

Building Chromosome for ARM

Summary

[1 How to setup working environment 3](#_Toc324343084)

[1.1 Introduction 3](#_Toc324343085)

[1.2 Create Folder 3](#_Toc324343086)

[1.3 Eclipse 3](#_Toc324343087)

[1.4 CMake 3](#_Toc324343088)

[1.5 GNU toolchain 3](#_Toc324343089)

[1.6 Miscellaneous tools 4](#_Toc324343090)

[1.7 libusb 4](#_Toc324343091)

[1.8 OpenOCD 4](#_Toc324343092)

[1.9 STM Board: Add TFTP firewall exception 4](#_Toc324343093)

[1.10 STM Board: ARM-JTAG-EW driver 5](#_Toc324343094)

[1.11 TOOLCHAIN & PATH environment variables 5](#_Toc324343095)

[2 How to configure a project and import it in Eclipse. 7](#_Toc324343096)

[2.1 Run Eclipse for the first time 7](#_Toc324343097)

[2.2 CMake Configuration 7](#_Toc324343098)

[2.3 Run Eclipse & Load Project 9](#_Toc324343099)

[3 Eclipse external tools 9](#_Toc324343100)

[3.1 OpenOCD 9](#_Toc324343101)

[3.2 OpenOCD\_Kill 9](#_Toc324343102)

[3.3 Reset\_Run 9](#_Toc324343103)

[3.4 Reset\_Halt 10](#_Toc324343104)

[3.5 Upload 10](#_Toc324343105)

[3.6 Dump 10](#_Toc324343106)

[4 STM Board: Ethernet Bootloader 10](#_Toc324343107)

[4.1 LEDs 10](#_Toc324343108)

[4.2 Buttons 10](#_Toc324343109)

# How to setup working environment

## Introduction

Some of the actions are only required for the STM microcontroller, not for the LM3S. There are labeled with "STM Board".

## Create Folder

Create a folder called <PREFIX>/devenv/toolchain. PREFIX is a arbitrary folder that should be located on your disk.

## Eclipse

Download site: <http://www.eclipse.org/downloads>. Please download *Eclipse IDE for C/C++ Developers Helios SR2.* More recent version of Eclipse (i.e., Indigo) are not recommended since they have issues with the problem reporting facilities.

Installation: After downloading it extract it and place in a path named <PREFIX>/devenv/toolchain/eclipse

## CMake

CMake program can be downloaded from the following site: <http://www.cmake.org/cmake/resources/software.html>. After downloading it can installed by running the .exe file. At least version 2.8.6 is required (2.8.7 or higher is recommended)

## GNU toolchain

The arm-gcc compiler can be found and downloaded from the following site

<http://www.codesourcery.com/sgpp/lite/arm> (Note: registration is required). After downloading, it should be extracted to <PREFIX>/devenv/toolchain. It is recommended to use version 2011.03-42. Extract it to the folder devenv\toolchain\arm-gcc

Note: Fortiss employees can take over the toolchain from the fortiss Venus server folder [\\VENUS\fb2\Projects\CHROMOSOME\toolchain\ARMv7-M\arm-gcc](file:///\\VENUS\fb2\Projects\CHROMOSOME\toolchain\ARMv7-M\arm-gcc).

## Miscellaneous tools

**This folder which is named Tools is provided by Fortiss (XME-Folder: xme\trunk\tools\toolchain\tools\bin) and contains two executables: The MinGW-make and TFTP. The files in this folder should be located under <PREFIX>/devenv/toolchain/tools/bin.**

## libusb

Download a binary version of libusb for win32 from http://sourceforge.net/projects/libusb-win32/files/libusb-win32-releases/ and install it according to the instructions in the bin folder. A restart is not required.

Note: It is recommended to use the binary provided by Fortiss. The driver can be found in the SVN **XME:** folder **xme\trunk\tools\toolchain\libusb-win32-bin-1.2.6.zip**

## OpenOCD

In the directory*<Prefix>/devenv/toolchain* should exist also a folder named *OpenOCD*. A recent version of OpenOCD must be downloaded from <http://freddiechopin.info/en/download/category/4-openocd>.

Note: It is recommended to use the binary provided by Fortiss. Fortiss employees can take over OpenOCD from the fortiss Venus server folder [\\VENUS\fb2\Projects\CHROMOSOME\toolchain\ARMv7-M\openocd](file:///\\VENUS\fb2\Projects\CHROMOSOME\toolchain\ARMv7-M\openocd).

## STM Board: Add TFTP firewall exception

The following is only required to use the bootloader to upload code to via the

bootloader. As the TFTP connection is blocked from the Windows firewall by default,

a firewall TFTP exception should be added.  
  
 This can be done (in Windows 7) via Control Panel 🡪 Windows-Firewall🡪  
 Allow a program or feature from the Windows-Firewall 🡪 Change Configuration 🡪   
 Allow another program.

In this step the *TFTP.exe* file should be selected which is located under the path:   
 \devenv\toolchain\tools\bin. After selecting and adding it we should see it in the

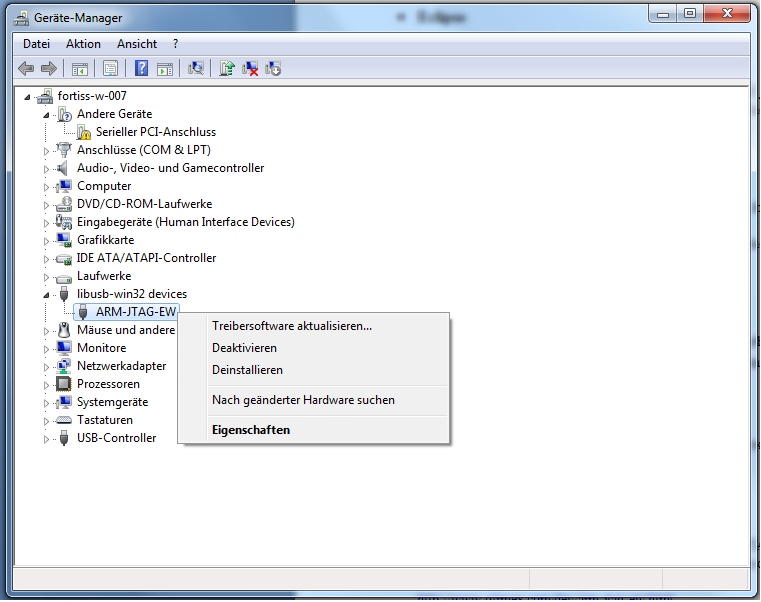
Windows-Firewall exception tab selected with a tick.

## STM Board: ARM-JTAG-EW driver

For debugging the ARM microcontroller we use the ARM-JTAG-EW which is a JTAG probe. The user documentation and the drivers for this can be found and downloaded from the following site:

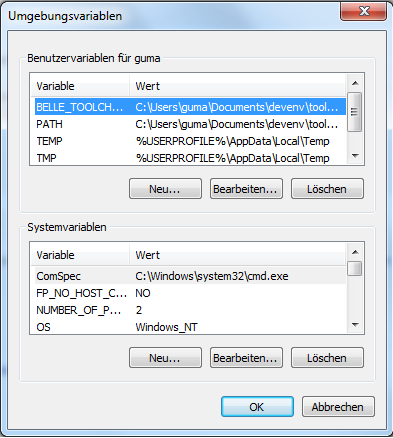
<http://www.olimex.com/dev/arm-jtag-ew.html>.

After downloading and extracting the files, from the Driver-Manager tab we can search and load the driver by a right click on the ARM-JTAG-EW. This tab can be reached via Control Panel 🡪System🡪 Driver Manager (Windows 7).



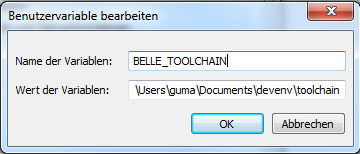
## TOOLCHAIN & PATH environment variables

The environment variable tab can be found via Control Panel 🡪System 🡪 Advanced System Settings 🡪Advanced 🡪 Environment Variables.

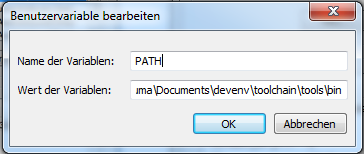


Here we can insert our variables. The *BELLE\_TOOLCHAIN* variable should be

located under <Prefix>\devenv\toolchain path as shown below.



The *PATH* variable should be located under <Prefix>\devenv\toolchain\tools\bin path as shown below.



# How to configure a project and import it in Eclipse.

In order to configure and import a project with Eclipse the following steps should be

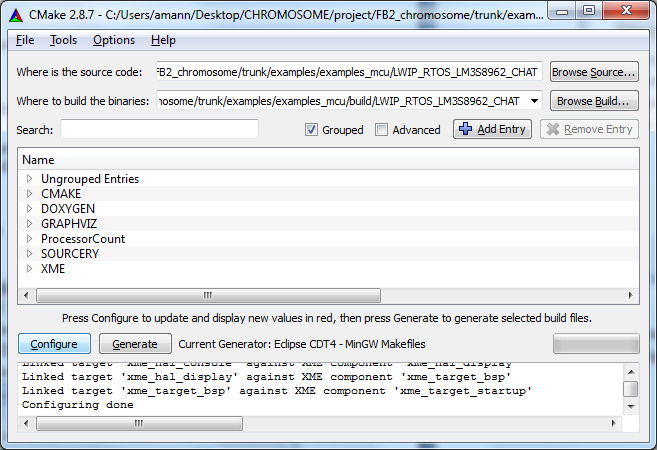
followed:

## Run Eclipse for the first time

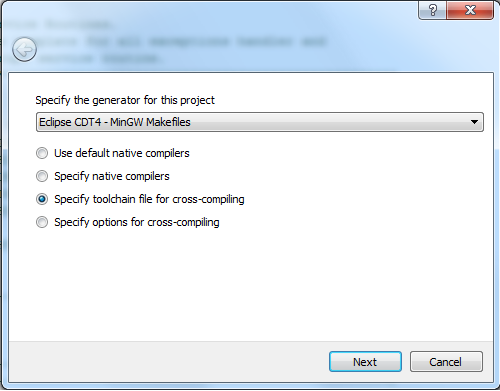
Run Eclipse once (workspace: <Prefix>\devenv\workspace) and after that close it. This is for the initialization of the eclipse workspace

## CMake Configuration

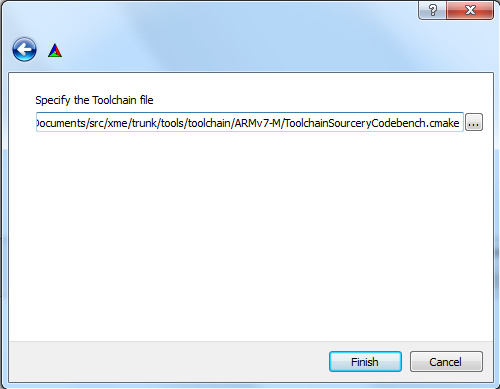
After installing and running the *CMake* program we have to define the location of the source code ( E.g. a subfolder of XME-Folder: xme\examples\examples\_mcu that matches your board) and the location in which we would like the binaries to be built. This is shown in the below figure. Do not copy the folder from the SVN source folder to another location because it is not going to build from there.



When pressing the Configure button for the first time we have to select the generator which is *Eclipse CDT4–MinGW Makefiles* and select the *Specify toolchain file for cross-compiling* as shown in the below figure.



After pressing the next button in the Specify the Toolchain file the *ToolchainSourceryCodebench.cmake* file should be selected as shown in the below figure.



After performing the above (only the first time after *CMake* installation) we make the configuration 2 times (by pressing twice the Configure button) and after that, generate the files (by pressing the Generate button).

## Run Eclipse & Load Project

After running *Eclipse* for the second time we can import the project created from the

*CMake* program. The path is the same with the one that we specified in

the *CMake* program, on where to build the binaries. In eclipse these files can be

imported via Files🡪Import🡪Existing files in the workspace🡪Select Root

Directory🡪Browse. Do not check "Copy into workspace".

By right-clicking on the project and selecting Build project the project can be build.

Note: In case of a problem with Java, uninstall all your Java versions and Install the latest JDK from <http://www.oracle.com/technetwork/java/javase/downloads/jdk-7u3-download-1501626.html>

# Eclipse external tools

After importing the project in Eclipse, by using the external tools (can be found via Run🡪External Tools) connection with the ARM device can be established, the project can be flashed on the device and it can be run. These tools are installed if the workspace has been recognized correctly by cmake. Please note that connect must be executed before doing anything with the device.

## OpenOCD

OpenOCD is responsible for talking with the device.

## OpenOCD\_Kill

OpenOCD\_Kill is responsible for disconnecting with the device.

## Reset\_Run

Reset\_Run is responsible for running the application.

## Reset\_Halt

Reset\_Halt is responsible for resetting the application.

## Upload

Upload is responsible for flashing (inserting the data) to the device.

## Dump

Dump is responsible for reading the contents of the entire flash and dumping those in a file.

# STM Board: Ethernet Bootloader

## LEDs

PE8 & PE9: These two LEDs have a special functionality.

-***LED PE8*** shows the state of the boot loader. When this LED blinks 3 times then it

means bad or missing application. Permanently ON means that it is waiting for a

connection. Also when each packet is received this LED is toggled once.

- ***LED PE9*** shows the IP address whether it is static or dynamic. When it blinks 3

times means that it makes a DHCP connection. When green means that it has an IP

address.

## Buttons

S1 & S2 & S4: Each of these three buttons has a special functionality:

***- Button S1(User)***: When this button is pressed and the application is good then it

enters in boot loader mode.

***- Button S2(Tamper):*** When this button is pressed thena static IP is entered.

- ***Button S4(Reset)***: With this button we reset our program and start again from the

beginning.