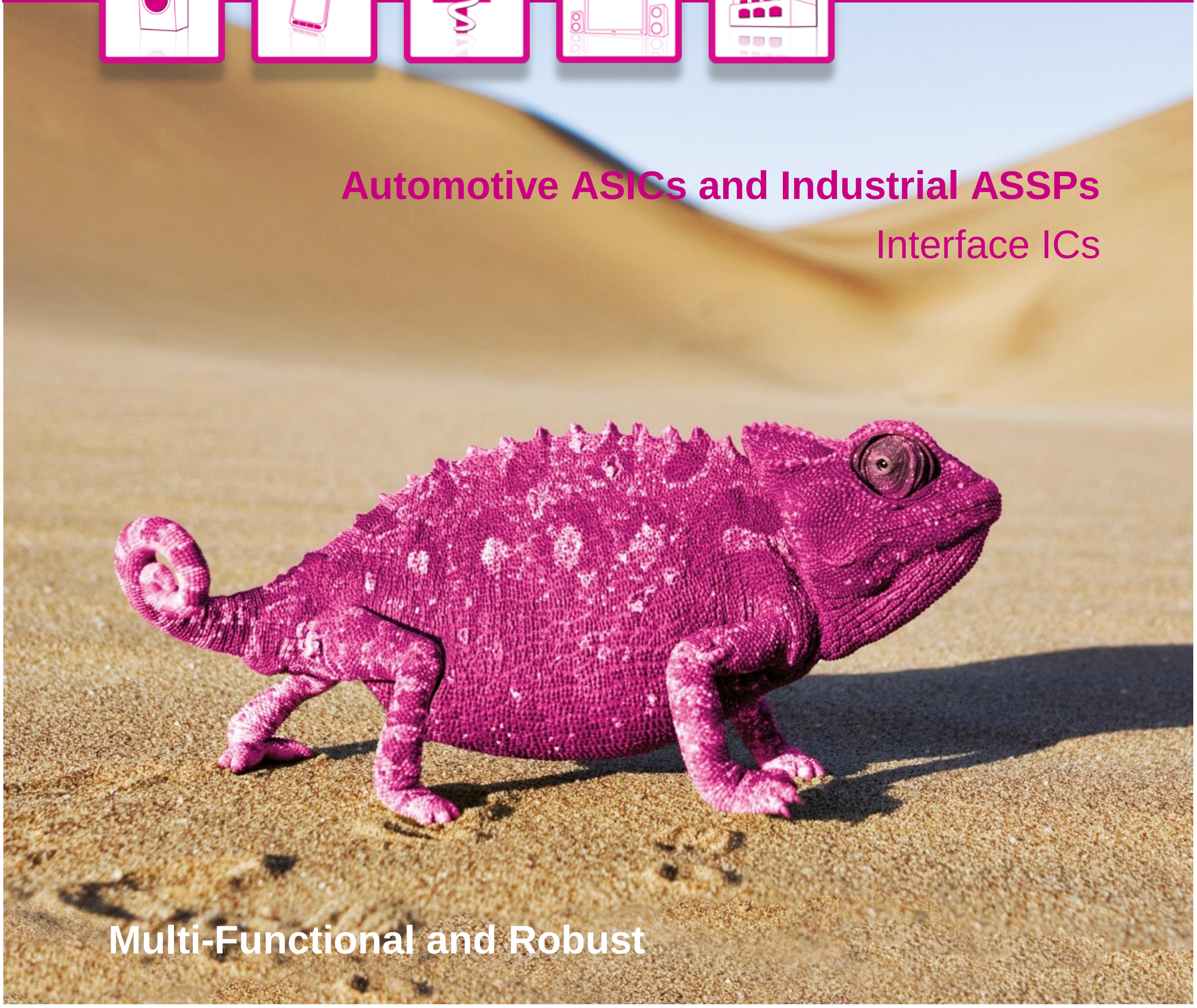
The Analog Mixed Signal Company

Technical Note

Rev. 1.00 / October 2014

# ZWIR45xx

## Setting Up an Open Source Development Environment



Automotive ASICs and Industrial ASSPs

Interface ICs

Multi-Functional and Robust

## Content

*For more information, contact ZMDI via wpan@zmdi.com.*

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

# 1 Introduction

Application development for embedded systems is typically a rather complex task, considering the number of tools involved in this process (see section 1.1). Typically these tools are hidden to some extent by an integrated development environment (IDE), which makes use of the individual tools to accomplish the tasks that the developer is requesting. Although other IDEs can be used, ZMDI provides integration and technical support for Rowley Associates' CrossWorks™, which is a bundle of tools integrated with the CrossStudio™ IDE. CrossWorks™ is a commercial bundle, partially based on open-source tools, such as the Gnu Compiler Collection (GCC).

Purchasing a commercial full-featured IDE, such as Rowley Associates' CrossWorks™, provides the advantage of getting easy-to-use project set up, code completion, easy access to project settings, and, most important, easy debugging and seamless integration of all these features in a single IDE. Under some circumstances, however, a commercial IDE with its associated license cost is not appropriate for a project. This document describes how a comparable functionality can be provided without the cost of purchasing a license for such projects. Note that the setup described might be less stable than a commercial tool chain and that ZMDI does not provide commercial support for this setup. The steps described in this document provide a quick entry point into a cost-free working environment for those who are able to maintain this setup. For this document, it is assumed that debugging is done using the USB JTAG connector as it is used on the ZWIR4512 Development Boards in the ZWIR4512 Development Kit. Different JTAG adaptors can be used, but the user must determine the appropriate OpenOCD configuration.

It is strongly recommended that the user implement only the versions of the tools referenced in these procedures when setting up a development environment. It is important to use only the specific path names given in these steps as changing path settings affects multiple steps needed for the environment setup. If alternative paths are used, avoid white spaces in path names, as these can lead to undefined behavior if not used appropriately.

## 1.1. Required Components

### 1.1.1. Compiler and Binary Tool Chain

The main components of the compiler and binary tool chain are the preprocessor, compiler, assembler, and linker. There are also tools available for the analysis and further processing of the binary output. The proposed open-source tool chain uses the GCC-based ***bleeding-edge-toolchain*** for bare-metal ARM microcontrollers. The version used in this document is the version published July 8, 2014, based on ***GCC Version 4.8.4***.

### 1.1.2. Standard Library

The standard library contains all the standard functionality that is provided by common C headers. The functions **`printf`** or **`malloc`** are commonly used functions, and their functionality is implemented in the standard library. The standard library is one of the differentiators between commercial and non-commercial tool chains. Commercial tool chains typically provide size-optimized implementations of the standard library, resulting in binary sizes up to 20kB smaller than tool chains linked with non-optimized libraries. The proposed tool-chain setup uses the standard library provided with the *bleeding-edge-toolchain*. This library is based on ***Newlib version 2.1***.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

### 1.1.3. Debugger

The debugger provided is used to run programs with the option to set breakpoints for stopping the program at a specific point of interest. When the program is stopped, current values of variables, memory, and register contents can be examined. For the proposed tool-chain setup, the debugger is the GNU debugger ***gdb version 7.8.50***, which is included with the *bleeding-edge-toolchain.*

### 1.1.4. On-Chip Debugger

An on-chip debugger is a program that allows debugging to take place directly on the chip running in its target environment. For that purpose, the on-chip debugger accesses the chip via the JTAG protocol, providing a command interface for the debugger on the other side. In addition to on-chip debugging, the on-chip debugger allows programming the devices. The setup described in this document uses ***OpenOCD Version 0.7.0*** as the on-chip debugger.

### 1.1.5. Integrated Development Environment

The integrated development environment (IDE) is a graphical user interface (GUI) that provides some form of project manager, source code editor, and debugger frontend. This is optional, as all the other tools can be used separately as well. However, the IDE is wrapped around all the tools used in application development, providing a unified, easy to use interface to their functionality. The setup proposed in this document is based on the ***Eclipse Version 4.4*** (Eclipse Luna), which has a very powerful source code editor and is widely configurable.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

# 2 Environment Setup

## 2.1. ZWIR45xx Libraries and Tools

Download ***ZWIR45xx.msi*** from the ZWIR4512 product page www.zmdi.com/zwir4512 on the ZMDI website www.zmdi.com or use the version of this file that is distributed on the ZWIR45xx Development Kit DVD. Install the package and choose the complete setup.

## 2.2. Compiler and Binary Tool Chain, Debugger, and Standard Library

Navigate to the download page bleeding-edge-toolchain for bare-metal ARM microcontrollers and download the appropriate version of *bleeding-edge-toolchain* for the user's computer (32 or 64 bits). Unzip the tool chain to a user-selected directory.

Alternatively, the ***Sourcery CodeBench Lite*** tool chain can be used. To download this tool chain, navigate to the Sourcery CodeBench Lite website and choose "Download EABI Release" from the section "ARM Processors." This will open a registration request. Registration is required but free of charge. A download link for the compiler package will be sent to the e-mail address that is used for registration. Download and install the package on the user's computer.

## 2.3. Installation of *OpenOCD* for Windows®

Due to license restrictions, the *OpenOCD* on-chip debugger must be downloaded and compiled from source code. Follow these installation steps if the operating system for the user's computer is Windows®:

1. Create directory `C:\temp\openocd-build`.
2. Download the appropriate FTDI driver package for the user's computer.
3. Unzip the contents of the downloaded zip file to `C:\temp\openocd-build\ftdi`.
4. Download OpenOCD 0.7.0 source code.
5. Unzip the contents of the zip file to `C:\temp\openocd-build`.
6. Download the MinGW installer.
7. Execute *MinGW installer:*
   - Choose *Install.*
   - Keep proposed settings and click *Continue*.
   - Choose *Continue*.
   - Mark the following packages for installation:
     - *mingw-developer-tools*
     - *mingw32-base*
     - *msys-base* (should be selected automatically with *mingw-developer-tools*)
   - Select menu *Installation* → *Apply Changes*.

© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

8. Open the Windows® command prompt.
9. Enter the following sequence of commands. (Note: the command starting with **find**… typically generates some error messages, which can be ignored.)

```
C:\Windows\System32> \MinGW\msys\1.0\bin\bash
bash-3.15$ export PATH=/c/MinGW/bin:/c/MinGW/msys/1.0/bin
bash-3.15$ cd /c/temp/openocd-build/openocd-0.7.0
bash-3.15$ find -name "*.c" -exec sed -I -e "s/\<isascii\>/__isascii/g" {} \;
bash-3.15$ mkdir build
bash-3.15$ cd build
bash-3.15$ ../configure  --enable-ft2232_ftd2xx
--with-ftd2xx-win32-zipdir=/c/temp/openocd-build/ftdi/ --disable-werror.
bash-3.15$ make
```

## 2.4. Eclipse IDE

Follow these steps to install the *Eclipse* integrated development environment.

1. Navigate to http://www.eclipse.org/downloads/.
2. Download *Eclipse IDE for C/C++ Developers*.
3. Unzip the file to a user-selected directory (e.g., `C:\Program Files\`).
4. Execute *eclipse.exe.*
5. On the top menu, select *Help* → *Install New Software*….
6. Select *Luna – http://download.eclipse.org/releases/luna* in the *Work with* dropdown.
7. After *Eclipse* has read the component repository, mark the item *C/C++ GDB Hardware Debugging* from the *Mobile and Device Development* group.
8. Click *Next* and follow the instructions to complete the installation of the new component.
9. Reopen the *Help* → *Install New Software…* dialog.
10. Click on the *Add* button.

| | |
|---|---|
| • Under *Name* enter | *GNU ARM Eclipse* |
| • Under *Location* enter | *http://gnuarmeclipse.sourceforge.net/updates* |
| • Click *OK*. | |

© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

11. After *Eclipse* has read the target repository, mark the following items from group *GNU ARM C/C++ Cross Development Tools*:
    - *GNU ARM C/C++ Cross Compiler Support*
    - *GNU ARM OpenOCD Debugging Support*
    - *GNU ARM Packs Support*
12. Click *Next* and follow the instructions to complete the installation of the new components.
13. Reopen the *Help → Install New Software…* dialog.
14. Click the *Add* button.
    - Under *Name* enter *Embedded System Register Viewer*
    - Under *Location* enter *http://embsysregview.sourceforge.net/update*
    - Click *OK.*
15. After *Eclipse* has read the target repository, mark all items for installation.
16. Click *Next* and follow the instructions to complete the installation of the new components.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

## 3 Project Setup

This section gives step-by-step instructions on how to set up a new project. The newly generated project will be set up in such way that it can be used as template for future projects, reducing the project setup effort. Exercising the whole project setup from the beginning can help the user understand *Eclipse* configurability and shows many important locations in the configuration settings dialogs.

In order to prepare the project setup, execute the following steps:

1. Open *Eclipse*.
2. Select a location for the workspace. A workspace is a file system directory containing one or multiple projects in separate subdirectories.
3. Close the welcome screen if shown.

### 3.1. Creating a Template Project

Follow these steps to create a template, which can also be used for future projects.

1. From the main menu, select *File → New → C Project.*
2. Specify the *Project name* as "*ZWIR4512 Template*".
3. Under *Project type*, select *Executable → Empty Project.*
4. Under *Toolchains*, choose *Cross ARM GCC*.
5. Click *Next.*
6. Keep the proposed settings on this page and click *Next* again.
7. Select *GNU Tools for ARM Embedded Processors (arm-none-eabi-gcc)* under *Toolchain name*.
8. Under *Toolchain path*, enter the installation directory of the compiler tool chain (from section 2.2).
9. Click *Finish*.
10. *Eclipse* should now show the C/C++ development perspective. Use these steps to import the required source and system files:
    - Right click on the project in the project explorer and select *Import…*
    - Select *Archive File* from group *General* and click *Next*.
    - Click the *Browse* button and navigate to the file `ZWIR4512EclipseTemplate.zip`, which is located in `<ZWIR45xx installation directory>/src`
    - Click *Finish* to import files.
11. Right click on the project in the project explorer and select *Properties*.
12. In the list of property groups on the left side of the dialog, select *C/C++ Build*.
    - Set the item *Builder type* in the *Builder Settings* tab to *External Builder*.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

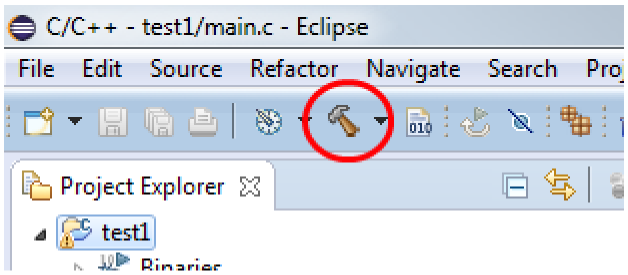
13. Open the *C/C++ Build* group on the left and select *Settings*. Apply the following changes in this section:
    - Select *[ All configurations ]* from the *Configuration* dropdown.
    - Select *Optimization* in the list of *Tool Settings*
        - Checkmark *No common uninitialized*
    - Select *Includes* in group *Cross ARM C Compiler*
        - Add new item under *Include paths* containing this text: `"${ZWIRRoot}/include"` (be sure to include the quotation marks).
    - Select *General* in group *Cross ARM C Linker*
        - Add a new script: `"${workspace_loc:/${ProjName}/system/ZWIR4512.ld}"`
        - Enable the option "Do not use standard start files."
    - Select *Libraries* in Group *Cross ARM C Linker*
        - Add the following entries to the *Library* list and ensure they are ordered as shown below:

          *ZWIR45xx-6LoWPAN*

          *ZWIR451x-UART1*

          *ZWIR451x-Devkit*

          *ZWIR4512*
        - Add this entry to the *Library search path* list: `"${ZWIRRoot}/lib"` (be sure to include the quotation marks).
14. Close the Settings dialog by clicking *OK*.
15. Try building the current configuration by pressing the *Build* button, which is an image of a hammer:



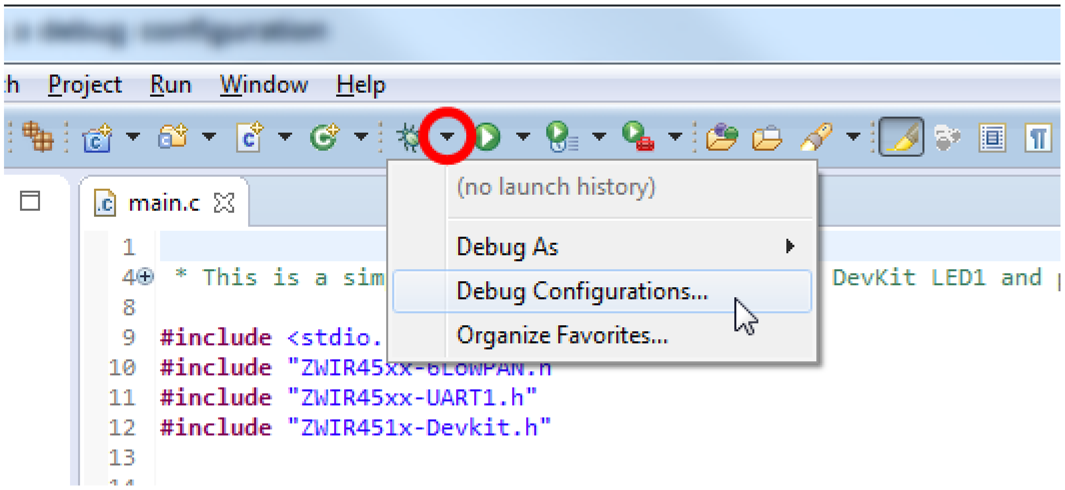
16. If the build fails, try restarting *Eclipse* after a system reboot to ensure that all new path variable settings are applied properly.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

## 3.2. Adding a debug configuration

Follow these steps to add *GDB OpenOCD Debugging:*

1. Click on the arrow adjacent to the *Debug* button and choose *Debug configurations…*:



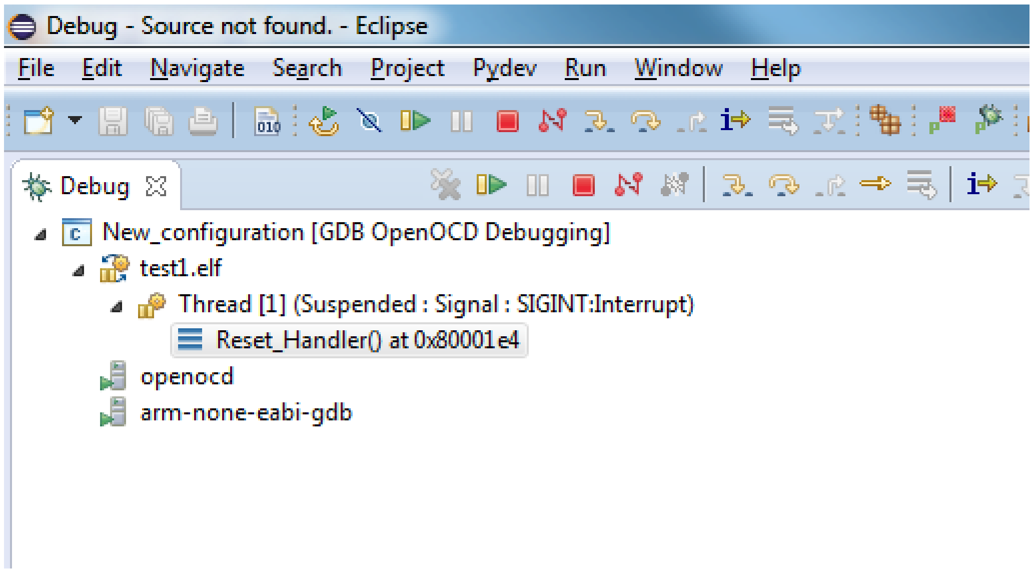
```
 * This is a sim                              DevKit LED1 and

#include <stdio.
#include "ZWIR45xx-6LOWPAN.h"
#include "ZWIR45xx-UART1.h"
#include "ZWIR451x-Devkit.h"
```

2. Double click on *GDB OpenOCD Debugging* to generate a new debug configuration.
3. Verify that the project *.elf* file is shown in the *C/C++ Application* entry field and the *Project* entry field contains the project name on the *Main* tab of the debug configuration.
4. Change to the *Debugger* tab and complete these steps:
   - Click on *Variables* in the *OpenOCD Setup* group
   - Click *Edit variables…* in the resulting dialog window.
   - Double click *openocd_path* and select the location of `openocd.exe` in the *Value* field (for Windows®, this should be `C:\temp\openocd-build\openocd-0.7.0\build\src` when adhering to the instructions in this document).
   - Close the dialogs with *OK* (three times).
   - Enter a name for the *Log file*: "`log/OpenOCD.log`".
     Important: the path separators must be slashes (/), not backslashes (\), regardless of the operating system.
   - Add "`--file system/ZWIR4512-OpenOCD.cfg`" to the *Config Options* field (don't include the quotation marks).
     Important: the path separators must be slashes (/), not backslashes (\), regardless of the operating system.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

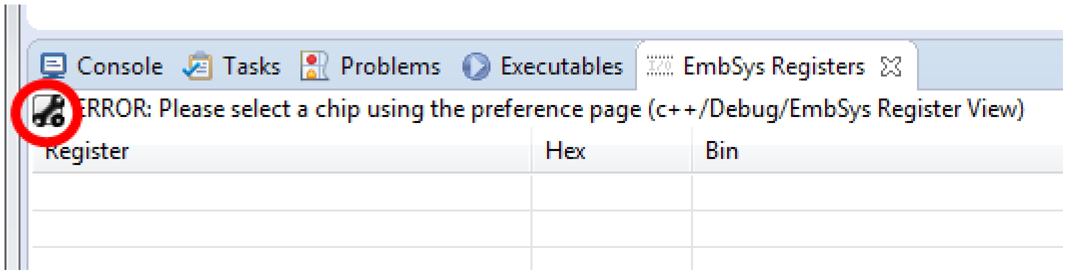
5. Change to the *Startup* tab
   - Deselect the following checkboxes:
     - *Initial Reset*
     - *Enable ARM Semihosting*
     - *Set breakpoint at*
   - Click *Debug*.
   - When a dialog appears, asking to switch to the Debug Perspective, confirm.
   - The debugger should now connect to the device, stopping at the function *Reset_Handler* (see below). If not, refer to section 3.3 below for troubleshooting methods.


Debug - Source not found. - Eclipse
File Edit Navigate Search Project Pydev Run Window Help
Debug

New_configuration [GDB OpenOCD Debugging]
test1.elf
Thread [1] (Suspended : Signal : SIGINT:Interrupt)
Reset_Handler() at 0x80001e4
openocd
arm-none-eabi-gdb

   - The program can be run and stopped using the and buttons.
   - Option: It is possible to add a view showing the processor's core registers:
     - In the menu bar, select *Windows* → *Show View* → *Registers.*
     - Now a new window should be displayed showing the STM32 core registers.
   - Option: It is possible to add a view showing the processor's peripheral registers:
     - In the menu bar select *Windows* → *Show View* → *Other.*
     - In the resulting dialog, select *EmbSys Registers* from the *Debug* group.
     - Click *OK*. The result is that a new view is displayed, which needs to be configured in order to show the STM32 processor registers.

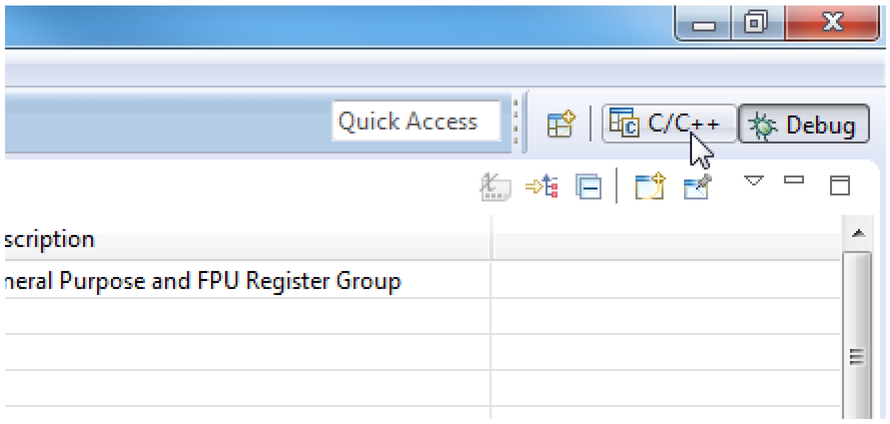
**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.



ERROR: Please select a chip using the preference page (c++/Debug/EmbSys Register View)

- Click on the wrench icon that is circled in above figure.
- Choose the following values in the resulting dialog:
  - Architecture: *cortex-m3*
  - *Vendor:* *STMicro*
  - Chip: *STM32F10X_HD*
- Click *OK*. The result is that all accessible STM32 registers are displayed, grouped by components.
- To view the value of a particular register, open the corresponding group and double click the register that should be read.

Note: Launching the debug configuration causes *Eclipse* to switch to the Debug Perspective. To switch back to the C/C++ Development Perspective, the corresponding button in the top-right corner of the *Eclipse* window must be clicked (see image below). This will bring return to the Project explorer and source code editor.

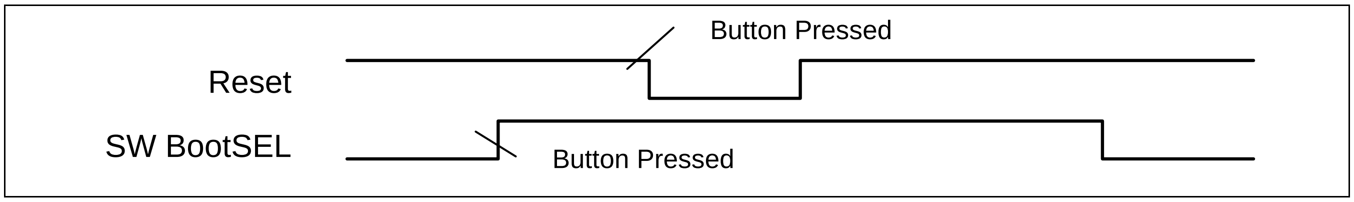


© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

## 3.3. Debug Troubleshooting

The ZWIR45xx network stack is highly optimized toward power consumption. This includes sending the module to Sleep Mode when no processing is to be performed. The network stack includes detection of whether the program being executed is running under debugger control. If so, the low-power debug functionality, which can be enabled explicitly using the API function **`ZWIR_EnableLowPowerDebug ( )`**, is enabled implicitly. However, after power-on, there is no debugger attached and consequently low-power debug support is disabled. This can lead to debugger connection problems, causing *Eclipse* to show an error message when trying to launch a debug configuration. In this case, it might be necessary to launch the debug configuration multiple times. However, the most effective way to connect the debugger is by forcing the device into its internal boot-loader mode before launching the debug configuration. This is done by a pulse on its Reset pin while the BSEL pin is active. On the ZWIR4512 Development Board, this is easily accomplished using the “Reset” and “SW BootSEL” buttons as shown in Figure 3.1.

***Figure 3.1 Forcing the ZWIR45xx Module into its Internal Boot-Loader Mode***



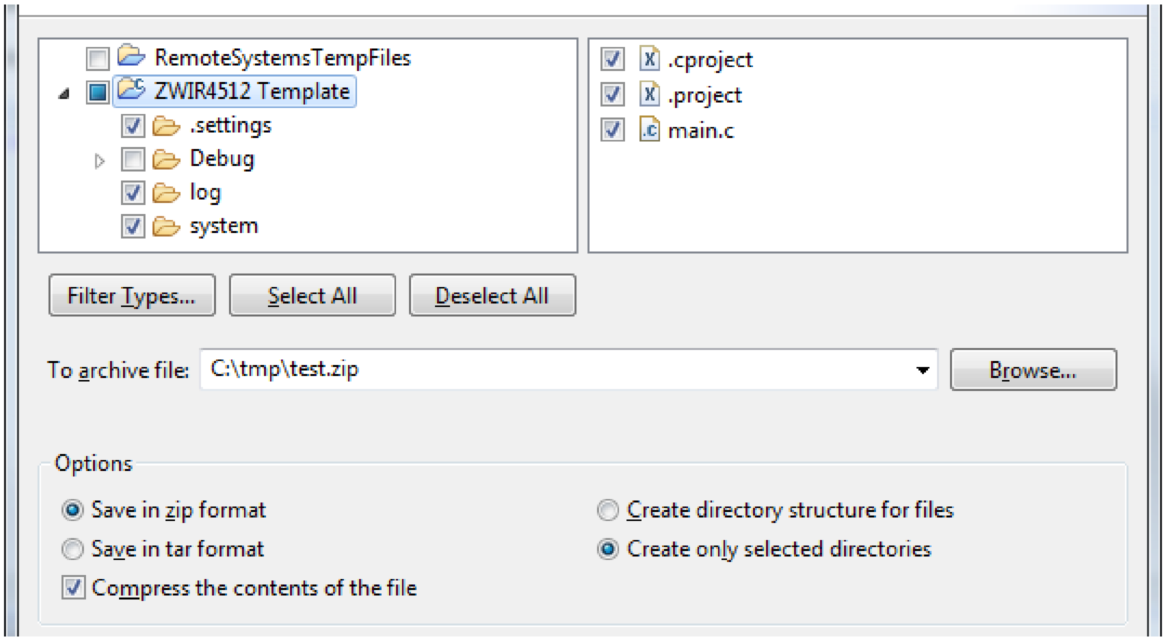
In order to avoid connection problems, the function **`ZWIR_EnableLowPowerDebug ( )`** can be called from **`ZWIR_AppInitHardware ( )`**. However, for applications that are using the low-power features of the network stack, it must be ensured that firmware delivered to customers does not use this function, as this significantly increases the power consumption of the device. Refer to the *ZWIR451X Programming Guide* for further information.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

## 3.4. Exporting the Project and Debug Configuration as Templates

To avoid the time consuming setup of all configuration parameters for new projects, the project generated in section 3.1 can be exported as a template for new projects using these steps:

1. Right click on the project in the Project Explorer and select *Export*.
2. Select *Archive File* from group *General* as the *Export destination*.
3. Select items as shown in the dialog below and choose an appropriate archive file name.



[ ] RemoteSystemsTempFiles
[■] ZWIR4512 Template
[✓] .settings
[ ] Debug
[✓] log
[✓] system

[✓] .cproject
[✓] .project
[✓] main.c

To archive file: C:\tmp\test.zip

Options
[✓ Save in zip format] Save in tar format
[✓] Compress the contents of the file
Create directory structure for files [✓ Create only selected directories]

4. Click *Finish* to export the project.
5. Debug configurations can be exported in a similar way by choosing *Run/Debug → Launch Configurations* as the *Export destination*.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

## 3.5. Creating a New Project from the Template

This section describes how a new project is generated from a previously saved project template.

Note: A workspace must not already contain a project with the same name as the saved project template. When a new project is generated in the same workspace that was used in section 3.1, the project that was saved as a template must be renamed. Otherwise it will be impossible to import. In order to rename the project, right click on the existing project in the Project Explorer, choose "Rename" and assign a new name.

1. From the main menu bar, choose *File → Import …*
2. Select *Existing Projects into Workspace* from group *General* and click *Next*.
3. Select *Select Archive File* and browse for the saved project template.
4. Select the project from the list of projects (there should only be the ZWIR4512 Template) and then click *Finish*.
5. The workspace should now contain a copy of the template project that can be renamed to the desired name.
6. Building the project should be possible without problems.

# 4 Related Documents

In addition to the following ZMDI documents, refer to the STMicroelectronics document *STM32™ and STM8™ Flash Loader Demonstrator User Manual*, which is available on the STMicroelectronics website www.st.com.

Note: X_xy refers to the current revision of the document.

| Document | File Name |
|---|---|
| *ZWIR4512 Data Sheet* | *ZWIR4512_DataSheet_Rev_X_xy.pdf* |
| *ZWIR4512 Development Kit Description** | *ZWIR4512_Dev_Kit_Rev_X_xy.pdf* |
| *ZWIR451X Programming Guide** | *ZWIR451x_Prog-Guide_Rev_X_xy.pdf* |

Visit the ZWIR4512 product page www.zmdi.com/zwir4512 on ZMDI's website www.zmdi.com or contact your nearest sales office for the latest version of these documents.

Note: A free customer login account is required for web access to documents marked with an asterisk (*). To set up a login account, click on login in the top right corner the www.zmdi.com web page and follow the instructions.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.

## 5 Glossary

| Term | Description |
|---|---|
| API | Application Programming Interface |
| GCC | GNU Compiler Collection |
| GDB | GNU Debugger |
| GUI | Graphical User Interface |
| IDE | Integrated Development Environment |

## 6 Document Revision History

| Revision | Date | Description |
|---|---|---|
| 1.00 | October 2, 2014 | First release. |

**Sales and Further Information** www.zmdi.com wpan@zmdi.com

| | | | | |
|---|---|---|---|---|
| **Zentrum Mikroelektronik Dresden AG**<br>Global Headquarters<br>Grenzstrasse 28<br>01109 Dresden, Germany<br>Central Office:<br>Phone +49.351.8822.306<br>Fax +49.351.8822.337 | **ZMD America, Inc.**<br>1525 McCarthy Blvd., #212<br>Milpitas, CA 95035-7453<br>USA<br>USA Phone 1.855.275.9634<br>Phone +1.408.883.6310<br>Fax +1.408.883.6358 | **Zentrum Mikroelektronik Dresden AG, Japan Office**<br>2nd Floor, Shinbashi Tokyu Bldg.<br>4-21-3, Shinbashi, Minato-ku<br>Tokyo, 105-0004<br>Japan<br>Phone +81.3.6895.7410<br>Fax +81.3.6895.7301 | **ZMD FAR EAST, Ltd.**<br>3F, No. 51, Sec. 2,<br>Keelung Road<br>11052 Taipei<br>Taiwan<br>Phone +886.2.2377.8189<br>Fax +886.2.2377.8199 | **Zentrum Mikroelektronik Dresden AG, Korea Office**<br>U-space 1 Building<br>11th Floor, Unit JA-1102<br>670 Sampyeong-dong<br>Bundang-gu, Seongnam-si<br>Gyeonggi-do, 463-400<br>Korea<br>Phone +82.31.950.7679<br>Fax +82.504.841.3026 |

**European Technical Support**
Phone +49.351.8822.7.772
Fax +49.351.8822.87.772

**European Sales** (Stuttgart)
Phone +49.711.674517.55
Fax +49.711.674517.87955

DISCLAIMER: This information applies to a product under development. Its characteristics and specifications are subject to change without notice. Zentrum Mikroelektronik Dresden AG (ZMD AG) assumes no obligation regarding future manufacture unless otherwise agreed to in writing. The information furnished hereby is believed to be true and accurate. However, under no circumstances shall ZMD AG be liable to any customer, licensee, or any other third party for any special, indirect, incidental, or consequential damages of any kind or nature whatsoever arising out of or in any way related to the furnishing, performance, or use of this technical data. ZMD AG hereby expressly disclaims any liability of ZMD AG to any customer, licensee or any other third party, and any such customer, licensee and any other third party hereby waives any liability of ZMD AG for any damages in connection with or arising out of the furnishing, performance or use of this technical data, whether based on contract, warranty, tort (including negligence), strict liability, or otherwise.

**© 2014 Zentrum Mikroelektronik Dresden AG — Rev. 1.00**
All rights reserved. The material contained herein may not be reproduced, adapted, merged, translated, stored, or used without the prior written consent of the copyright owner. The information furnished in this publication is subject to changes without notice.