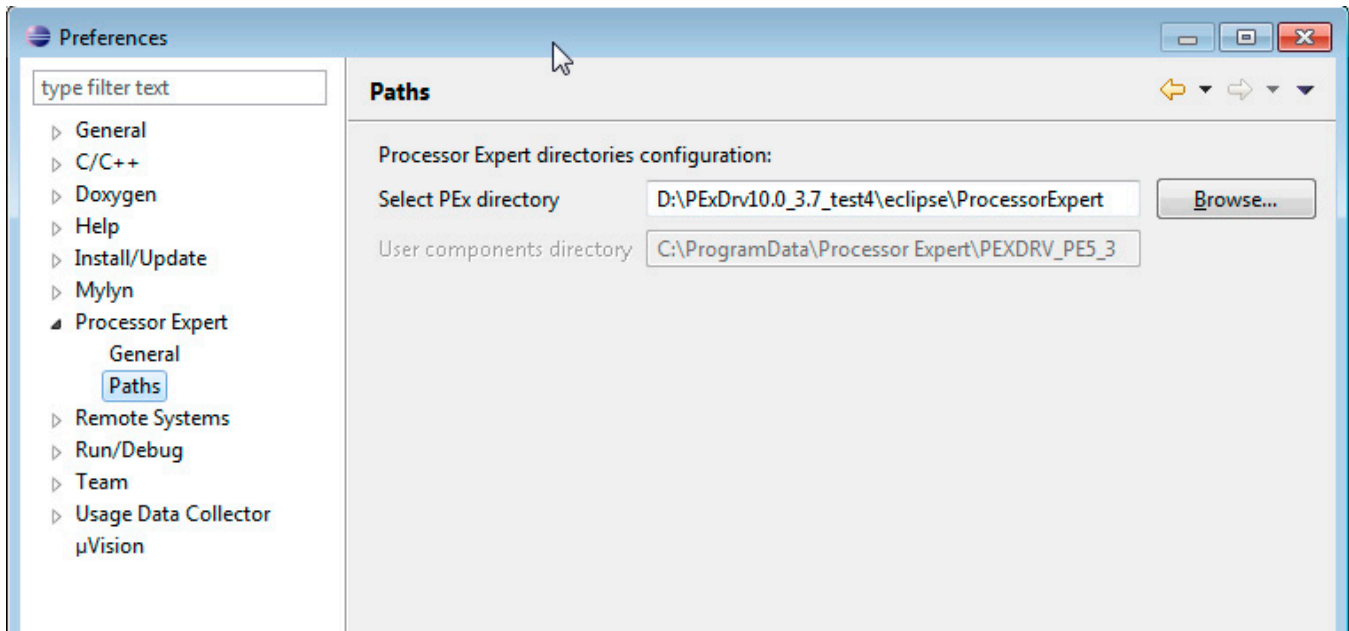


Figure 18. Processor Expert Directory Location

4.2 Project Linker File

Linker file specific for the selected target CPU and compiler is generated by Processor Expert. The file is stored at the following location: `{ProjectDirectory}\Project_Settings\Linker_Files`

The name of the linker file is ProcessorExpert, extension of the linker file is compiler specific.

The code generated by Processor Expert components is tailored to the selected target compiler. Make sure that the selected compiler matches used build tools. The target compiler is selected during project creation. It can be later changed in the **Build options** tab of the CPU component.

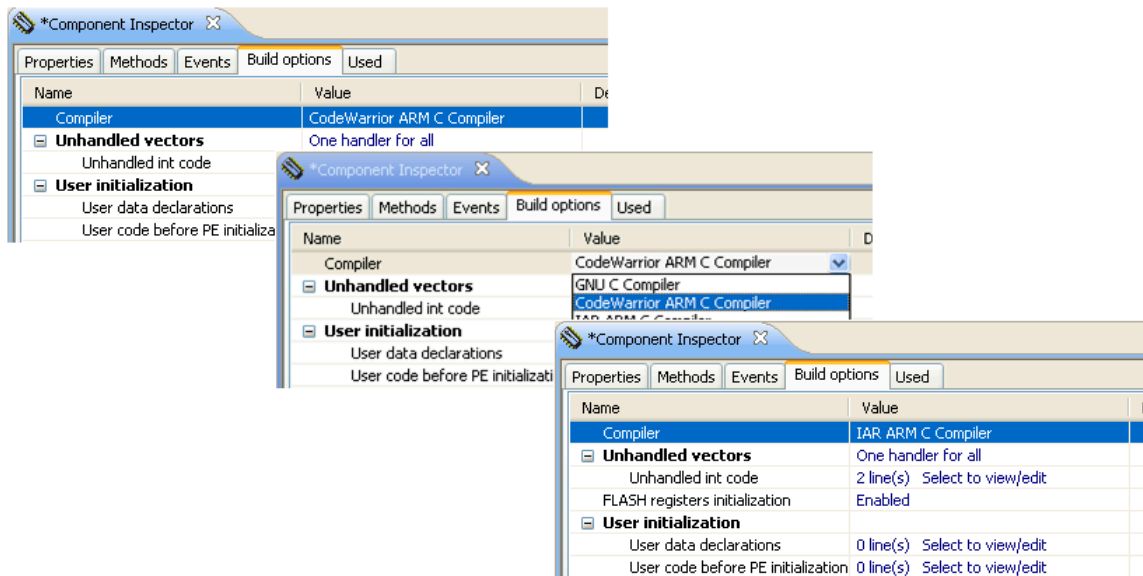


Figure 19. Changing Target Compiler

5 Importing Example Projects

The Microcontrollers Driver Suite comes with a set of example projects. The example projects are stored in the **ProcessorExpert\Projects** folder. To import an example project:

1. Select **File > Import** to open the Import wizard shown below.
2. Select **General > Existing Projects into Workspace**.

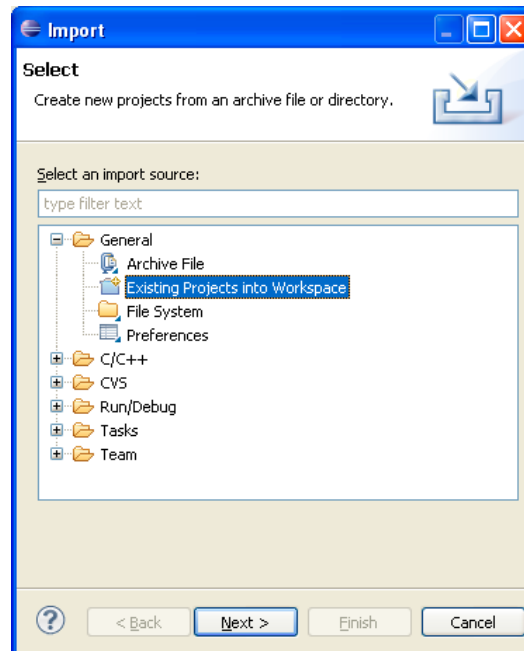


Figure 20. Import Screen

3. Click **Next** to display the **Import Projects** screen.

- Click **Browse** to select the **ProcessorExpert\Projects** folder. The **Projects** list box lists all the projects available for import.

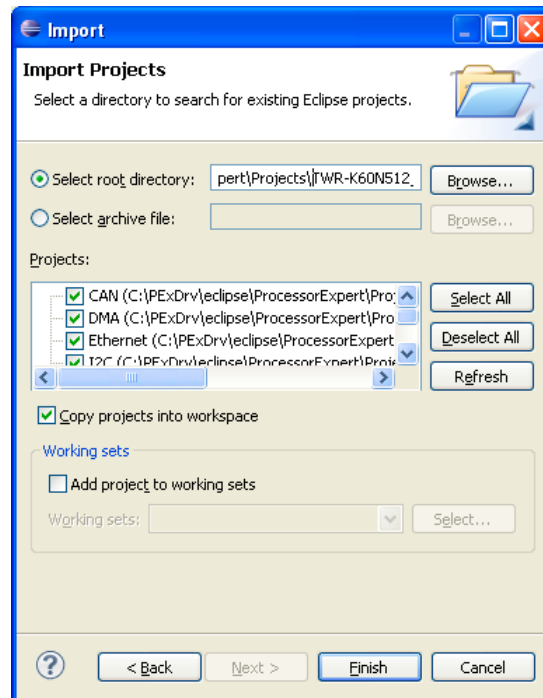


Figure 21. Import Projects Screen

- Check all projects that you want to import into your workspace. To copy the example project into your workspace, check the **Copy projects into workspace** checkbox.
- Click **Finish** to import selected projects.

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
+86 010 5879 8000
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution
Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor 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 Freescale Semiconductor 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. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale, the Freescale logo, CodeWarrior, Processor Expert are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. ARM is the registered trademark of ARM Limited. ARMnnn is the trademark of ARM Limited.

© Freescale Semiconductor, Inc. 2012.