Freescale Semiconductor
Application Note

Document Number: AN4771

Integrating FreeMASTER-Time Debugging Tool with CodeWarrior for Microcontrollers v10.x Project

1. Introduction

This application note describes the steps to integrate the project created in the CodeWarrior for Microcontrollers v10.x product with FreeMASTER-Time Debugging Tool v1.3 (formerly known as PC Master).

The FreeMASTER tool is a user-friendly real-time debug monitor and data visualization tool that you can use for any application development and information management. The FreeMASTER tool allows control of an embedded application from a graphical environment running on a personal computer.

Contents

2. Integrate Projects Manually

To integrate the project created in the CodeWarrior for Microcontrollers v10.x product with application created in the FreeMASTER v1.3 tool, perform the following steps.

1. Create a New Workspace in Eclipse IDE product.
2. Create a new application in the FreeMASTER

v1.3 tool.

3. Establish connection between the CodeWarrior project and the FreeMASTER application.

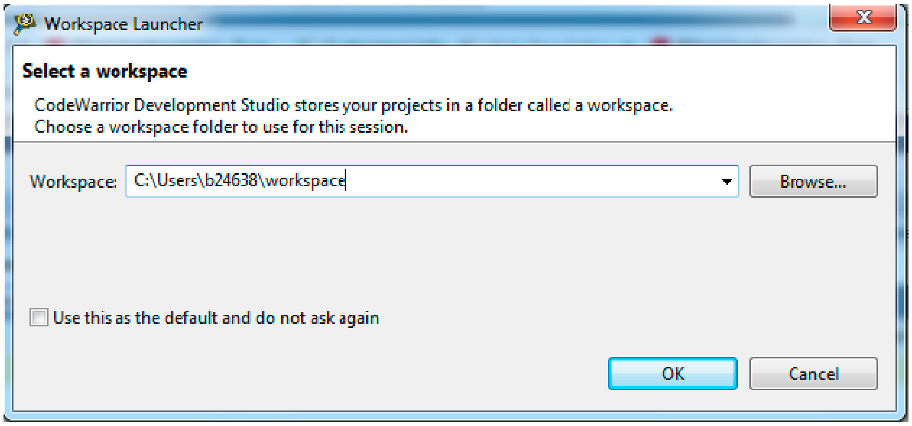
3. Create a New Workspace in Eclipse IDE

1. Start CodeWarrior for Microcontrollers v10.x tool.

   a) Select **Start** > **Programs** > **Freescale CodeWarrior** > **CW for MCU v10.4** > **CodeWarrior IDE.**

   The **Workspace Launcher** dialog box appears.

Figure 1: Workspace Launcher Dialog Box



Workspace Launcher

Select a workspace

CodeWarrior Development Studio stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

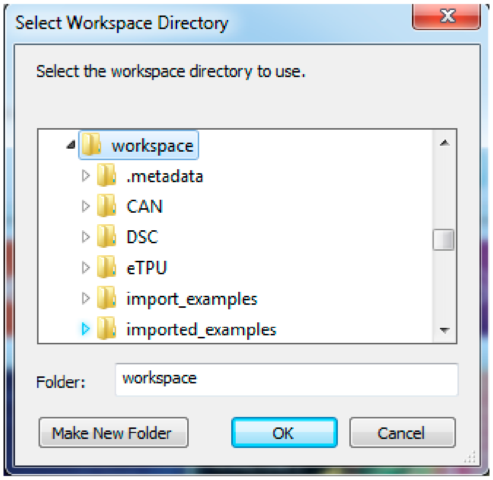
Workspace: C:\Users\b24638\workspace

□ Use this as the default and do not ask again

   b) Click **OK** button to accept the default workspace. To use a different workspace, click **Browse** button.

   The **Select Workspace Directory** dialog box appears.

Figure 2: Select Workspace Directory Dialog Box



Select Workspace Directory

Select the workspace directory to use.

- workspace
  - .metadata
  - CAN
  - DSC
  - eTPU
  - import_examples
  - imported_examples

Folder: workspace

c) Select the required folder or click the **Make New Folder** button to create a new folder for storing the new created project.

d) Click **OK** button.
The **Select Workspace Directory** dialog box closes.

3.1. Open a New Project in CodeWarrior MCU 10.x product

NOTE: Create a new project by using the new project wizard tool integrated in the CodeWarrior for Microcontrollers v10.x product.

1. Open the CodeWarrior for Microcontrollers v10.x IDE.

2. Select **File** > **New > Bareboard Project**.
   The **New Bareboard Project** dialog box appears.

3. Type a name for the project you want to create in the **Project name** text box. For example, “Test 5604P Project” and select particular location to store the new project.
4. Click **Next** button to continue the action.

5. Select the required device from the list of available devices. For example, expand **Qorivva** > **MPC56xxB Family** > **MPC5604B** and select MPC5604B device as target processor from the list of available Qorivva devices.

6. Click **Next** button to continue the action.

7. Specify the required parameters of the selected board by the **Project Wizard** tool. Click **Finish** button to finish the action.

   The **New Bareboard Project** dialog box closes.

   The wizard creates a new project according to your specifications. You can access the project from the **CodeWarrior Projects** view in the opened **Workbench** window.

NOTE: There is description of the steps how to create a new project by the project wizard tool integrated into the CodeWarrior for Microcontrollers 10.x product line at the following link: http://cache.freescale.com/files/soft_dev_tools/doc/quick_ref_guide/926-77846.pdf?fpsp=1

3.2. Create a New Application in CodeWarrior v10.x product

NOTE: There is necessary to update the main function of the new project created in the CodeWarrior for Microcontrollers v10.x product.

1. Open “main.c” source file from **Sources** folder in project tree window of the new project opened in the CodeWarrior for Microcontrollers v10.x product.

   The **Edit** window with the selected source file appears.

2. Edit and update the main function of the application, as shown in Listing 1.

Listing 1. The updated main function of the project

```
#include "MPC5604B.h"

/* Global variables definition to display the signals on the scope of FreeMASTER tool */
volatile int Light_Signal_OUT = 0;

void initModesAndClock(void) {
  ME.MER.R = 0x0000001D;          /* Enable DRUN, RUN0, SAFE, RESET modes */
                                  /* Initialize PLL before turning it on: */
/* Use 1 of the next 2 lines depending on crystal frequency: */
  CGM.FMPLL_CR.R = 0x02400100;    /* 8 MHz xtal: Set PLL0 to 64 MHz */
/*CGM.FMPLL_CR.R = 0x12400100;    /* 40 MHz xtal: Set PLL0 to 64 MHz */
  ME.RUN[0].R = 0x001F0074;       /* RUN0 cfg: 16MHzIRCON, OSC0ON,PLL0ON, syclk=PLL */
  ME.RUNPC[1].R = 0x00000010;     /* Peri. Cfg. 1 settings: only run in RUN0 mode */
  ME.PCTL[68].R = 0x01;           /* MPC56xxB/S SIUL:  select ME.RUNPC[1] */

  ME.MCTL.R = 0x40005AF0;         /* Enter RUN0 Mode & Key */
  ME.MCTL.R = 0x4000A50F;         /* Enter RUN0 Mode & Inverted Key */
  while (ME.GS.B.S_MTRANS) {}     /* Wait for mode transition to complete */
                                  /* Note: could wait here using timer and/or I_TC IRQ */
  while(ME.GS.B.S_CURRENTMODE != 4) {}  /* Verify RUN0 is the current mode */
}

void initPeriClkGen(void) {
}

void disableWatchdog(void) {
  SWT.SR.R = 0x0000c520;     /* Write keys to clear soft lock bit */
  SWT.SR.R = 0x0000d928;
  SWT.CR.R = 0x8000010A;     /* Clear watchdog enable (WEN) */
}

int main(void) {
  const int LIGHT_ON  = 64;
  const int LIGHT_OFF = 0;
  const int TimeDelay = 9500000;
  volatile int i = 0;

  initModesAndClock();     // Initialize mode entries and system clock
  initPeriClkGen();        // Initialize peripheral clock generation for peripherals
  disableWatchdog();       /* Disable watchdog */

  SIU.PCR[64].R = 0x0100;  // configure PE0 pin (SW1) as GPIO input
                           // external pull up is used on TKR board

  SIU.PCR[68].R = 0x0200;  // configure PE4 pin (LED1) as GPIO output

  /* Loop forever */
  for (;;) {

       Light_Signal_OUT = LIGHT_ON;

       for(i=0;i<TimeDelay;i++){};      // generate a time delay

    if(SIU.GPDI[64].R) {                // if PE0 pin is high toggle PE0 pin
       SIU.GPDO[68].R ^= 0x1;           // it works as following:
                                        // if SW1 not pressed LED1 is blinking
                                        // if SW1 pressed LED1 is ON or OFF
    }

       Light_Signal_OUT = LIGHT_OFF;

       for(i=0;i<TimeDelay;i++){};      // generate a time delay
  }
}
```

3. Insert global variables into the main function of the application, as shown in Listing 2.The defined global variables will be showed in the scope of the FreeMASTER application.

NOTE: There have to be the variables showed in the scope defined as global variables in user application created in the CodeWarrior for Microcontrollers v10.4 product.

Listing 2. Inserted global variables into the main function In the project

```
/* Global variables definition to display the signals on the scope of FreeMASTER tool */
volatile int Light_Signal_OUT = 0;
```

4. Right-click on the target project opened in the CodeWarrior for Microcontrollers v10.x and select **Build Project** item from the showed pop-up menu.

The changes done in the source code file are saved and project is build.

4. Install the FreeMASTER v1.3 tool

To install the FreeMASTER v1.3 tool perform the following steps.

1. Download the **FMASTERSW** tool and **FMASTERSCIDRV** Communication Driver from the following Freescale web site:

   http://www.freescale.com/freemaster/

   There are FMASTERSW.exe and FMASTERSCIDRV.exe executable files successfully downloaded from the web side.

2. Install the FreeMASTER v1.3 tool on the target PC. Double-click on FMASTERSW.exe executable file and then follow the installation instructions of **Install Shield** Wizard tool on the screen.

   The FreeMASTER v1.3 tool is successfully installed to target computer at the selected path.

3. Install the FreeMASTER Communication Driver on the target PC. Double-click on FMASTERSCIDRV.exe executable file and then follow the installation instructions of **Install Shield** Wizard tool on the screen.

   The communication driver of the FreeMASTER v1.3 tool is successfully installed to target computer at the selected path.

5. Create an Application in FreeMASTER v1.3 tool

The FreeMASTER v1.3 product allows you to create applications to control of an embedded application from a graphical environment running on a PC. The application can be distributed either as a part of Freescale CodeWarrior tool or as a standalone executable file.

NOTE: For more information see the FreeMASTER for Embedded Applications User Guide Rev. 2.2, 10/2012 manual of the product.

To establish the connection of the FreeMASTERv1.3 tool with the target board application, follow the instructions provided with the embedded-side FMASTERSCIDRV driver.
The recommended and the fastest way to start using the FreeMASTER v1.3 tool is by trying the sample application.

NOTE: The sample application name may still refer to the “PC Master” software, which is the previous name of FreeMASTER v1.3 tool.

5.1. Create a New Project in the FreeMASTER v1.3 tool

1. Open the FreeMASTER v1.3 tool.

   Select **Start- > Programs -> FreeMASTER** from Windows operating system **Start** menu.

2. Select **File -> New Project** from main menu of the tool.

3. Right-click on the **New Project** item in **Project Tree** window and select **Properties** item from the pop-up menu.

   The **Project Block Properties** dialog box appears.

4. Type a name for the project. For example, “Test Project 5604B”.

   Click **OK** button to confirm the action.

5. Right-click on **Test Project 5604B** in the **Project Tree** window and select **Create Scope** item from the showed pop-up menu.

   The **Scope Properties** dialog box appears.

6. Insert a name of the scope. For example, “Test Scope”.

   Click **OK** button to confirm the action.

7. Open the inserted scope in the project.

8. Double click on **Test Scope** in showed scope **Project Tree** window.

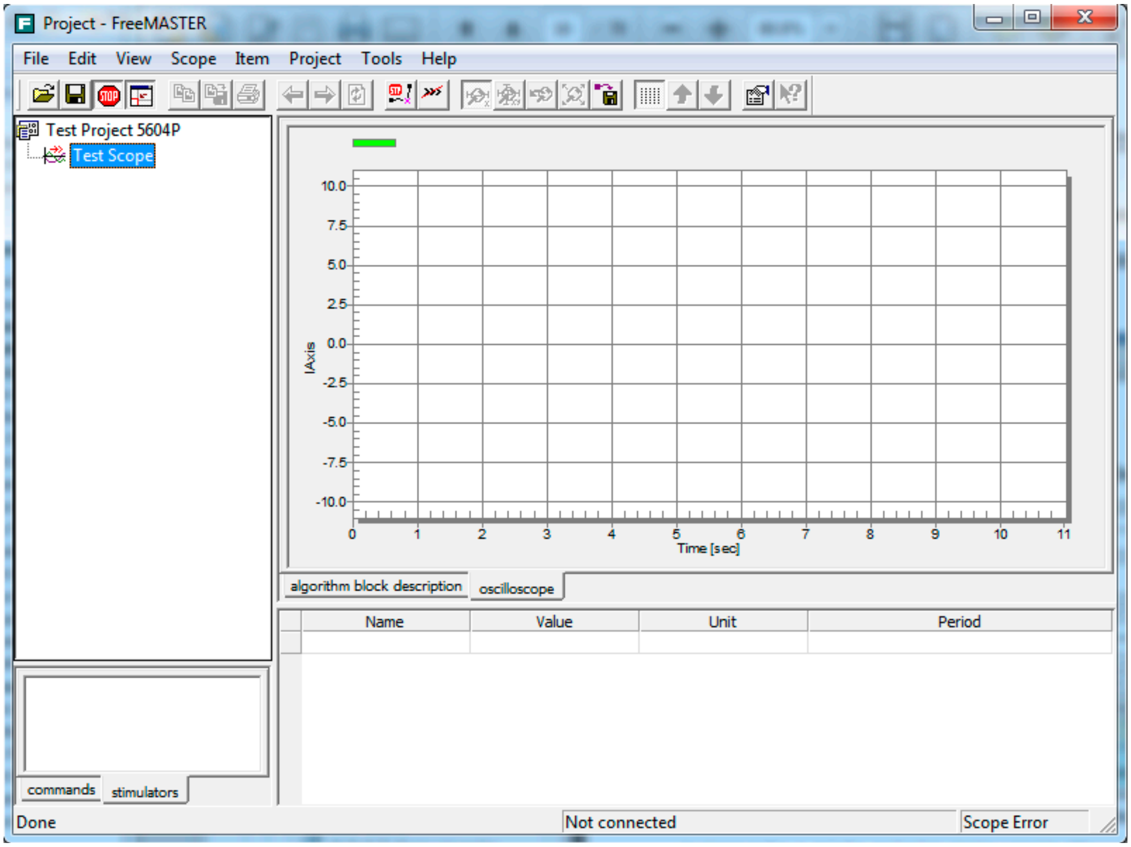
   The selected scope is opened, as shown on Figure 3.

9. Set the parameters of the scope. Right click on the scope and select **Properties** from the showed pop-up menu.

   The **Scope Properties** window appears.

10. Set or change the required parameters of the scope on **Main** page and click **OK** button to confirm the action.

Figure 3: Scope of project created in FreeMASTER v1.3 tool



6. Integrate the FreeMASTER Project

NOTE: There have to be correctly done and valid project in CodeWarrior for Microcontrollers v10.x product and project created in FreeMASTER v1.3 tool.

6.1. Establish the Communication and Connection Parameters Settings

1. Select **Project -> Options** from the main menu of FreeMASTER v1.3 tool.
   The **Options** window appears, as shown in Figure 4.

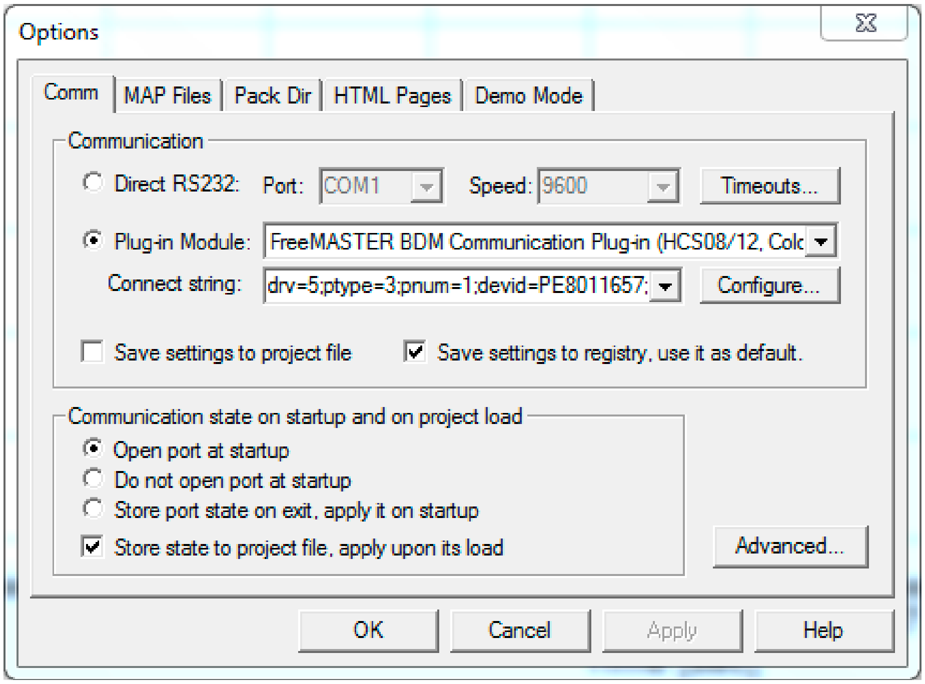
2. Click **Plug-in** module on **Comm** page and select **FreeMASTER BDM communication plug-in** from the list of available Plug-ins for the communication.

3. Click **Configure** button to set configuration parameters of the communication.

The **HCS08/HCS12/CFVx and Kinetis BDM communication** window appears, as shown in Figure 5.

4. Select required Driver type used for the communication and set parameters of the connection. Click **OK** button to confirm the action.

Figure 4: Communication parameters setting window of FreeMASTER v1.3 tool



Options

Communication

○ Direct RS232: Port: COM1 Speed: 9600

● Plug-in Module: FreeMASTER BDM Communication Plug-in (HCS08/12, Colc

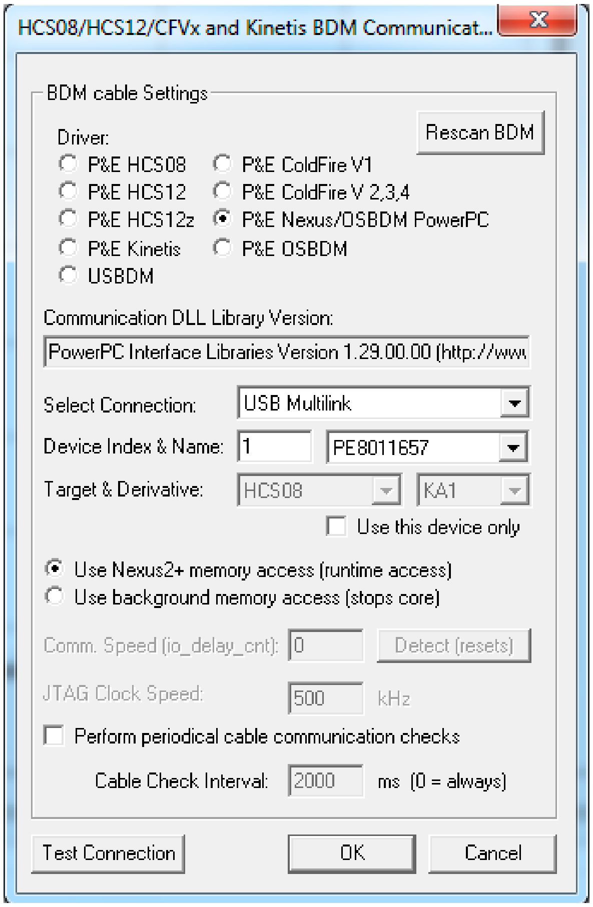
Connect string: drv=5;ptype=3;pnum=1;devid=PE8011657;

□ Save settings to project file ☑ Save settings to registry, use it as default.

Communication state on startup and on project load

● Open port at startup
○ Do not open port at startup
○ Store port state on exit, apply it on startup
☑ Store state to project file, apply upon its load

Figure 5: Connection Driver configuration and parameters settings window



HCS08/HCS12/CFVx and Kinetis BDM Communicat...

BDM cable Settings

Driver:

○ P&E HCS08 ○ P&E ColdFire V1
○ P&E HCS12 ○ P&E ColdFire V 2,3,4
○ P&E HCS12z ● P&E Nexus/OSBDM PowerPC
○ P&E Kinetis ○ P&E OSBDM
○ USBDM

Communication DLL Library Version: PowerPC Interface Libraries Version 1.29.00.00 (http://www

Select Connection: USB Multilink

Device Index & Name: 1 PE8011657

Target & Derivative: HCS08 KA1

□ Use this device only

● Use Nexus2+ memory access (runtime access)
○ Use background memory access (stops core)

Comm. Speed (io_delay_cnt): 0

JTAG Clock Speed: 500 kHz

□ Perform periodical cable communication checks

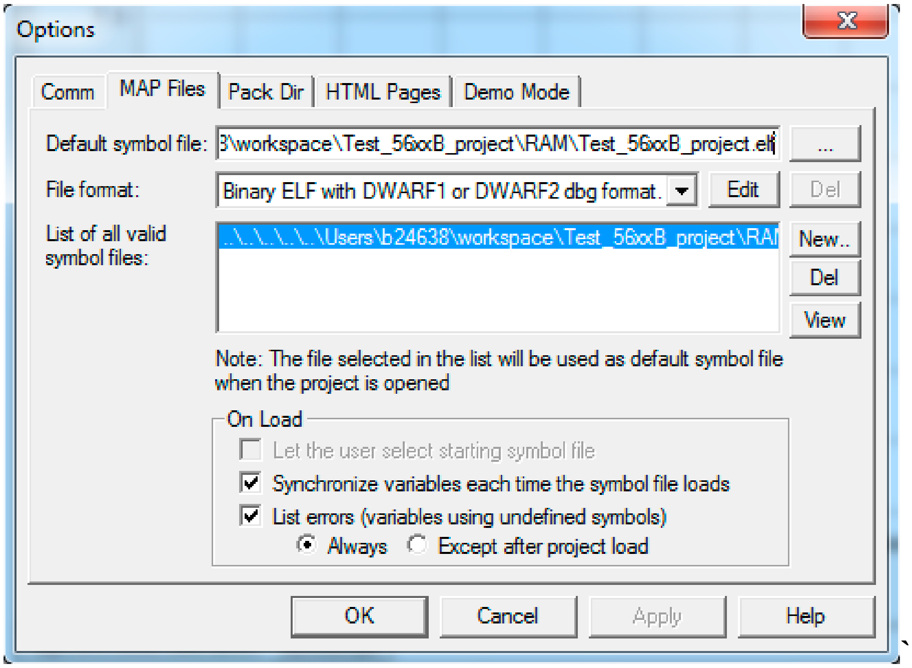
Cable Check Interval: 2000 ms (0 = always)

6.2. Establish .ELF file for the communication and variables definition

1. Click **Default symbol file** item on the **MAP Files** page and browse the Debug Executable file of the CodeWarrior for Microcontrollers v10.x project, as shown on Figure 7.

2. Set the file format property of the selected .ELF type file from the list of available file formats. Select **Binary ELF with DWARF1 or DWARF2 dbg format** item as file format from the list as current file types. Click **OK** button to confirm the action.

Figure 6: Select .ELF file for the communication and variables definition



Options

Comm | MAP Files | Pack Dir | HTML Pages | Demo Mode

Default symbol file: 3\workspace\Test_56xxB_project\RAM\Test_56xxB_project.elf

File format: Binary ELF with DWARF1 or DWARF2 dbg format.

List of all valid symbol files: ..\..\..\..\..\Users\b24638\workspace\Test_56xxB_project\RAM

Note: The file selected in the list will be used as default symbol file when the project is opened

On Load

□ Let the user select starting symbol file

☑ Synchronize variables each time the symbol file loads

☑ List errors (variables using undefined symbols)

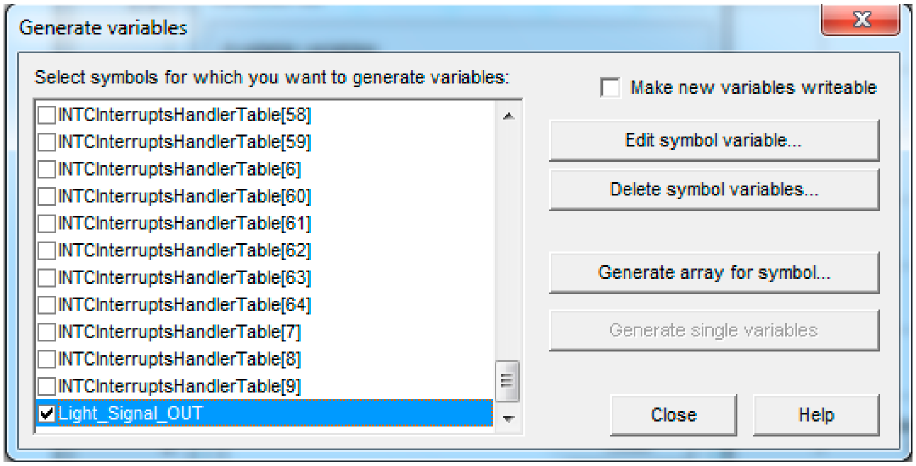
● Always ○ Except after project load

6.3. Displayed Variables in the Scope Definition

1. Select **Project** > **Variables** from the main menu of FreeMASTER v1.3 tool. The **Generate Variables** window appears, as shown in Figure 6.

2. Click **Generate…** button on the **Generate Variables** window to generate a list of variables to be displayed in the Scope.

Figure 7: Generate a list of variables for the Scope window



Generate variables

Select symbols for which you want to generate variables:

□ INTCInterruptsHandlerTable[58]
□ INTCInterruptsHandlerTable[59]
□ INTCInterruptsHandlerTable[6]
□ INTCInterruptsHandlerTable[60]
□ INTCInterruptsHandlerTable[61]
□ INTCInterruptsHandlerTable[62]
□ INTCInterruptsHandlerTable[63]
□ INTCInterruptsHandlerTable[64]
□ INTCInterruptsHandlerTable[7]
□ INTCInterruptsHandlerTable[8]
□ INTCInterruptsHandlerTable[9]
☑ Light_Signal_OUT

□ Make new variables writeable

Edit symbol variable...
Delete symbol variables...
Generate array for symbol...
Generate single variables
Close
Help

3. Select symbols for which will generate variables. Check **Light_signal_OUT** symbol and click **Generate single variable** button. Click **Close** button to finish the action.

6.4. Scope Properties and Parameters definition

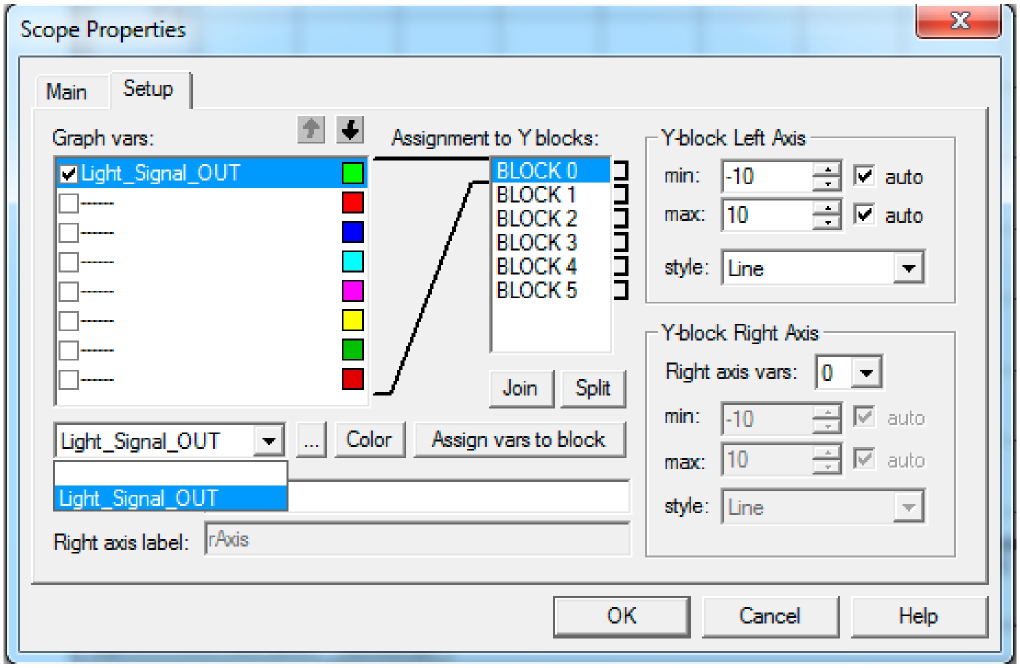
1. Right click on the Scope and select **Scope properties** item from the showed pop-up menu. The **Scope Properties** window appears, as shown in Figure 6.

NOTE: There is possible to change/set definition and parameters of the created variables. Click **Edit Symbol Variable** button to open the Variable window for selected variable.

2. Select **Setup** page of the opened window and add created symbols into the list of Graph variables.
3. Click on position in **Graph Vars** page and select **Light_signal_OUT** symbol from the list of available symbols, as shown in Figure 7. Click **OK** button to finish the action.

   The **Light_signal_OUT** symbol is added into the list of Graph variables to be displayed in the Scope of the FreeMASTER v1.3 tool.

Figure 8 - Scope parameters and properties setting window



Scope Properties

Graph vars:
☑ Light_Signal_OUT 🟩
□ ------- 🟥
□ ------- 🟦
□ -------
□ ------- 🟪
□ ------- 🟨
□ ------- 🟩
□ ------- 🟥

Assignment to Y blocks: BLOCK 0, BLOCK 1, BLOCK 2, BLOCK 3, BLOCK 4, BLOCK 5

Light_Signal_OUT

Right axis label: rAxis

Y-block Left Axis
min: -10 ☑ auto
max: 10 ☑ auto
style: Line

Y-block Right Axis
Right axis vars: 0
min: -10 ☑ auto
max: 10 ☑ auto
style: Line

TIP: There is possible to change/set Scope properties and parameters on the **Main** page of the showed window.

4. Select **File** > **Save** from the main menu of the FreeMASTER v1.3 tool to save the changes done in the project.

7. Target Device Setting and Projects Run

The demonstration application works like following:

- If SW1 input pin is not pressed then LED1 is blinking on MPC5604B target board.
- If SW1 input pin is pressed then LED1 is in ON or OFF status on MPC5604B target board.

There has to be done the following steps to run the demonstration application:

1. Connect the required voltage levels to the pins for the digital inputs of the microprocessor target device.

NOTE: For detailed description of the digital signals connection see comments of “main.c” source code file in the “Test 5604B Project” application.

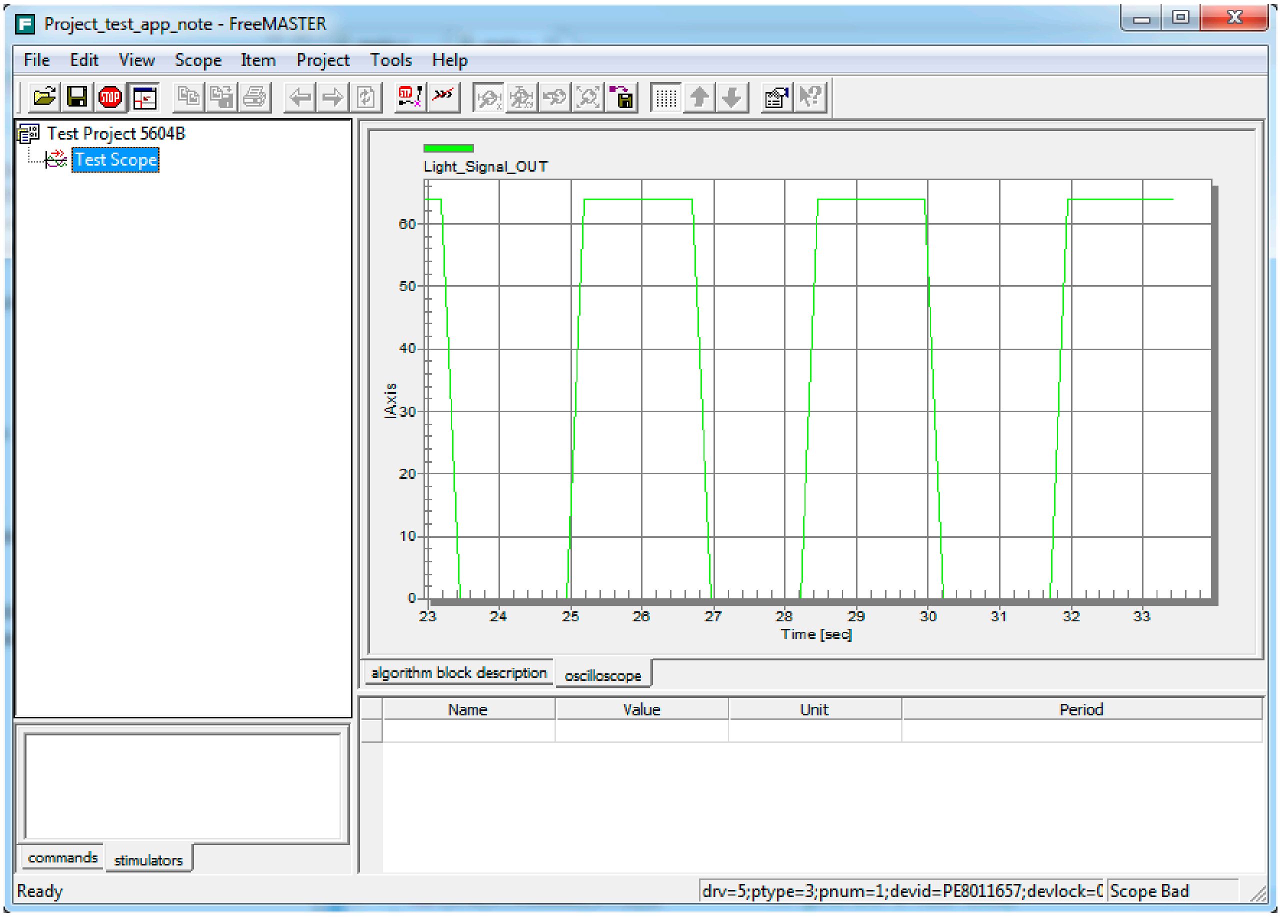
2. Open the test project created in CodeWarrior for Microcontrollers v10.x tool to the MPC5604B target device, debug and run the project.

   The “Test 5604B Project” application is running on the target device.

3. Select **File** > **Start** communication from main menu of the FreeMASTER v1.3 tool.

   The “Test Project 5604B” application is running.

Figure 9 - Running project in FreeMASTER v1.3 tool



Project_test_app_note - FreeMASTER

Name	Value	Unit	Period

Ready

drv=5;ptype=3;pnum=1;devid=PE8011657;devlock=(Scope Bad

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

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)
support@freescale.com

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 Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.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.