

Linux-Kit
phyFLEX-i.MX6
Quickstart Instructions

Using Eclipse and the GNU Cross Development Toolchain

Note: The PHYTEC Linux-phyFLEX-i.MX6-Disc includes the electronic version of the English phyFLEX-i.MX6 Hardware Manual

Edition March 2013

A product of a PHYTEC Technology Holding company

Copyrighted products are not explicitly indicated in this manual. The absence of the trademark (™) and copyright (©) symbols does not imply that a product is not protected. Additionally, registered patents and trademarks are similarly not expressly indicated in this manual.

This product contains software components that are licensed by the holder of the rights as free software, or open source software, under GNU General Public License, Version 2. For 5 euros (to cover the costs of providing the data carrier and shipping) you can obtain the source code for these software components from us on a data carrier (CD or DVD) by submitting a request to our customer service department at the following address within three years after we distribute the product:

Phytec Messtechnik GmbH

Phytec Support

Robert-Koch-Str. 39

55129 Mainz

Phone: +49 6131 9221 31

Source code: phyFLEX-i.MX6

The information in this document has been carefully checked and is considered to be entirely reliable. However, PHYTEC Messtechnik GmbH assumes no responsibility for any inaccuracies. PHYTEC Messtechnik GmbH neither gives any guarantee nor accepts any liability whatsoever for consequential damages resulting from the use of this manual or its associated product. PHYTEC Messtechnik GmbH reserves the right to alter the information contained herein without prior notification and accepts no responsibility for any damages that might result.

Additionally, PHYTEC Messtechnik GmbH offers no guarantee nor accepts any liability for damages arising from the improper usage or improper installation of the hardware or software. PHYTEC Messtechnik GmbH further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

© Copyright 2013 PHYTEC Messtechnik GmbH, D-55129 Mainz.

Rights - including those of translation, reprint, broadcast, photo-mechanical or similar reproduction and storage or processing in computer systems, in whole or in part - are reserved. No reproduction may be made without the explicit written consent from PHYTEC Messtechnik GmbH.

	EUROPE	NORTH AMERICA
Address:	PHYTEC Messtechnik GmbH Robert-Koch-Str. 39 D-55129 Mainz GERMANY	PHYTEC America LLC 203 Parfitt Way SW, Suite G100 Bainbridge Island, WA 98110 USA
Ordering Information:	+49 (6131) 9221-30 order@phytec.de	1 (800) 278-9913 sales@phytec.com
Technical Support:	+49 (6131) 9221-31 support@phytec.de	1 (800) 278-9913 support@phytec.com
Fax:	+49 (6131) 9221-33	1 (206) 780-9135
Web Site:	http://www.phytec.de http://www.phytec.eu	http://www.phytec.com

Editon March 2013

Table of Contents

1	Introduction	1	
1.1	Rapid Development Kit Documentation.....	1	
1.2	Professional Support Packages Available.....	1	
1.3	Overview of these QuickStart Instructions	2	
1.4	Conventions used in this QuickStart	2	
1.5	System Requirements	4	
1.6	Contents of the Linux-phyFLEX-i.MX6-Kit-DVD	4	
1.7	About the Ubuntu distribution	5	
2	Getting started.....	6	
2.1	Some notes before starting	6	
2.2	First impressions of the Live DVD	7	
2.3	Starting the Hardware.....	9	
2.4	Copying an Example to the Target.....	14	
2.4.1	Copying a Program to the Target	14	
2.4.2	Using Telnet to execute a Program on the Target	16	
2.4.3	Using SSH to execute a Program on the Target	17	
2.5	Advanced Information	19	
2.5.1	Copying a Program to the Target with the command line	19	
2.5.2	Executing a program on the target	19	
2.5.3	Executing a program directly on the target using SSH	19	
3	Working with Eclipse.....	20	
3.1	Programming in the C/C++ perspective	20	
3.1.1	Handle with the demo project.....	20	
3.1.2	Creating a New Project	27	
3.1.3	Changing the Demo Application	37	
3.1.4	Starting a Program out of Eclipse on the Target.....	40	
3.2	Programming in the Qt C++ perspective	43	
3.2.1	Importing the demo application.....	43	
3.2.2	Handle with the demo application	50	
4	Debugging an example project.....	52	
4.1	Starting the GDB server on the target.....	52	
4.2	Configuring and starting the debugger in Eclipse.....	53	
4.3	Setting a Breakpoint.....	59	
4.4	Stepping and Watching Variable Contents	60	
4.5	Stepping and Watching Variable Contents	63	
4.6	Using the Memory Monitor	65	

5 Summary..... 68

6 Installing Linux on the phyFLEX-i.MX6 69

6.1 Configure Barebox Environments Variables69

6.2 Restoring the Barebox Default Configuration.....72

6.3 Update the Bootloader72

6.4 Writing the Kernel into Flash73

6.5 Writing the root filesystem into Flash.....74

7 Setup your own Linux-Host-PC 77

7.1 Essential settings77

7.1.1Installation of software packages.....77

7.1.2Setup of toolchain, PTXdist and BSP78

7.1.3Setting up Eclipse and integrate plug ins80

7.1.4Setting up a TFTP server81

7.2 Optional settings82

8 Installation of the modified Ubuntu..... 83

1 Introduction

**5 min**

In this QuickStart Instructions Manual you will find general information about the PHYTEC phyFLEX-i.MX6-Kit and instructions how to start-up with the phyFLEX-i.MX6. You will learn how to...

- ... connect to the target in different ways
- ... working with Eclipse and running example programs on the target with the GNU GCC C/C++ Cross-Development Toolchain
- ... and much more good to know things

This first Chapter gives a short introduction about the PHYTEC phyFLEX-i.MX6-Kit and it's Quickstart. Also you will find general requirements and information to successfully pass the Quickstart.

Please refer to the phyFLEX-i.MX6 Hardware Manual® for specific information on such board-level features as jumper configuration, memory mapping and pin layout. At this point we also mention the BSP-Quickstart phyFLEX-i.MX6 manual in which you will find more detailed instructions on how to handle the phyFLEX-i.MX6

1.1 Rapid Development Kit Documentation

This "Rapid Development Kit" (RDK) includes the following electronic documentation on the enclosed "PHYTEC Linux-phyFLEX-i.MX6-Disc" under *PHYTEC/Documentations* or if you are in the Live Environment under */opt/PHYTEC_Tools/Documentation*.

- The PHYTEC phyFLEX-i.MX6 Hardware Manual
- These Quickstart Instructions

1.2 Professional Support Packages Available

If you have any questions concerning installation and setup, you are welcome to contact our support department.

For more in-depth questions, we offer a variety of custom-tailored packages with different support options (e-mail, phone, direct contact to the developer) and different reaction times.

Please contact our sales team to discuss the appropriate support option if professional support beyond installation and setup is important for you.

For more information please refer to the following sources:

<http://www.phytec.de>

<http://www.phytec.eu>

support@phytec.de

Also more contact information can be found on *page 2*.

1.3 Overview of these QuickStart Instructions

This QuickStart gives you an overview of the phyFLEX-i.MX6. It's content ranges from the first startup, to building your own kernel and file system, to the point of building your own program with Eclipse. This Quickstart is structured as follows:

1. The "*Getting Started*" section describes the basics such as configure your host platform and starting the phyFLEX-i.MX6 platform.
2. The "*Programming with Eclipse*" section explains how to modify an example application, create and build a new project, and copy programs to the phyFLEX-i.MX6 using Eclipse with the C/C++ and QT-Plug in.
3. The "*Debugging*" section provides information on how to debug an application with the Eclipse debugging interface.
4. In the Appendix you will find an overview how to install the Live DVD on your system and how to setup your own Linux-Host-PC if you don't want to use our Live DVD.

1.4 Conventions used in this QuickStart

The following is a list of the typographical conventions used in this book:

<i>italic</i>	Used for file and directory names, program and command names, command-line options, menu items, URLs, and other terms that correspond to the terms on your desktop
Bold	Used in examples to show commands or other text that should be typed literally by the user.

Pay special attention to notes set apart from the text with the following icons:

	<p>At this icon you might leave the path of this QuickStart.</p>
	<p>This is a warning. It helps you to avoid annoying problems.</p>
	<p>You can find useful supplementary information about the topic.</p>
	<p>At the beginning of each chapter you can find information about the time required to read the following chapter.</p>
	<p>You have successfully completed an important part of this QuickStart.</p>
	<p>You can find information to solve problems.</p>

1.5 System Requirements

The following items will be needed to complete this Quickstart successfully :

- The PHYTEC phyFLEX-i.MX6 (i.MX6)
- The PHYTEC Development Board with the included DB-9 serial cable, Ethernet cross-over cable and AC adapter supplying 12 VDC (min. 2 A)
- PHYTEC Linux-phyFLEX-i.MX6-Kit-DVD
- An IBM-compatible host-PC (586 or higher) with 512 MB RAM (or more)
- DVD-drive
- Recommended free disk space: 20 GB if you want to install Ubuntu with our customization

1.6 Contents of the Linux-phyFLEX-i.MX6-Kit-DVD

There is a bootable modified *Ubuntu* distribution on the Linux-phyFLEX-i.MX6-Kit-DVD. You can run *Ubuntu* directly from your DVD without affecting your current system. Therefore you have the possibility to work with this Quickstart without installing the operating system. To allow a fast and smooth procedure some modifications are applied to the original Ubuntu. A short overview about these modifications:

- Design customization
- Installation of required software to work with this Quickstart
- Preparation of the Toolchain for cross-compilation
- Integration of the PHYTEC Board Support Package

You can find a detailed list in the appendix "*Setup your own Linux-Host-PC*".

1.7 About the Ubuntu distribution

Ubuntu - which you can find on the Linux-phyFLEX-i.MX6-Kit-DVD - is a free and open source operating system based on *Debian Linux*. Basically it is designed for desktop use. Web statistics suggest that *Ubuntu* is one of the most popular operating systems in the Linux desktop environment.

The *Ubuntu* release which we deliver is *10.04.3* and was released on 22 July 2011. *Ubuntu* 10.04 code name "*Lucid Lynx*" is designated as a **Long Term Support (LTS)** release and the first stable release was on 29 April 2010. LTS means that it will be supported and updated for three years.

Our *Ubuntu* version comes with *GNOME* as desktop environment, *dpkg* as package management system, the update method is based on *APT (Advanced Packaging Tool)* and the user space uses *GNU*.

2 Getting started



35 min

In this chapter we establish a basis to go through the steps in this Quickstart. First you will learn more about the Ubuntu Live DVD and its handling. Then you will be starting the phyFLEX-i.MX6 platform for the first time.

2.1 Some notes before starting

As mentioned in the beginning of this Quickstart the Linux-phyFLEX-i.MX6-Kit-DVD is an Ubuntu Live DVD with some modifications. This means that you have the freedom to choose if you want to first test the platform under the live-environment or if you will directly install Ubuntu on your hard drive. In the following you can find a list of advantages and disadvantages from the perspective of the live-environment to simplify your decision:

Live-environment vs. Installation	
Pro	Contra
<ul style="list-style-type: none"> starting directly without preparation if anything goes wrong, your system is clean after a reset testing without affecting your current system all Quickstart steps can be done 	<ul style="list-style-type: none"> slower performance after a restart all changes are lost depending on the size of your RAM you can only do a limited number of changes not intended for productive use

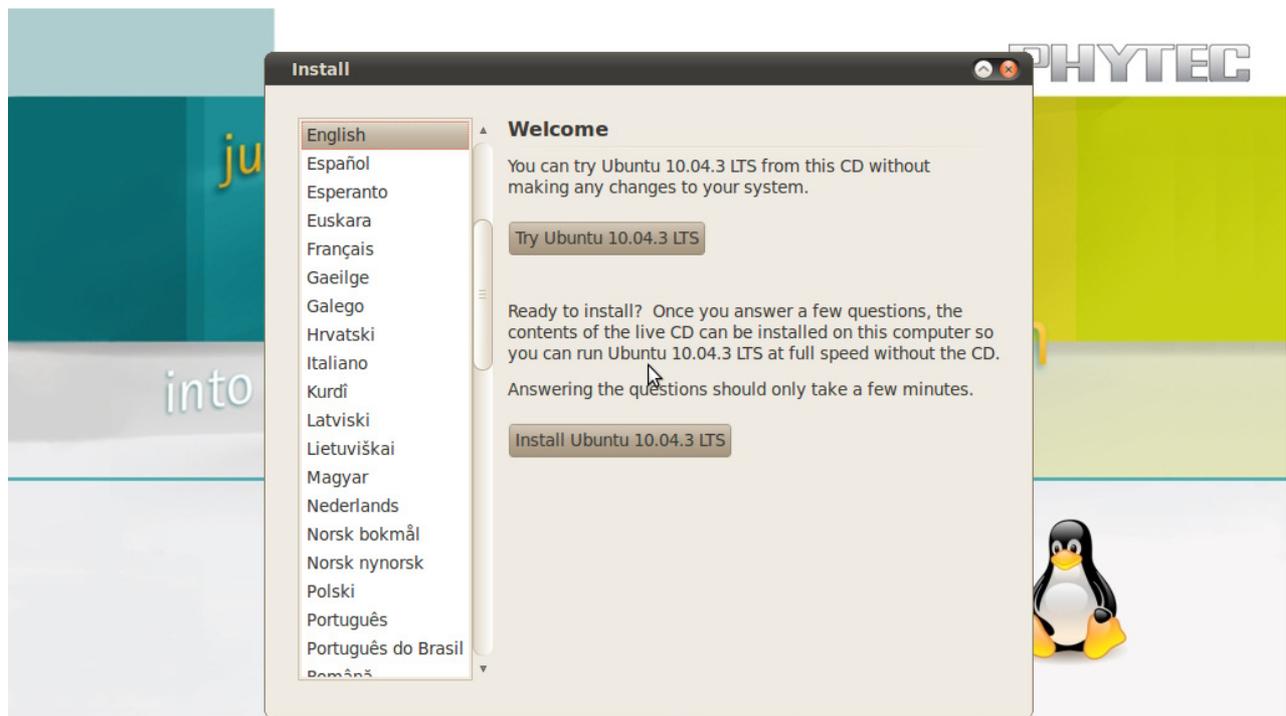


Of course more involved users can use their own - perhaps existing - distribution. In the appendix "*Setup your own Linux-Host-PC*" you will find a list of modifications which have been made to *Ubuntu*. These modifications can be done specifically to your system but we don't ensure support if you have any system specific problems. Also you can find the essential programs on our DVD under the directory */PHYTEC*

2.2 First impressions of the Live DVD

After reading the general notes it's now time for you to start activity. Regardless of which choice you made it is necessary to change the boot priority in the *BIOS*. Your system should first access to the DVD-drive and check if there is a bootable device. We are starting the Live DVD by inserting the Linux-phyFLEX-i.MX6-Kit-DVD and boot from the DVD-drive.

When the system has completely booted the following screen appears:

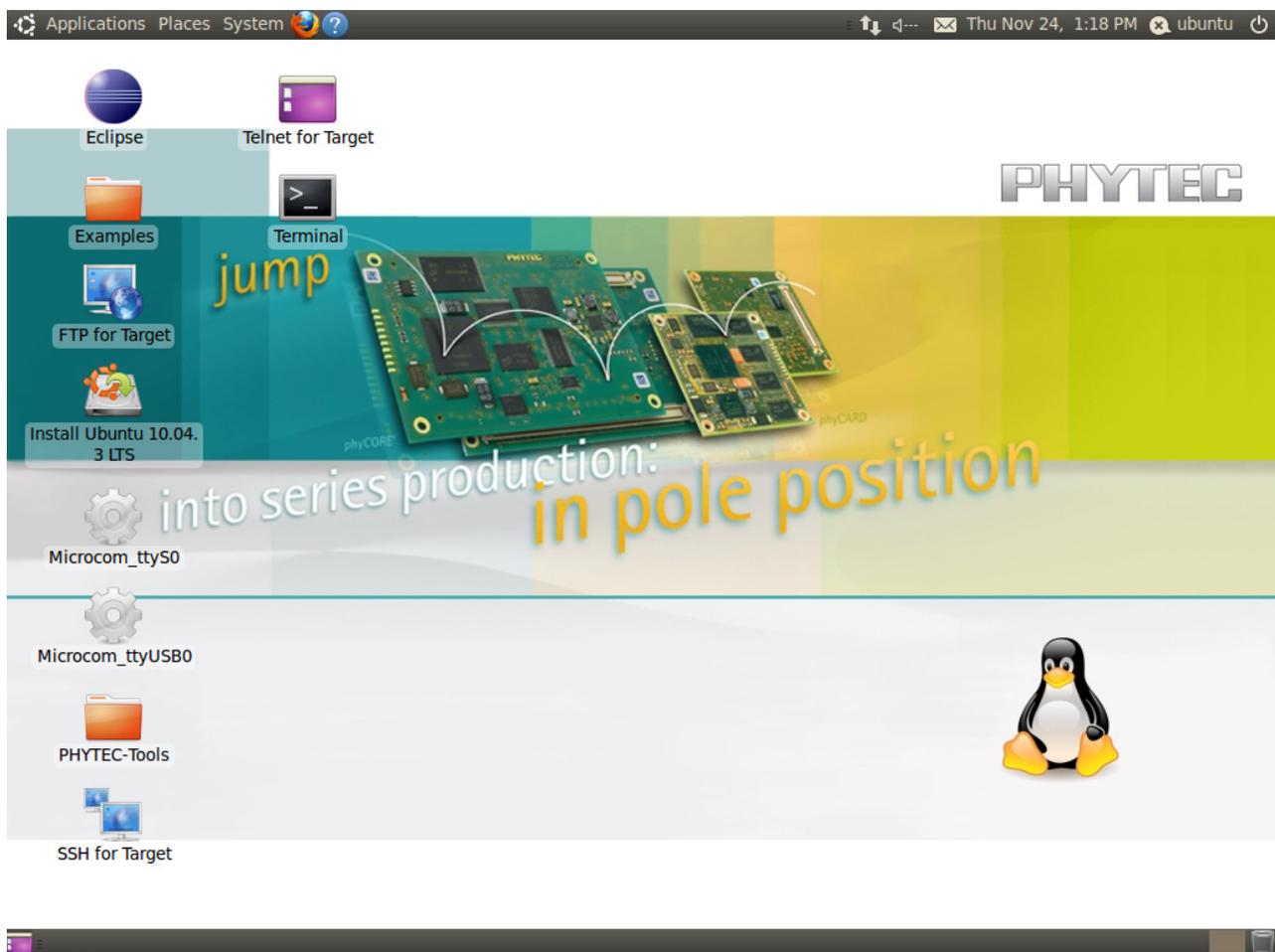


In this window you can choose your desired language and you can also choose between installing the operating system or only testing the system. In the following we assume that you are working with the live-environment.



If you want to install our Live DVD now you will find the relevant instructions in the appendix "*Install the Live DVD on your host*". After having completed this you can go back to this position and continue your work.

After clicking on "Try Ubuntu 10.04.3" the live-environment is loaded and the system welcomes you with the following desktop view.



The operating system is booted and in the next chapter we will focus on your first contact with the target.

2.3 Starting the Hardware

In this section you will learn how to connect your host PC to the target. The connection will be done using a cross-over Ethernet cable and a serial one-to-one cable. You will start Linux from the flash memory on the target and you will be able to log in with the serial communication program *Microcom* as well as via a *telnet* session using a peer-to-peer network connection.



By default every input and output is transmitted over the serial connection you built up earlier.

- Connect the serial cable to the UART0 (connector X51, TOP) port on the target and the first serial interface on your host



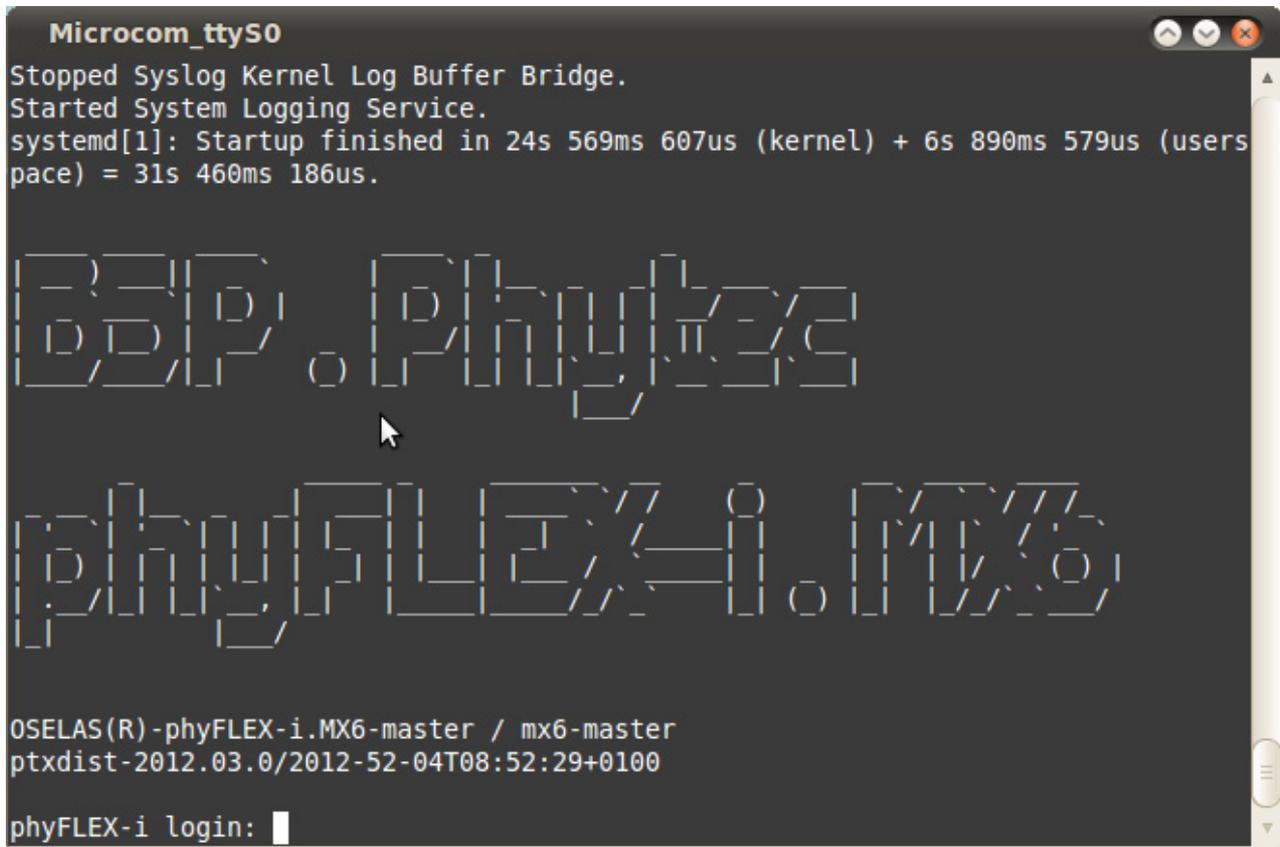
Be sure to use the one-to-one serial cable included in this Rapid Development Kit.

- Connect the cross-over Ethernet cable to the Ethernet connector (POE X28) on the target and to the appropriate network card of your host



- Click the *Microcom_ttyS0* icon on your desktop, if you have connected the target to your serial interface on your host. If you have connected the serial cable with an USB dongle to your PC click the *Microcom_ttyUSB0* icon on your desktop.
- Connect the AC adapter to the power supply connector PWR (12V) on your board

After connecting the board to the power supply, the target starts booting. When the target has finished loading the system, you should see a screen similar to the following:



```
Microcom_ttyS0
Stopped Syslog Kernel Log Buffer Bridge.
Started System Logging Service.
systemd[1]: Startup finished in 24s 569ms 607us (kernel) + 6s 890ms 579us (users
pace) = 31s 460ms 186us.

Barebox Phytec

phyFLEX-i.MX6

OSELAS(R)-phyFLEX-i.MX6-master / mx6-master
ptxdist-2012.03.0/2012-52-04T08:52:29+0100

phyFLEX-i login: █
```

- Type **root** to log in
- After you have successfully logged in, you can close *Microcom*

If you don't see the Barebox and Linux starting and don't get a login prompt, you probably have a kit with Windows CE pre-installed. Please refer to the chapter "Installing Linux on the phyFLEX-i.MX6" for instructions on how to install Linux in such a case.



When the target is connected to the power supply, the boot loader *Barebox* is first loaded from the flash memory. Then the boot loader is uncompressing and booting the Linux kernel from the flash. Then the kernel will install the root file system, which is also located in the target's flash.



If you don't see any output in the *Microcom* window, check the serial connection between the target and your host.

If you have more than one serial port try the others. By default *Microcom* uses `/dev/ttyS0`. If you want to use another port you can click on the *Microcom* Icon at your desktop with your right mouse button and select *Properties*. A window opens in which you can change the Properties of the *Microcom* Icon. Click in the *Command* field and search for `"-p /dev/ttyS0"`. Change it for example to `/dev/ttyS1` and so on if you have more than one serial port. If you are connecting the board via RS232 to USB change it to `/dev/ttyUSB0`.

The `-p` Parameter defines the port which *Microcom* will use.

If the *Microcom* window does not open, one reason could be a *lock*-file which was created when *Microcom* was not correctly terminated.

Delete this file by opening a terminal and type:

```
rm /var/lock/LCK.*
```

After starting the target and see the serial outputs in *Microcom* we want to connect to the target via Ethernet.

Before we can start connecting to the target we must configure the IP address of our host.



We recommend that the host PC is not connected to any other network. The target and host will be connected with a cross-over cable via a peer-to-peer connection. If your host is part of a company's network, we recommend disconnecting your host from such a network.

- In the GNOME-Panel at the top of the desktop click on *System* ► *Preferences* ► *Network Connections*
- Choose the right wired network if more than one is present and click on *Edit*
- Select the *IPv4 Settings* register and select *Manual* in the *Method* drop-down box
- Click on *Add* and enter as IP address **192.168.3.10** and subnet mask **255.255.255.0**
- At last click on *Apply* to save this connection and close the windows

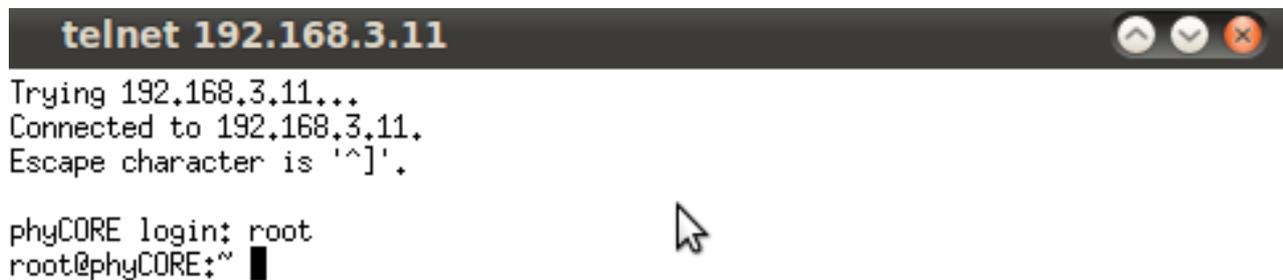
Now we are ready to test the network connection.



Telnet for Target

- Click the *Telnet for Target* icon on your desktop

A new window with a connection to the target opens.

A screenshot of a terminal window titled 'telnet 192.168.3.11'. The window shows the process of connecting to the IP address 192.168.3.11. The output text is: 'Trying 192.168.3.11...', 'Connected to 192.168.3.11.', and 'Escape character is '^]'. Below this, the login prompt 'phyCORE login: root' is shown, followed by the user input 'root@phyCORE:~' and a cursor. A mouse cursor is visible over the terminal text.

```
telnet 192.168.3.11
Trying 192.168.3.11...
Connected to 192.168.3.11.
Escape character is '^]'.

phyCORE login: root
root@phyCORE:~ █
```

If you can see the user login in the opened window, the network was configured correctly.

- Close the window

**Troubleshooting:**

If you don't see the user login, check the Ethernet connection between the target and the host. If you have installed more than one network card on your host, be sure to connect the cable to the network card which you have configured with the IP address **192.168.3.10**.

If you do not see the login, you may not have set up the right IP address for your host. You can check the settings of your network card by clicking on *System Preferences Network Connections*.

Another common reason is that the MAC address of the target isn't set correctly. You can check this by opening *Microcom* if it is not already started and reset the target. Press any key to stop autoboot and type: **edit /env/config**

Search for a line beginning with "*eth0.ethaddr*". After finding this entry compare it to the MAC address which you find on the target. If you don't find this line, insert it manually, followed by the MAC address which you find on the target. For example **eth0.ethaddr=00:50:C1:D1:F1:E1**. After you have made the change quit the program by pressing **CTRL+D**, type **save** and reset the target.

You have successfully set up all configurations to access your phyFLEX-i.MX6 from your host.

2.4 Copying an Example to the Target

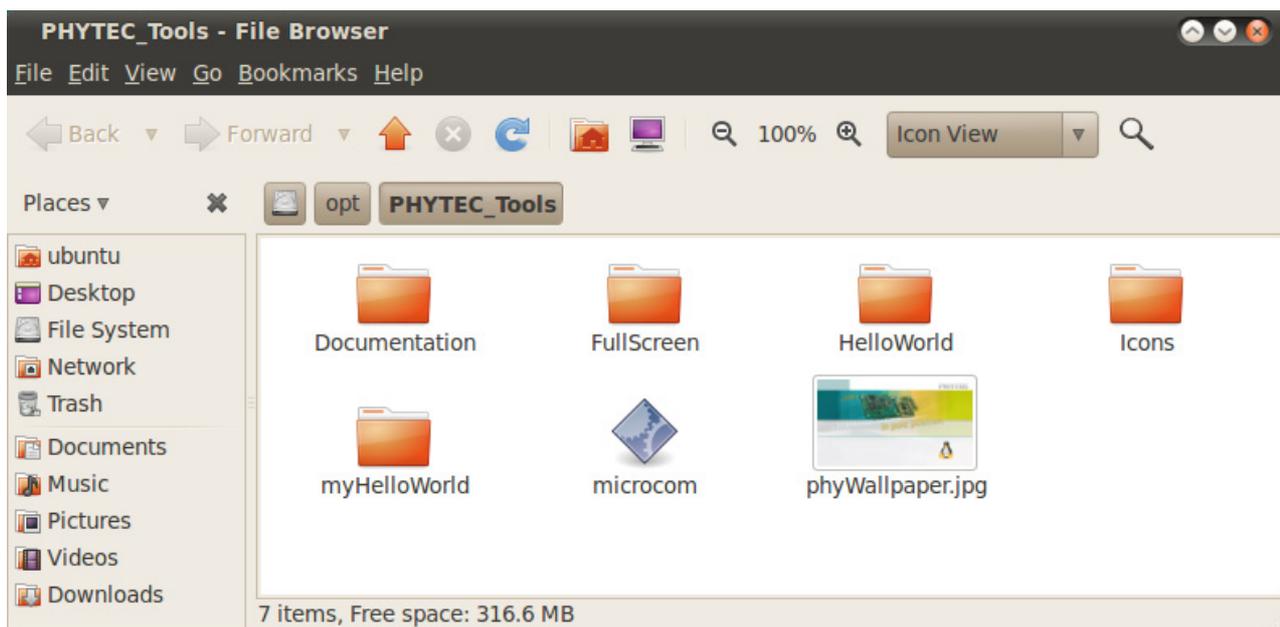
In this section you will learn how to copy an example program to the target using the FTP protocol with the *Nautilus* file browser. After that you will execute an example on the target. At the end of this passage you can find some information on how to copy and execute a file on the target using the command line.

2.4.1 Copying a Program to the Target



- First click the *PHYTEC-Tools* icon on your *GNOME* desktop

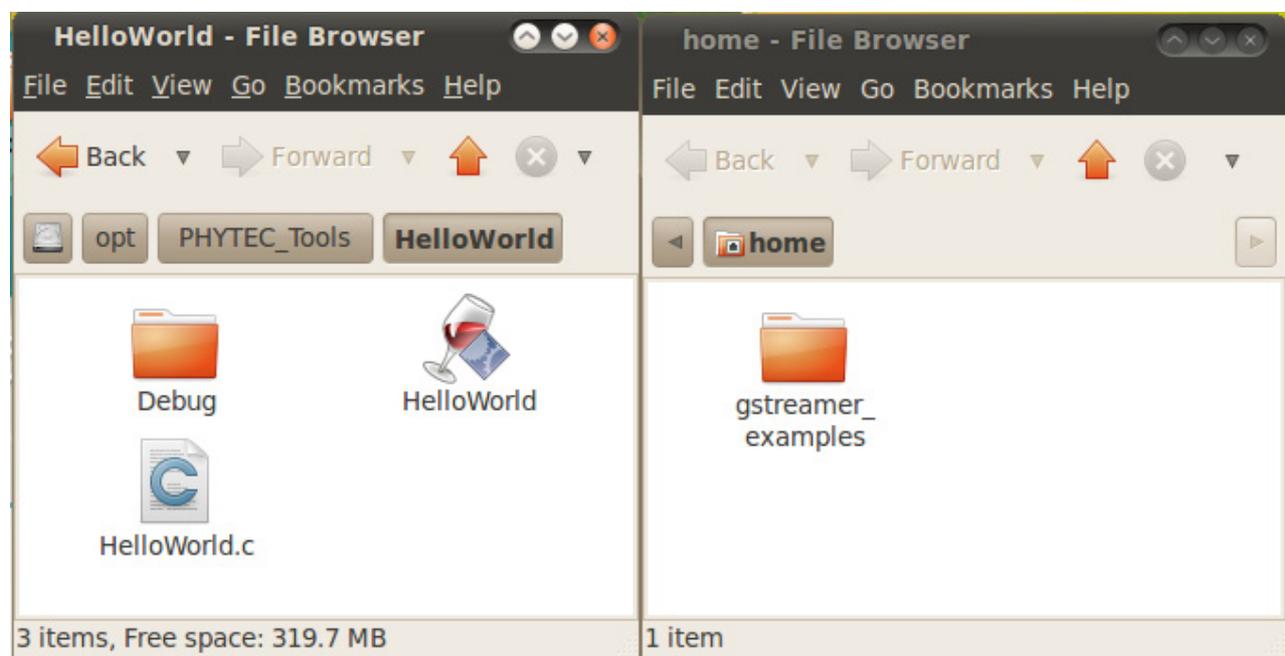
A new window with the contents from the PHYTEC_Tools directory opens.





- Click the *FTP for Target* icon on your desktop

A window with an FTP session with the target opens. User is *root* and password is empty. Now you have two windows opened, one for the target and one for the host. You can use these two windows to copy files per “drag and drop” from the host to the target (and vice versa).



- Select the window that lists the *HelloWorld* program on your hard disk
- Click the *HelloWorld* program with the right mouse button and choose *Copy*
- Click with the right mouse button on the FTP session with the target and choose *Insert*
- Close the two windows

2.4.2 Using Telnet to execute a Program on the Target



Telnet for Target

- Click the *Telnet for Target* icon on your *GNOME* desktop

```
telnet 192.168.3.11
Trying 192.168.3.11...
Connected to 192.168.3.11.
Escape character is '^]'.

phyCORE login: root
root@phyCORE:~ █
```

- Enter **root** as login and press **Enter**
- Enter **./HelloWorld** and press **Enter**

The program starts and you should see the following output:

Welcome to the World of the phyFLEX-i.MX6!

2.4.3 Using SSH to execute a Program on the Target

SSH can be used if you want to execute a program directly from the host on the target. Later, this will be used to execute programs out of Eclipse on the target. Before you can start programs out of Eclipse, you have to log in to the target via SSH from the command line for the first time. This is necessary to add the RSA public key of the target to the list of known hosts.



When the host connects to the target, the file `~/.ssh/known_hosts` (on the host) is consulted when using RSA host authentication to check the public key of the target. The key must be listed in this file to be accepted. When the host connects to the target for the first time, you will be asked to store the target's RSA public key to your `~/.ssh/known_hosts`. If you agree to do this, then the host will be able to connect to the target without entering a password.



- Click the *SSH for Target* icon on the desktop

A new window opens.

```
The authenticity of host '192.168.3.11 (192.168.3.11)' can't be established.  
RSA key fingerprint is 28:2d:35:dd:47:0d:f3:a3:42:c1:37:ab:95:63:77:c0.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added '192.168.3.11' (RSA) to the list of known hosts.  
root@phyCARD:~
```



In this window you can see that the authenticity of the phyFLEX-i.MX6 can't be established. This is normal if you want to create an SSH connection for the first time.

- Enter **yes** and press **Enter** to continue. The RSA public key of the target will be permanently added to the list of the known hosts.

**Troubleshooting:**

If an error occurs and you can't see the `root@phyFLEX:~>` prompt, open a terminal window and enter the following command:

```
rm ~/.ssh/known_hosts
```

Try to log in again by entering:

```
ssh root@192.168.3.11
```

Enter **yes** to add the target to the list of known hosts.

Now you should see the target's prompt.



We expect that you did not change the SSH configuration file on your host. If you have changed this file, the authentication may not work.

Now you are logged in and you can execute programs on the target.

- Type `./HelloWorld` to start the program you had copied to the phyFLEX-i.MX6 before

The program starts and you should see the following output:

Welcome to the World of the phyFLEX-i.MX6!

- Close the SSH window



You have successfully copied and executed an example application on the target.

2.5 Advanced Information

2.5.1 Copying a Program to the Target with the command line

- Open a new terminal window
- Change to `/opt/PHYTEC_Tools/`:
cd /opt/PHYTEC_Tools/
- Copy the application to the target by typing:
scp HelloWorld root@192.168.3.11:/home/
Be sure to enter a slash followed by a space after the IP address

2.5.2 Executing a program on the target

- Open a Telnet session to the target:
telnet 192.168.3.11
- Type **root** and press **Enter**
- Type **./HelloWorld** to start the application
- Type **exit**

2.5.3 Executing a program directly on the target using SSH

- To start the program, type:
ssh root@192.168.3.11 ./HelloWorld
After the program has finished, SSH will log out automatically

3 Working with Eclipse

**35 min**

With the help of example projects, we will teach you how to work with eclipse during this chapter. First we take a look on the C programming language. At the end of this chapter we explain how to execute your written programs automatically when booting the target.

3.1 Programming in the C/C++ perspective

We are starting with the C/C++ workbench. Therefore you will import an existing Eclipse project into your workspace. The imported example project will be compiled with the cross compiler. After compiling the project, you will copy and execute the newly created program on the target.

3.1.1 Handle with the demo project



- Click the *Eclipse* icon to start the application. You can find this icon on your desktop.

Select a workspace

Eclipse SDK stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace:

Use this as the default and do not ask again

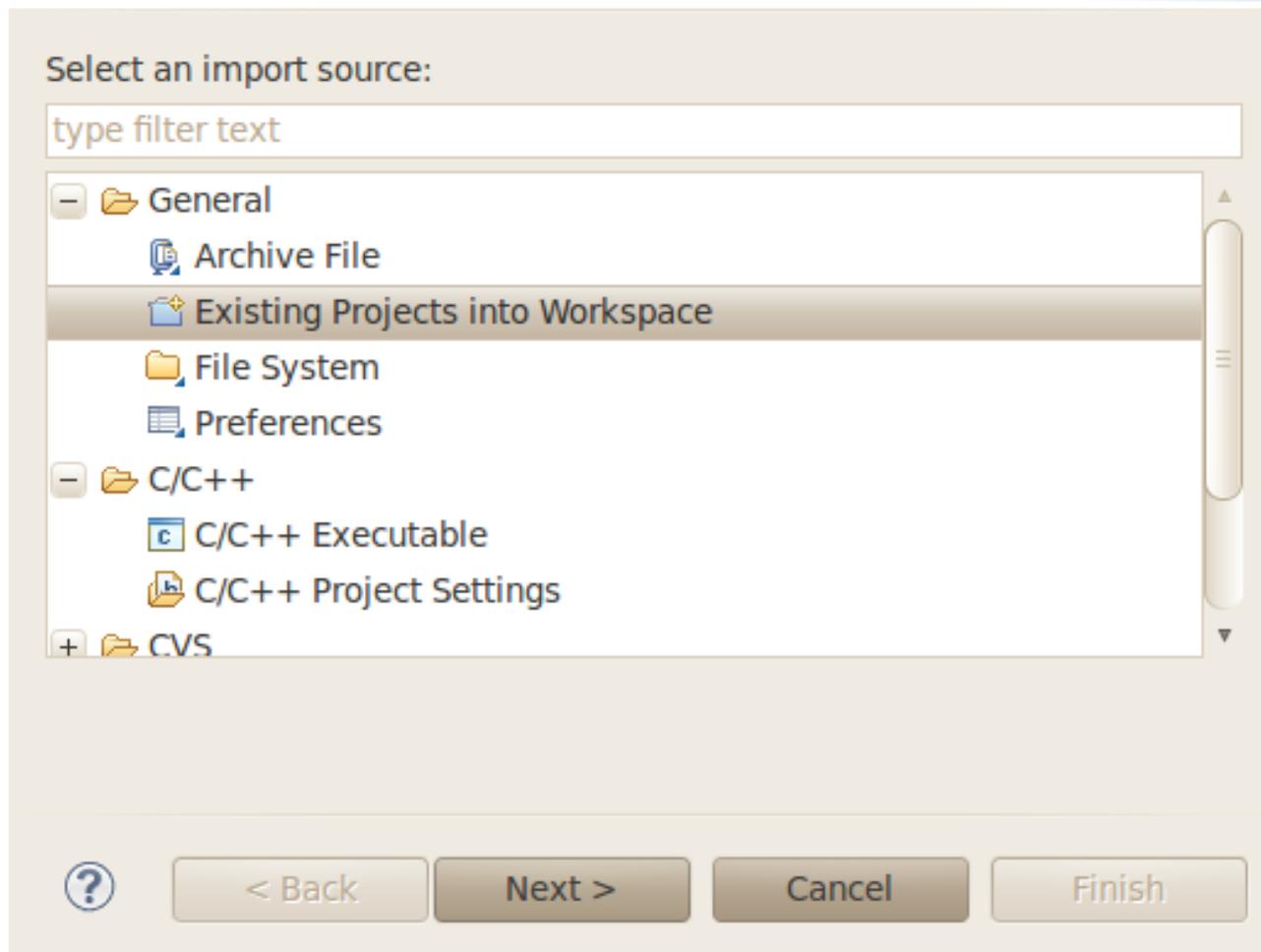
- Confirm the workspace directory with OK
- Close the "*Welcome to Eclipse*" screen by clicking on the "*Go to the workbench*" - button



- Select File ► Import from the menu bar

Select

Create new projects from an archive file or directory.



- Select *Existing Projects into Workspace*
- Click *Next*

Import Projects



Select a directory to search for existing Eclipse projects.

Select root directory:

Select archive file:

Projects:

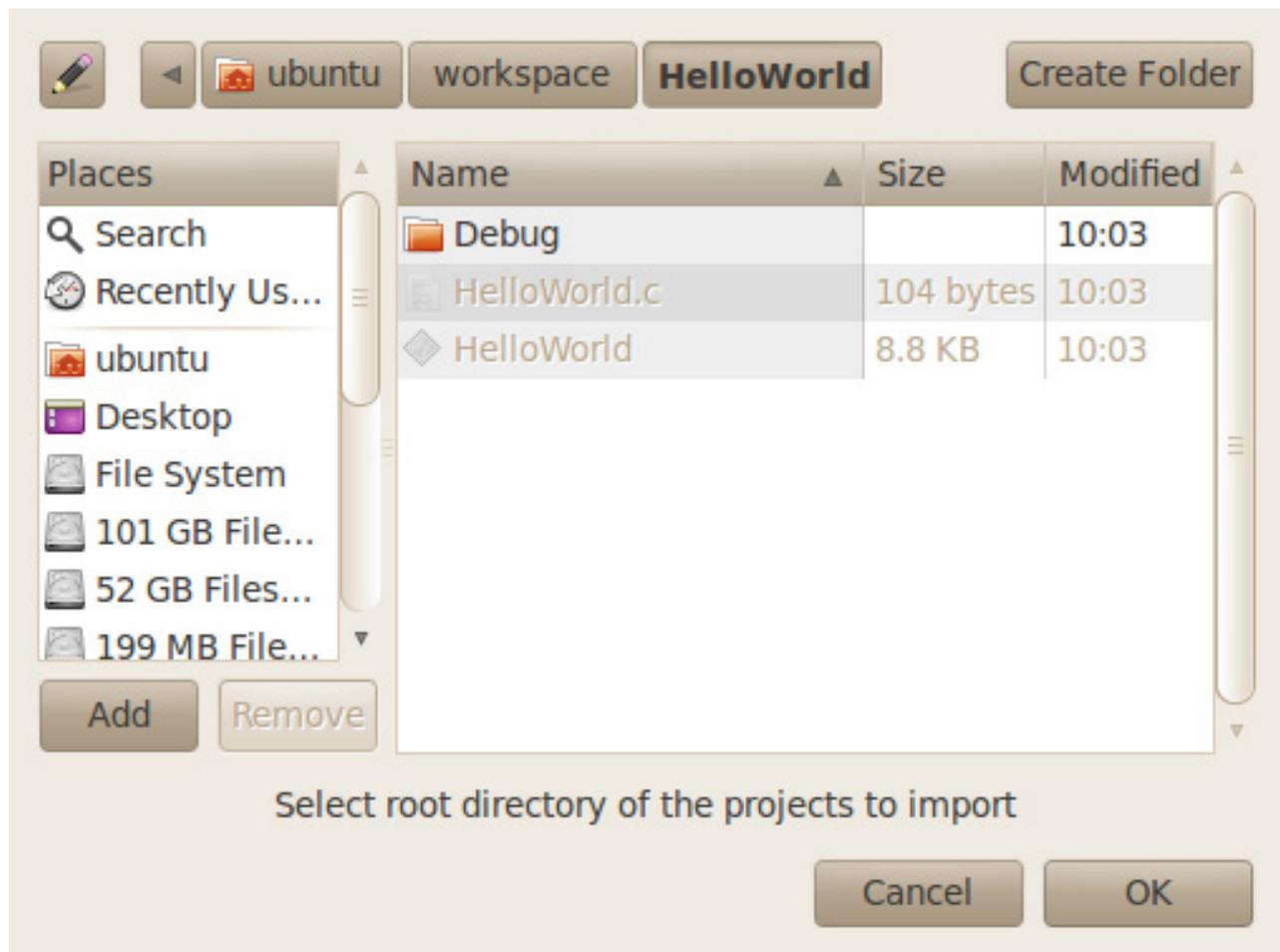
Copy projects into workspace

Working sets

Add project to working sets

Working sets:

- Select *Browse*



- Double-click the *HelloWorld* directory under */home/ubuntu/workspace/*
- Click *OK*

Import Projects

Select a directory to search for existing Eclipse projects.



Select root directory:

Select archive file:

Projects:

<input checked="" type="checkbox"/> HelloWorld (/home/ubuntu/workspace/HelloWorld)	<input type="button" value="Select All"/>
	<input type="button" value="Deselect All"/>
	<input type="button" value="Refresh"/>

Copy projects into workspace

Working sets

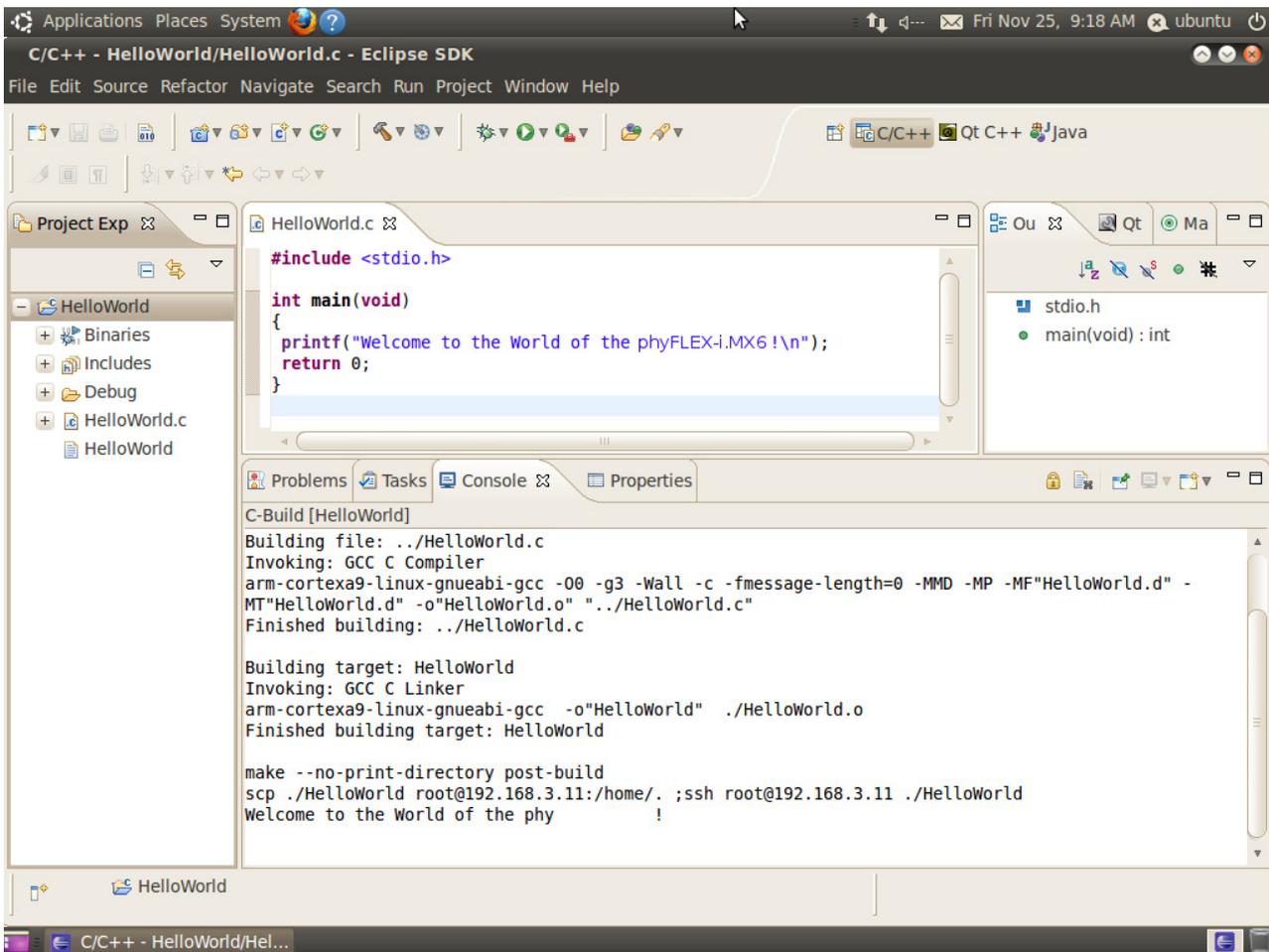
Add project to working sets

Working sets:

- Select *Finish* to import the project

The *HelloWorld* program will be compiled and the *HelloWorld* executable is built for the target. Then the *HelloWorld* file is copied to the target using *secure copy*. After the file has been copied to the target, the program is executed on the target using *SSH*.

You will see the following content in the *Console* window:



```

C/C++ - HelloWorld/HelloWorld.c - Eclipse SDK
File Edit Source Refactor Navigate Search Run Project Window Help

Project Exp
  HelloWorld
    Binaries
    Includes
    Debug
    HelloWorld.c
    HelloWorld

HelloWorld.c
#include <stdio.h>

int main(void)
{
    printf("Welcome to the World of the phyFLEX-i-MX6!\n");
    return 0;
}

Problems Tasks Console Properties
C-Build [HelloWorld]
Building file: ../HelloWorld.c
Invoking: GCC C Compiler
arm-cortexa9-linux-gnueabi-gcc -O0 -g3 -Wall -c -fmessage-length=0 -MMD -MP -MF"HelloWorld.d" -
MT"HelloWorld.d" -o"HelloWorld.o" "../HelloWorld.c"
Finished building: ../HelloWorld.c

Building target: HelloWorld
Invoking: GCC C Linker
arm-cortexa9-linux-gnueabi-gcc -o"HelloWorld" ../HelloWorld.o
Finished building target: HelloWorld

make --no-print-directory post-build
scp ./HelloWorld root@192.168.3.11:/home/. ;ssh root@192.168.3.11 ./HelloWorld
Welcome to the World of the phy
!
  
```



If the project is not built automatically, you will have to check *Project* ► *Build automatically* from the menu bar. To build it new select *Project* ► *Clean...* .



You have successfully passed the first steps with the Eclipse IDE. You are now able to import existing projects into the Eclipse workspace. You can compile an existing project and execute the program on the target.

3.1.2 Creating a New Project

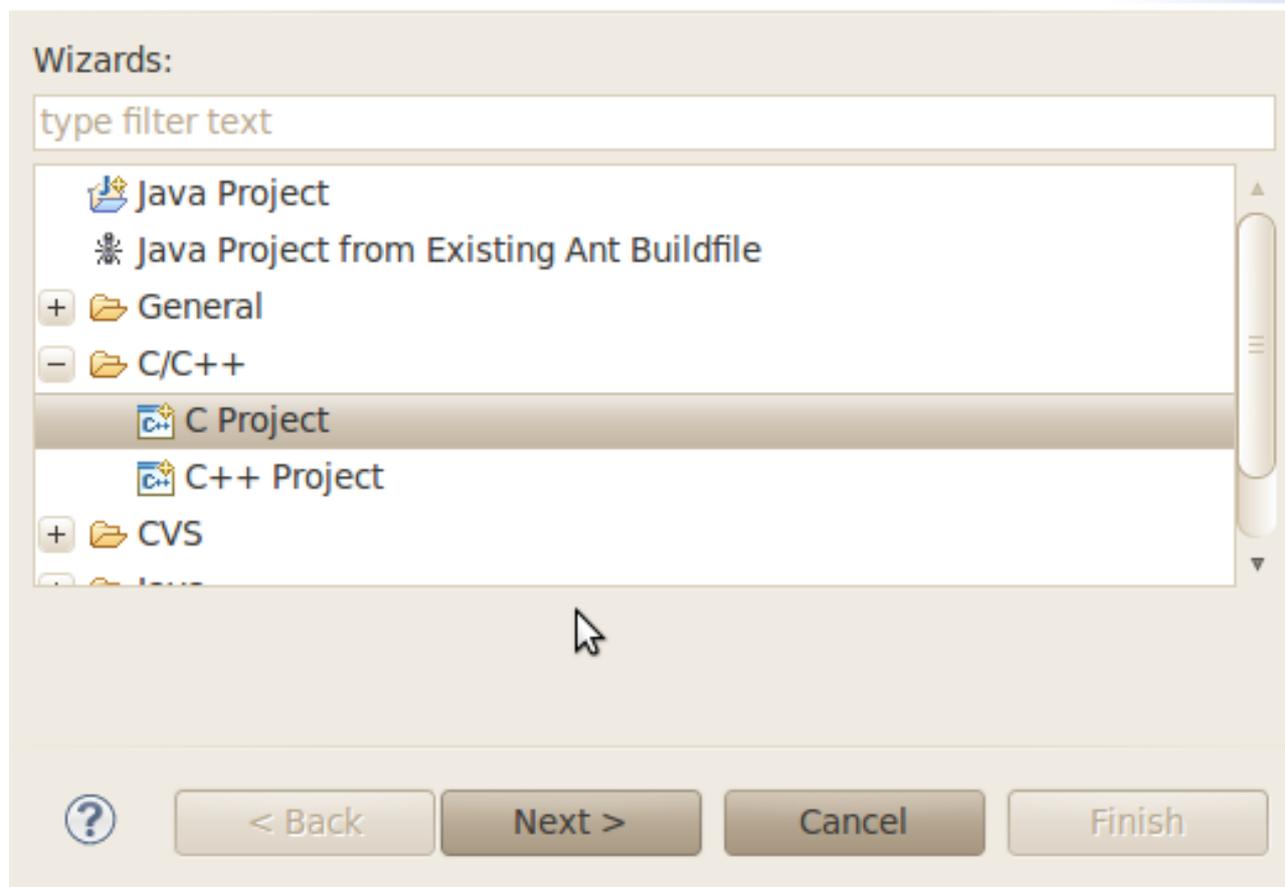
In this section you will learn how to create a new project with Eclipse and how to configure the project for use with the GNU C/C++ cross development toolchain.

- Open Eclipse if it isn't already opened
- Select *File* ► *New* ► *Project* from the menu bar

A new dialog opens.

Select a wizard

Create a new C project



- Select *C Project* and click *Next*

C Project

Create C project of selected type



Project name:

Use default location

Location:

Project type:

- Executable
- Empty Project
- Hello World ANSI C Project
- + Shared Library

Toolchains:

- Linux GCC

Show project types and toolchains only if they are supported on the platform

- Enter the project name *myHelloWorld* and click *Next*

Select Configurations

Select platforms and configurations you wish to deploy on



Project type: Executable
Toolchains: Linux GCC
Configurations:

 Debug
  Release

Select all

Deselect all

Advanced settings...

Use "Advanced settings" button to edit project's properties.

Additional configurations can be added after project creation.

Use "Manage configurations" buttons either on toolbar or on property pages.



< Back

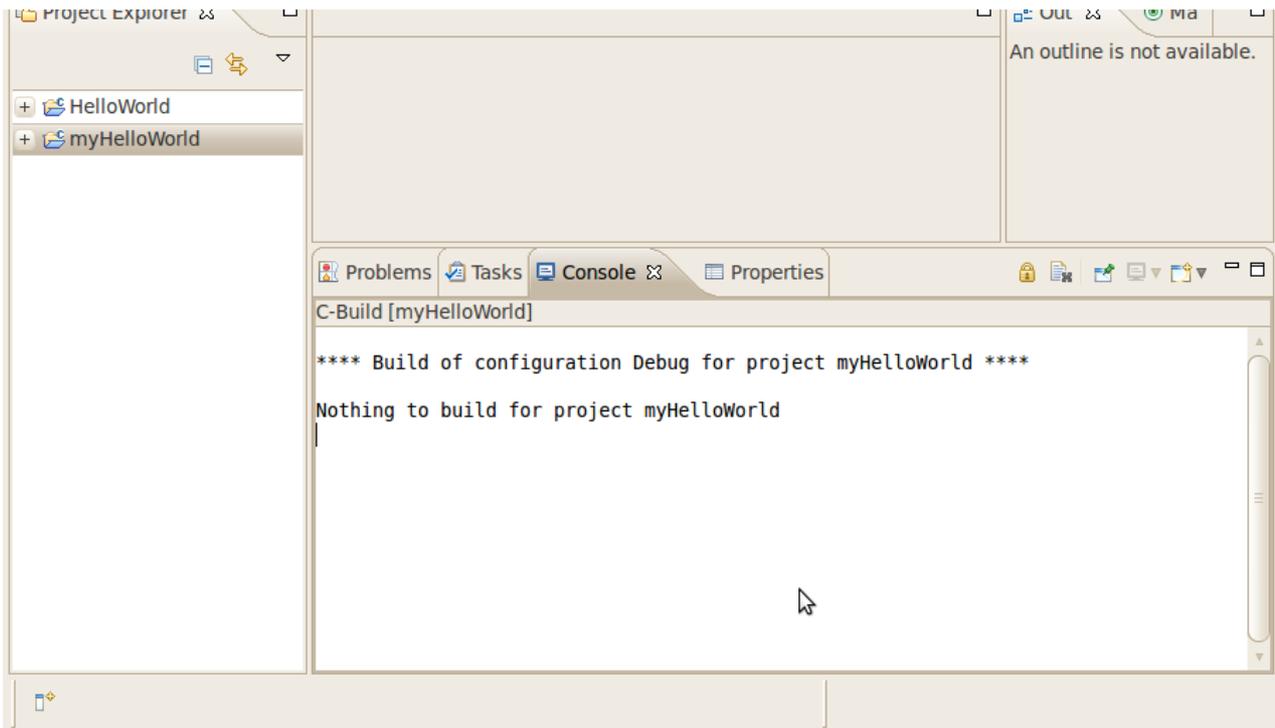
Next >

Cancel

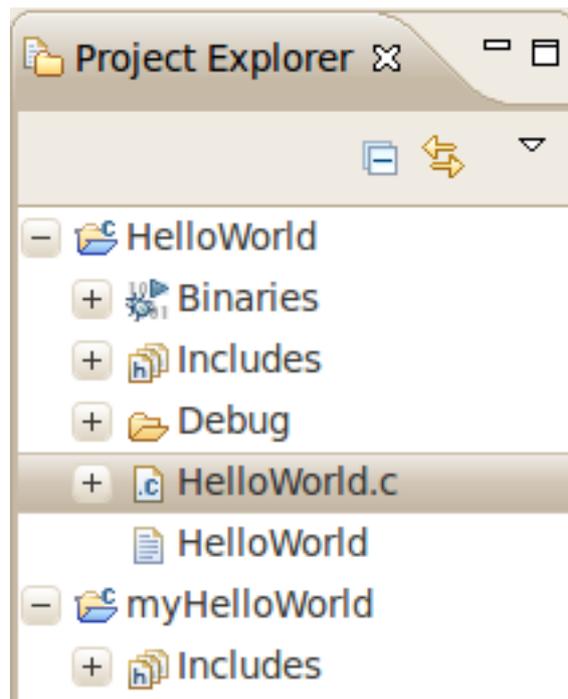
Finish

- Click *Finish*

You will see the C/C++ IDE with the *myHelloWorld* project.



- Double-Click the *HelloWorld* project which we have worked with previously
- Right-click on *HelloWorld.c* in the *HelloWorld* project
- Select *Copy*



- Select the *myHelloWorld* project
- Right-click the *myHelloWorld* project
- Select *Paste*
- Double-click on *HelloWorld.c* in the *myHelloWorld* project

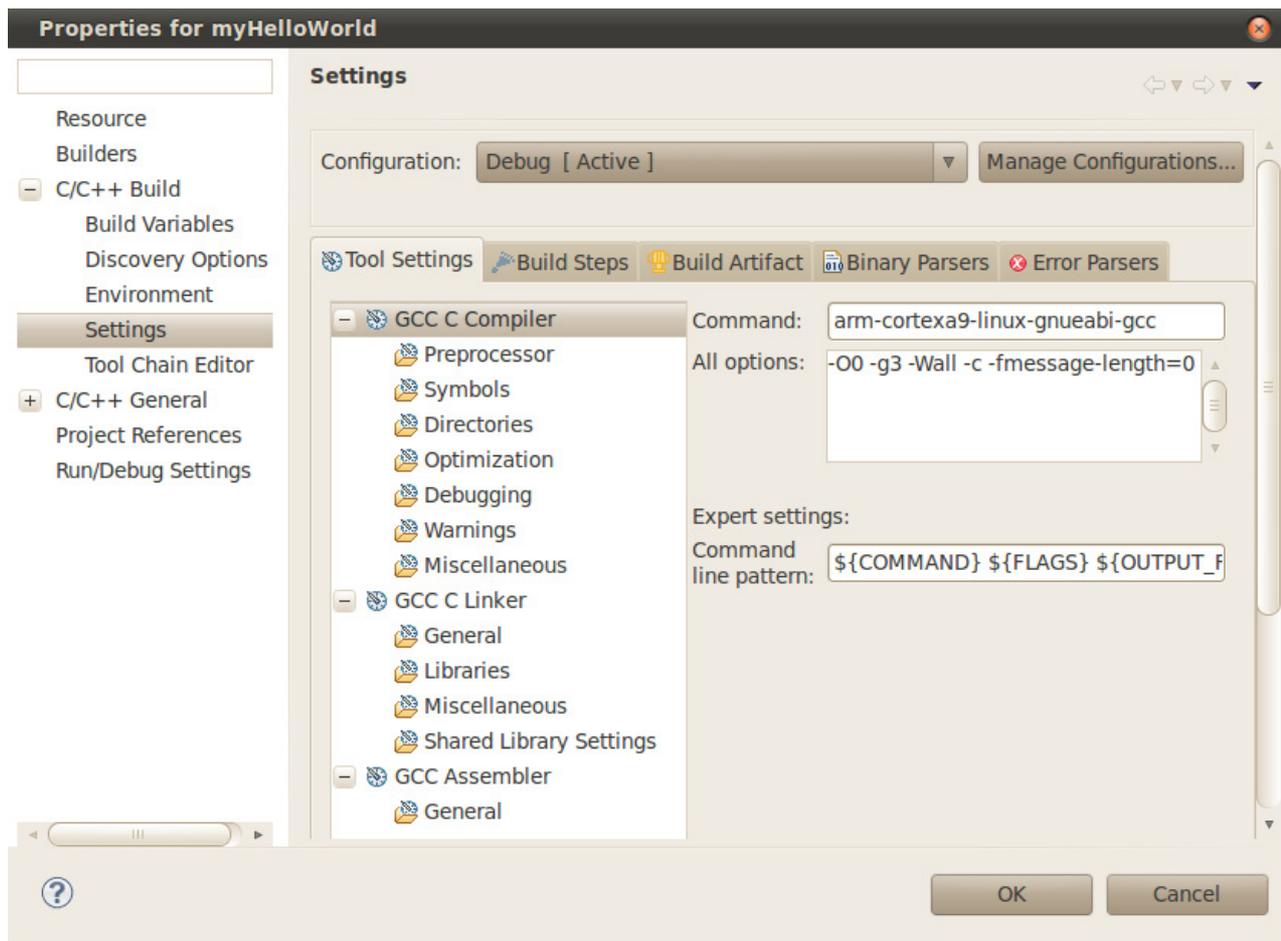
If *Build Automatically* from the *Project* menu is selected, the *HelloWorld* application will now be compiled and created with the standard GCC C/C++ compiler suitable for your host machine. You will find the executable file, which can only run on your host system, in the *workspace/myHelloWorld/Debug* directory.

To compile your project for the phyFLEX-i.MX6 instead, you will have to use the GNU C/C++ cross compiler.

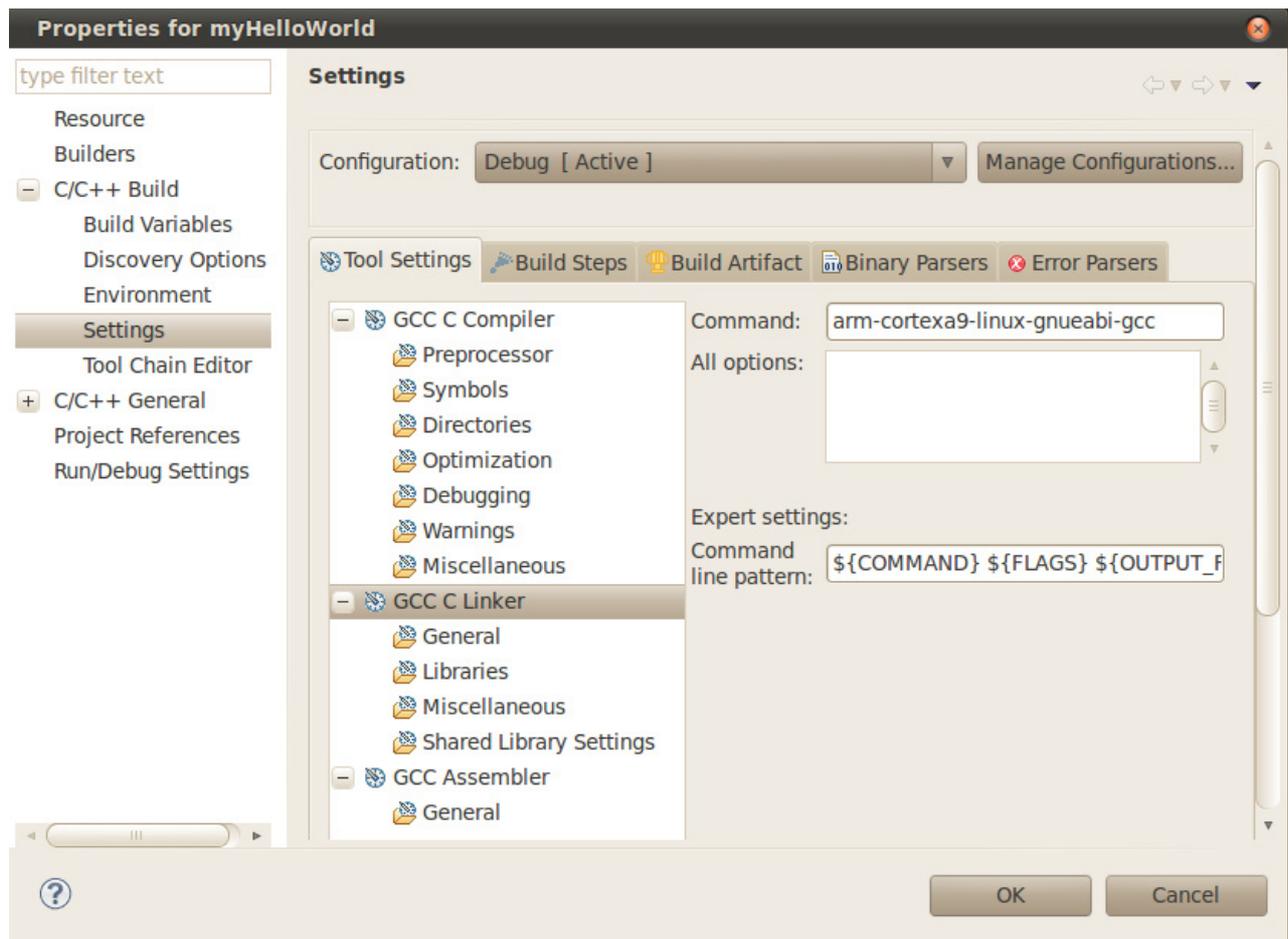
- Right-click the *myHelloWorld* project and choose *Properties*

The *Properties* dialog appears.

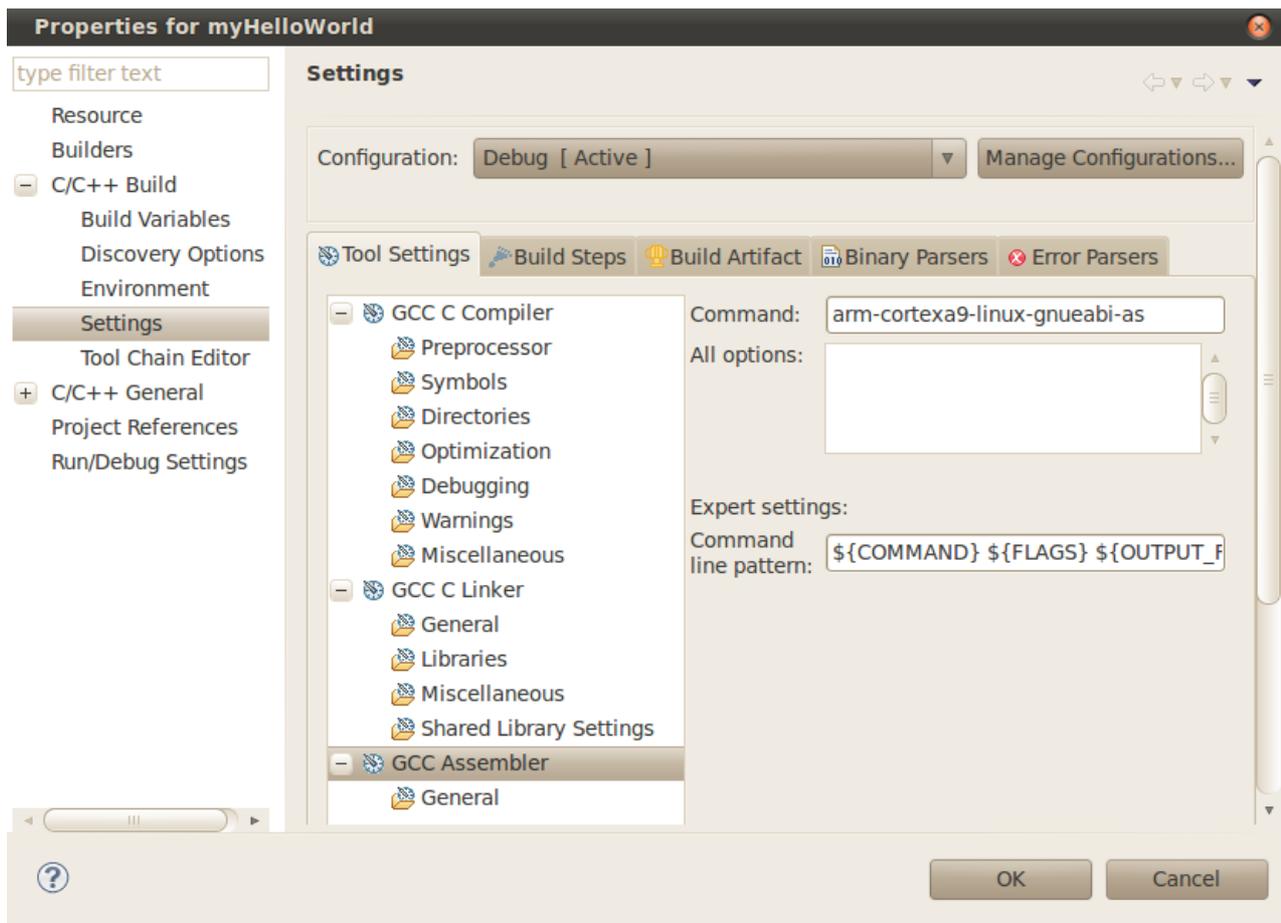
- Select *C/C++ Build*
- Enter **arm-cortexa9-linux-gnueabi-gcc** into the *Command* input field



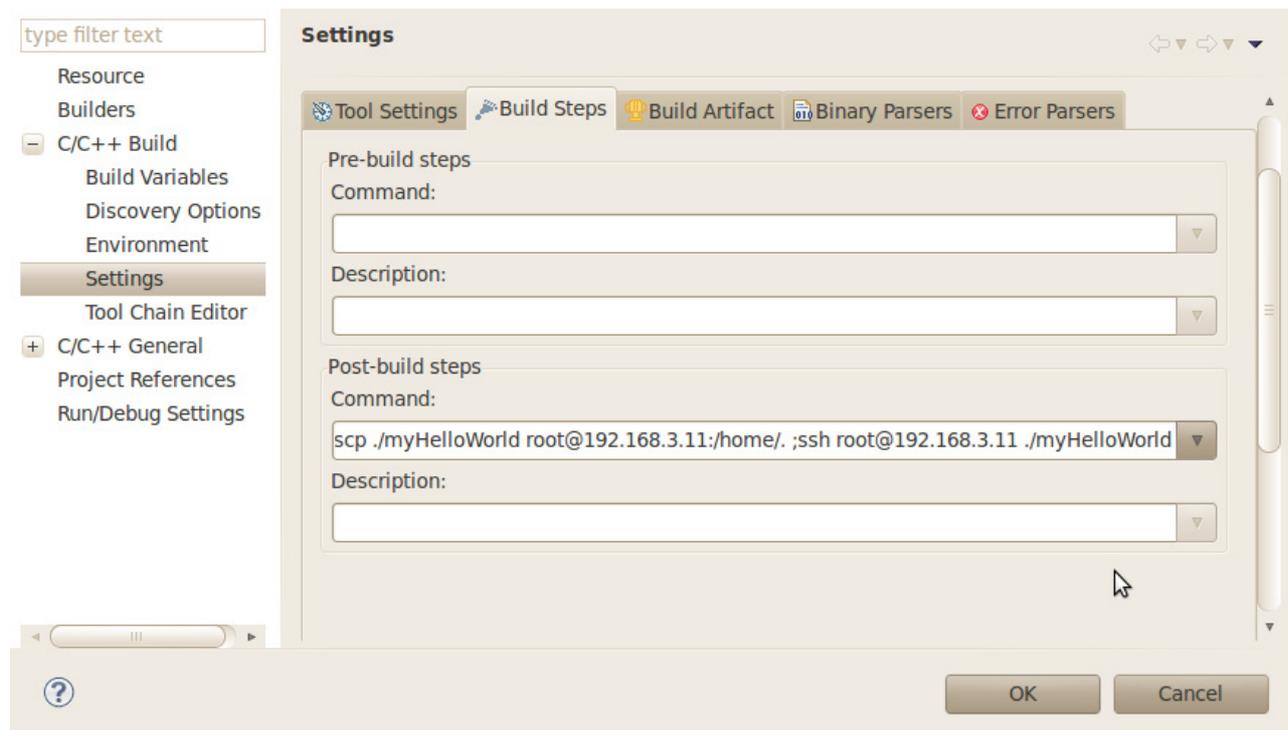
- Select *GCC C Linker*
- Enter **arm-cortexa9-linux-gnueabi-gcc** into the *Command* input field



- Select *GCC Assembler*
- In the *Command* input field, change the default **as** to **arm-cortexa9-linux-gnueabi-as**
- Click *Apply*

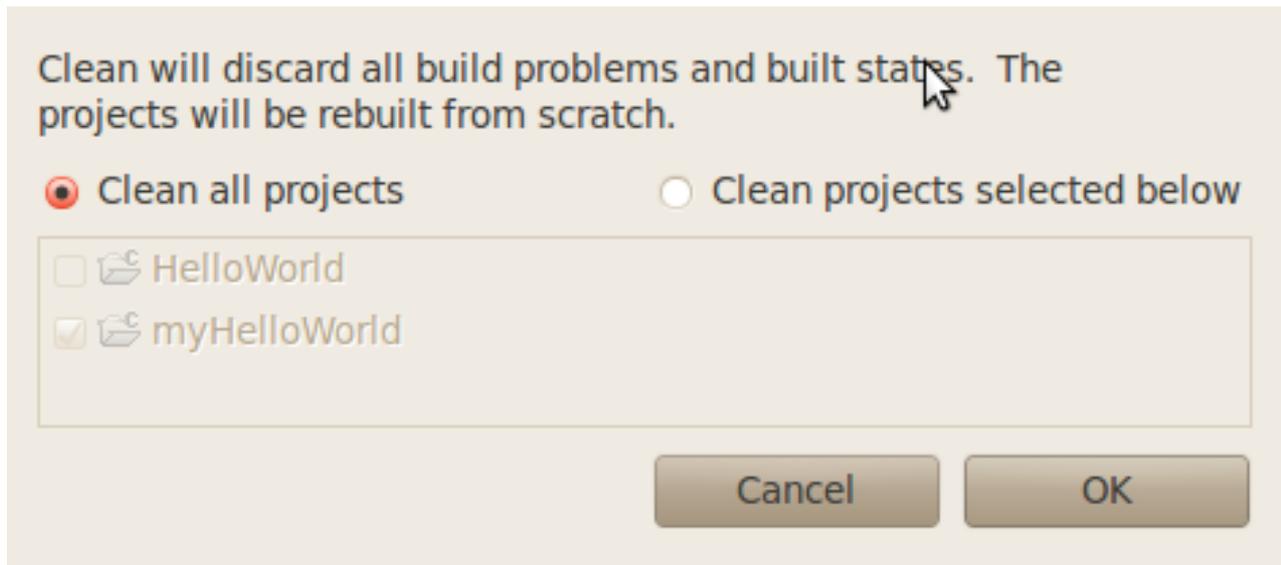


- Select the *Build Steps* tab
- Enter the following command in the Post-build steps *Command* input field:
scp ./myHelloWorld root@192.168.3.11:/home/. ;ssh root@192.168.3.11 ./myHelloWorld



Be sure to enter the semicolon before the ssh command.
Be sure that the file *myHelloWorld* on the target will have execution rights, because otherwise *ssh* will fail.

- Click *Apply*
- Click *OK*
- Select *Project* ► *Clean* from the menu bar



- Confirm with *OK*.
The project will be rebuilt.
- Select the *Console* tab.
If no errors occur while building the project, you will see the following output:

```
C-Build [myHelloWorld]

**** Build of configuration Debug for project myHelloWorld ****

make all
Building file: ../HelloWorld.c
Invoking: GCC C Compiler
arm-cortexa9-linux-gnueabi-gcc -O0 -g3 -Wall -c -fmessage-length=0 -MMD -MP -MF"HelloWorld.d" -
MT"HelloWorld.d" -o"HelloWorld.o" "../HelloWorld.c"
Finished building: ../HelloWorld.c

Building target: myHelloWorld
Invoking: GCC C Linker
arm-cortexa9-linux-gnueabi-gcc -o"myHelloWorld" ./HelloWorld.o
Finished building target: myHelloWorld

make --no-print-directory post-build
scp ./myHelloWorld root@192.168.3.11:/home/. ;ssh root@192.168.3.11 ./myHelloWorld
Welcome to the World of the phy      !
```



You have successfully created your first own project with the Eclipse IDE. You have configured the project to create an application for your target platform.

3.1.3 Changing the Demo Application

Now we will extend the *myHelloWorld* application. The extended *myHelloWorld* application will write an output to the first serial interface as well as to the standard output.

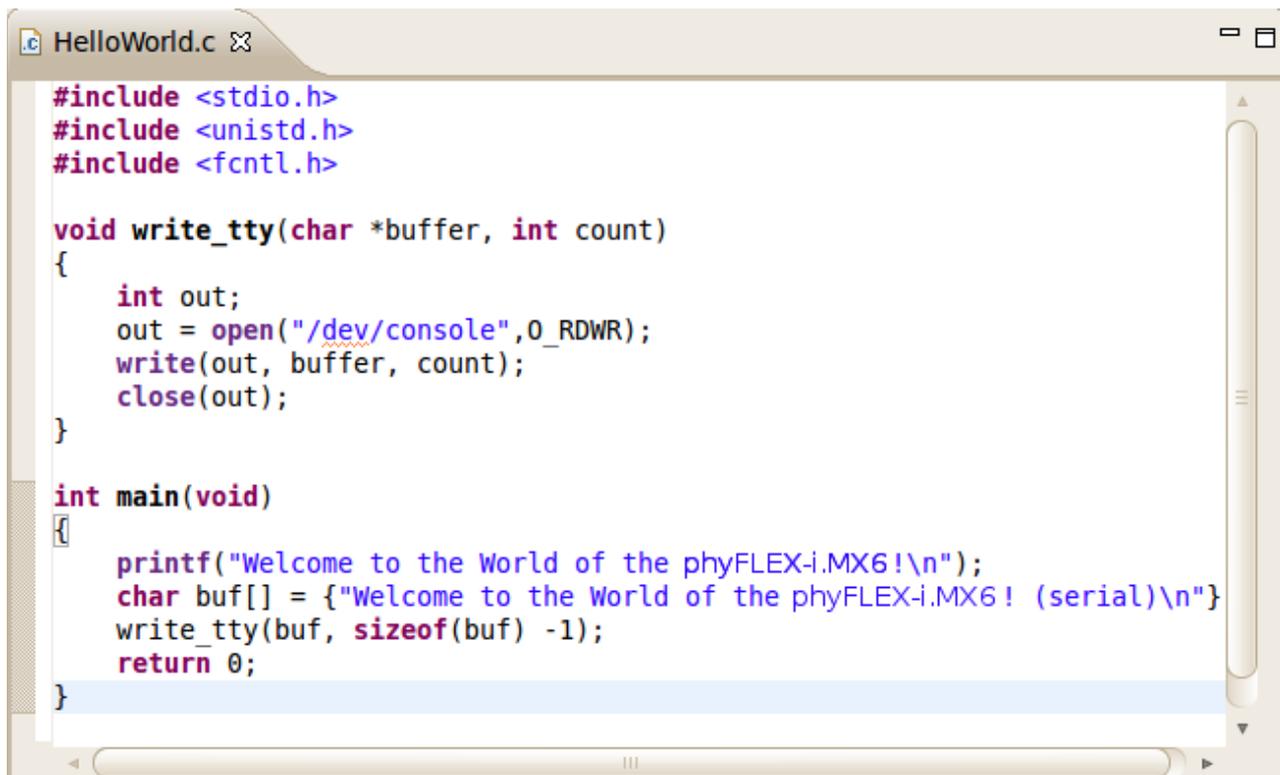
- Open Eclipse if it is not opened yet
- Double-click *HelloWorld.c* in the *myHelloWorld* project
- First include the following two additional header files:

```
#include <unistd.h>
#include <fcntl.h>
```
- Then add the function *write_tty()*, which writes *n* bytes to the first serial interface (which, on the phyFLEX-i.MX6, is connected to the system console */dev/console*):

```
void write_tty (char *buffer, int count) {
    int out;
    out = open ("/dev/console", O_RDWR);
    write(out, buffer, count);
    close(out);
}
```
- Enter the following two lines in the *main()* function to declare the buffer and call the *write_tty()* function.

```
char buf [] = { "Welcome to the World of the phyFLEX-i.MX6! (serial)\n" };
write_tty(buf, sizeof (buf) - 1);
```

In the next screenshot you can see the complete program.



```

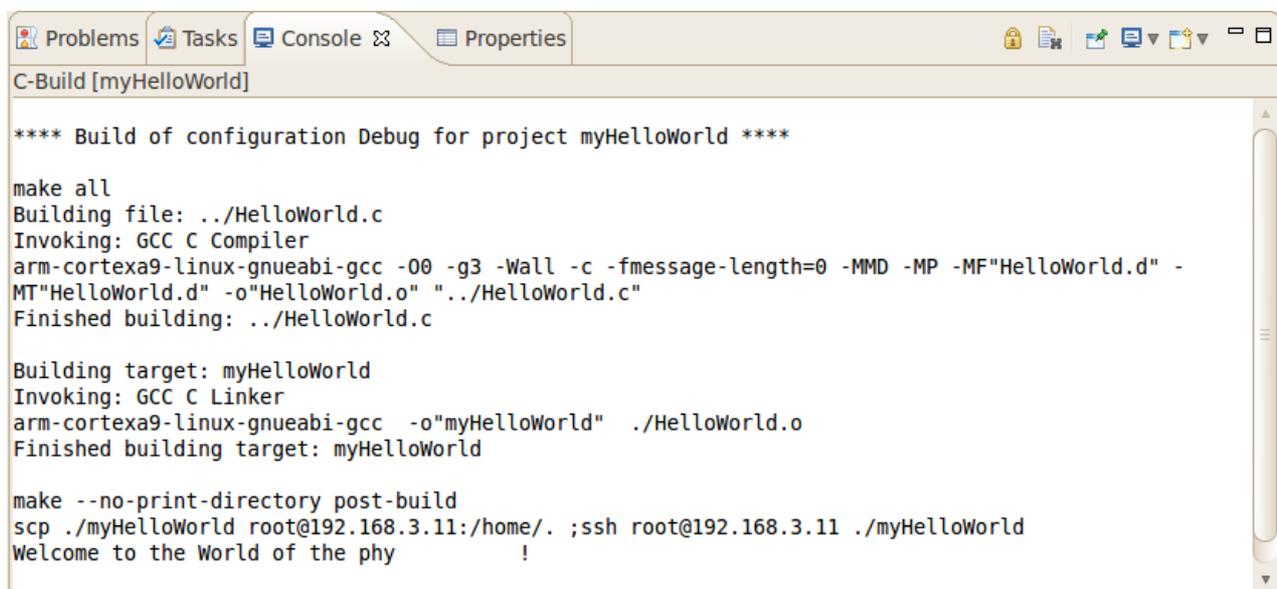
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>

void write_tty(char *buffer, int count)
{
    int out;
    out = open("/dev/console", O_RDWR);
    write(out, buffer, count);
    close(out);
}

int main(void)
{
    printf("Welcome to the World of the phyFLEX-i.MX6!\n");
    char buf[] = {"Welcome to the World of the phyFLEX-i.MX6! (serial)\n"};
    write_tty(buf, sizeof(buf) - 1);
    return 0;
}

```

- Save your program after changing the code.
The application will be compiled, built, copied to the target and executed.



```

C-Build [myHelloWorld]

**** Build of configuration Debug for project myHelloWorld ****

make all
Building file: ../HelloWorld.c
Invoking: GCC C Compiler
arm-cortexa9-linux-gnueabi-gcc -O0 -g3 -Wall -c -fmessage-length=0 -MMD -MP -MF"HelloWorld.d" -
MT"HelloWorld.d" -o"HelloWorld.o" "../HelloWorld.c"
Finished building: ../HelloWorld.c

Building target: myHelloWorld
Invoking: GCC C Linker
arm-cortexa9-linux-gnueabi-gcc -o"myHelloWorld" ../HelloWorld.o
Finished building target: myHelloWorld

make --no-print-directory post-build
scp ./myHelloWorld root@192.168.3.11:/home/. ;ssh root@192.168.3.11 ./myHelloWorld
Welcome to the World of the phy      !

```

- Click the *Microcom* icon on the desktop
- If you are not logged in, enter **root** and press *Enter*
- Type `./myHelloWorld` to start the application
- You will see the following output:
Welcome to the World of the phyFLEX-i.MX6! (serial)
Welcome to the World of the phyFLEX-i.MX6!
- Close Microcom

When you start the application via an SSH session, you only see one output line. When you execute the program with Microcom, you see two output lines.



The first line is a direct output on the serial interface. You can't see this line in a SSH session, because you are connected over a TCP/IP connection to the target. With Microcom, however, you have direct access to the serial interface, so you can also see the line that is written to the serial console.

In this section you have changed an existing application. You also learned how to access the serial interface. First you called the function `open()` on the device `/dev/console`. The return value of this function was a file descriptor. With the file descriptor you called the function `write()` to send `n` bytes to the device `/dev/console`. After that, the file descriptor was closed with the function `close()`.

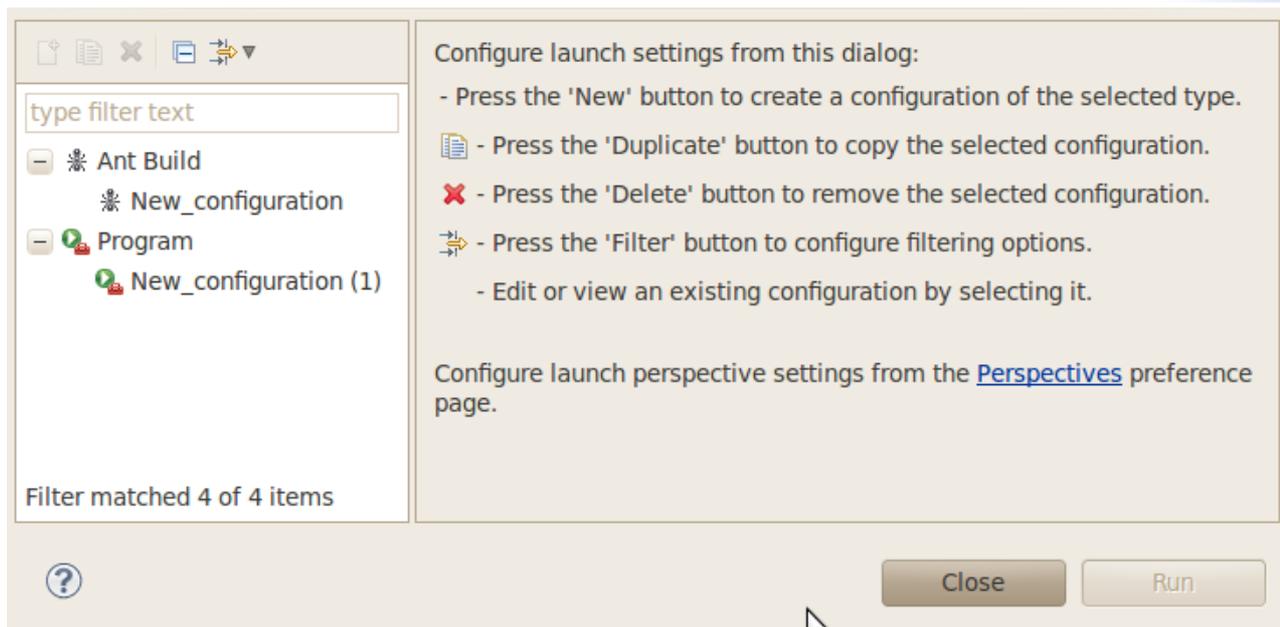
This procedure is in principle quite typical for Linux, because Linux treats everything like a file.

3.1.4 Starting a Program out of Eclipse on the Target

After compiling a project in Eclipse, the program is copied to the target and directly executed. A program can also be executed on the target without compiling a project. In the following section you will learn how to start a program on the target as an external tool.

- Select *Run* ► *External Tools* ► *External Tools Configurations* from the menu bar

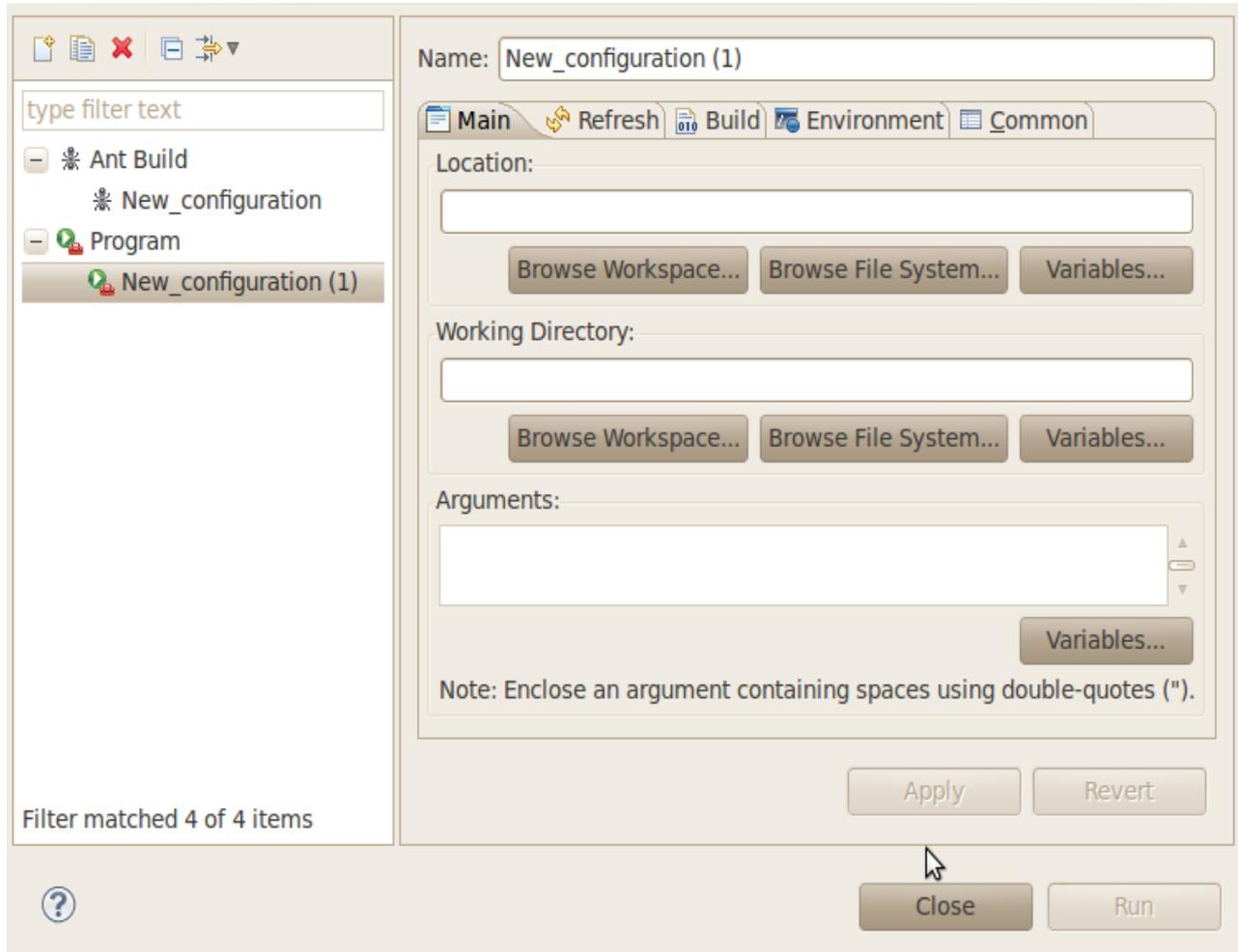
Create, manage, and run configurations



- Under *Program* select *New_configuration (1)*

Create, manage, and run configurations

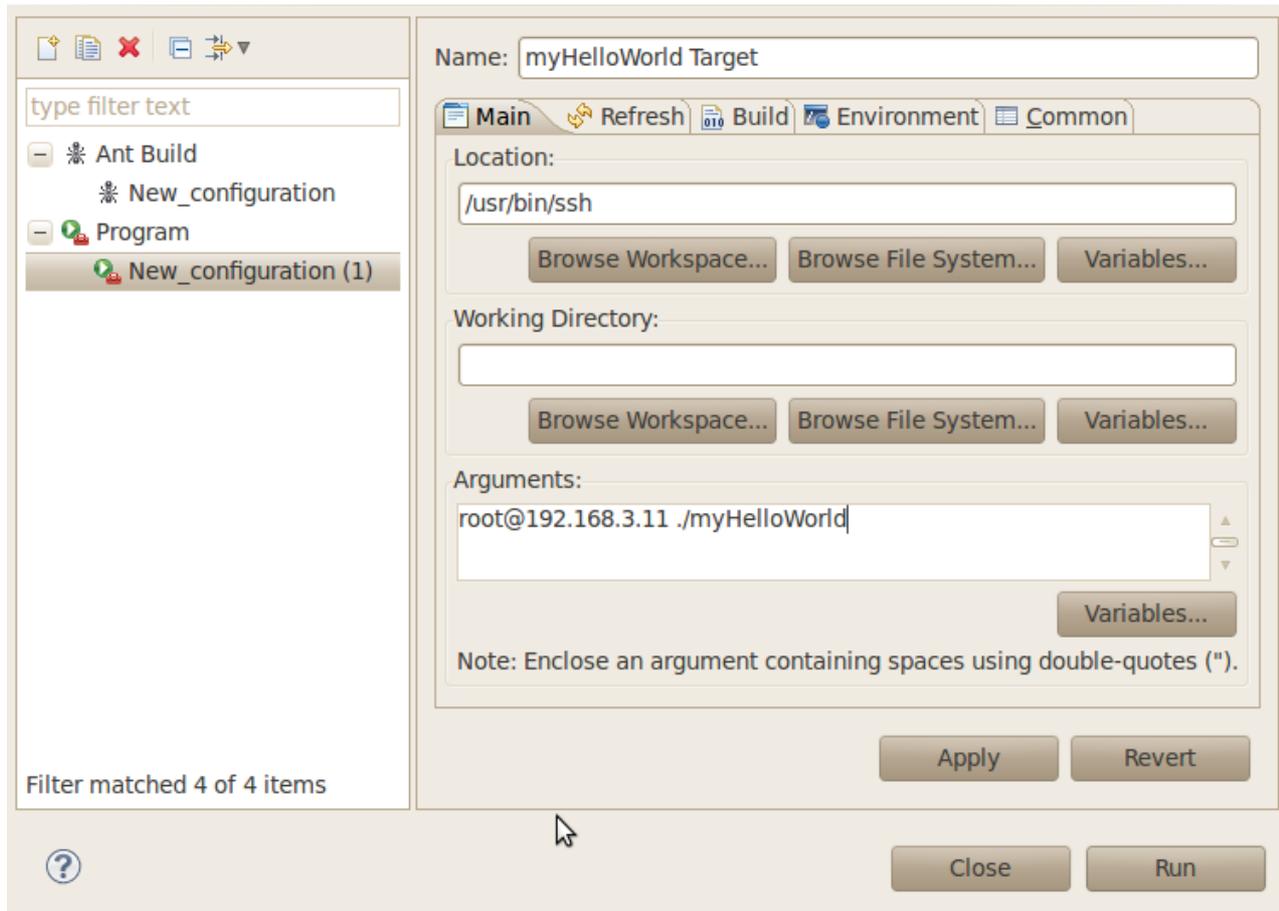
Please specify the location of the external tool you would like to configure.



- In the *Name* input field, enter: **myHelloWorld Target**
- Enter **/usr/bin/ssh** in the *Location* input field
- Enter **root@192.168.3.11 ./myHelloWorld** into the *Arguments* field
- Select *Apply*

Create, manage, and run configurations

Run a program



- Select *Run*

If you want to execute the program the next time, you can use the *Run External Programs*



button from the menu bar.



You have successfully create your own Eclipse project and you learned how to execute a program at the target.

3.2 Programming in the Qt C++ perspective

In this section our attention goes to the Qt framework, which gives us tools to develop graphical user interfaces. With the help of an example project we will give you a short introduction of how to work with Qt.

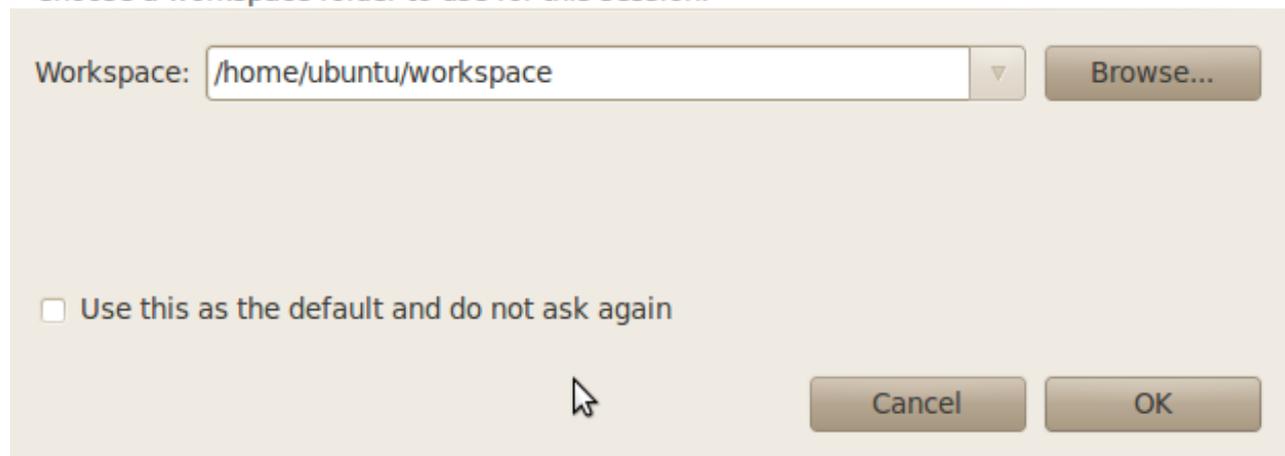
3.2.1 Importing the demo application



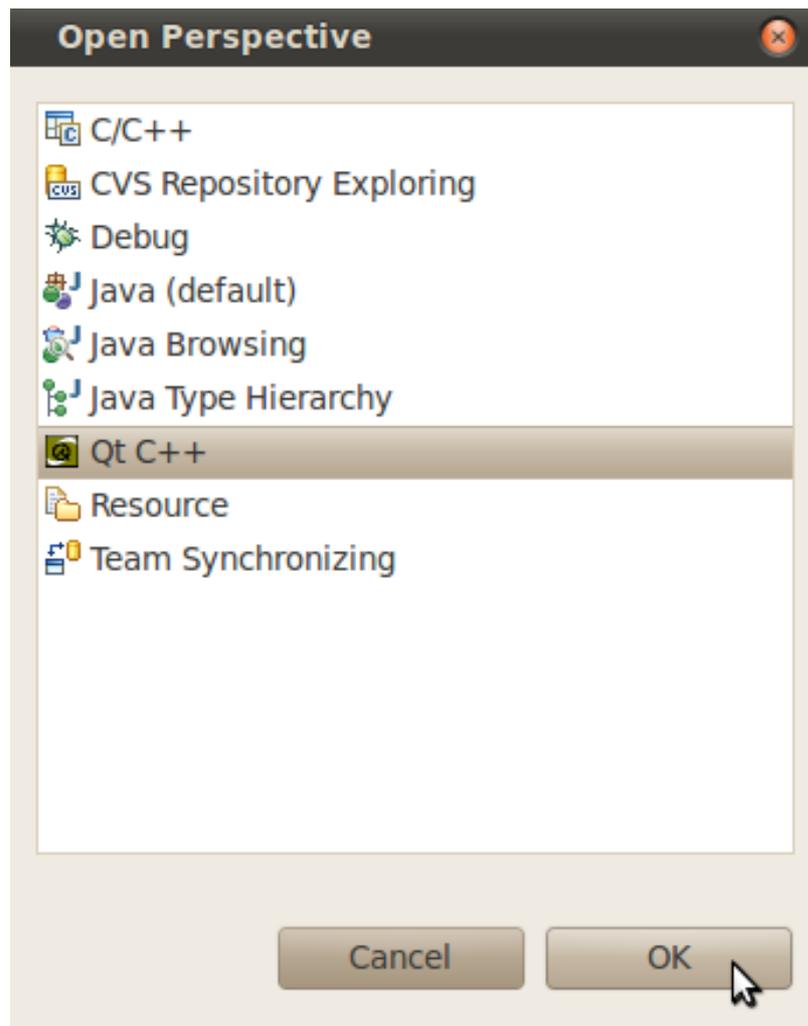
- Click the Eclipse icon to start the application, if it isn't already open. You can find this icon on your desktop.

Select a workspace

Eclipse SDK stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

A screenshot of the Eclipse workspace selection dialog. The dialog has a title bar and a main area. At the top, it says "Workspace:" followed by a text box containing "/home/ubuntu/workspace" and a "Browse..." button. Below this, there is a checkbox labeled "Use this as the default and do not ask again". At the bottom, there are "Cancel" and "OK" buttons. A mouse cursor is visible over the "Cancel" button.

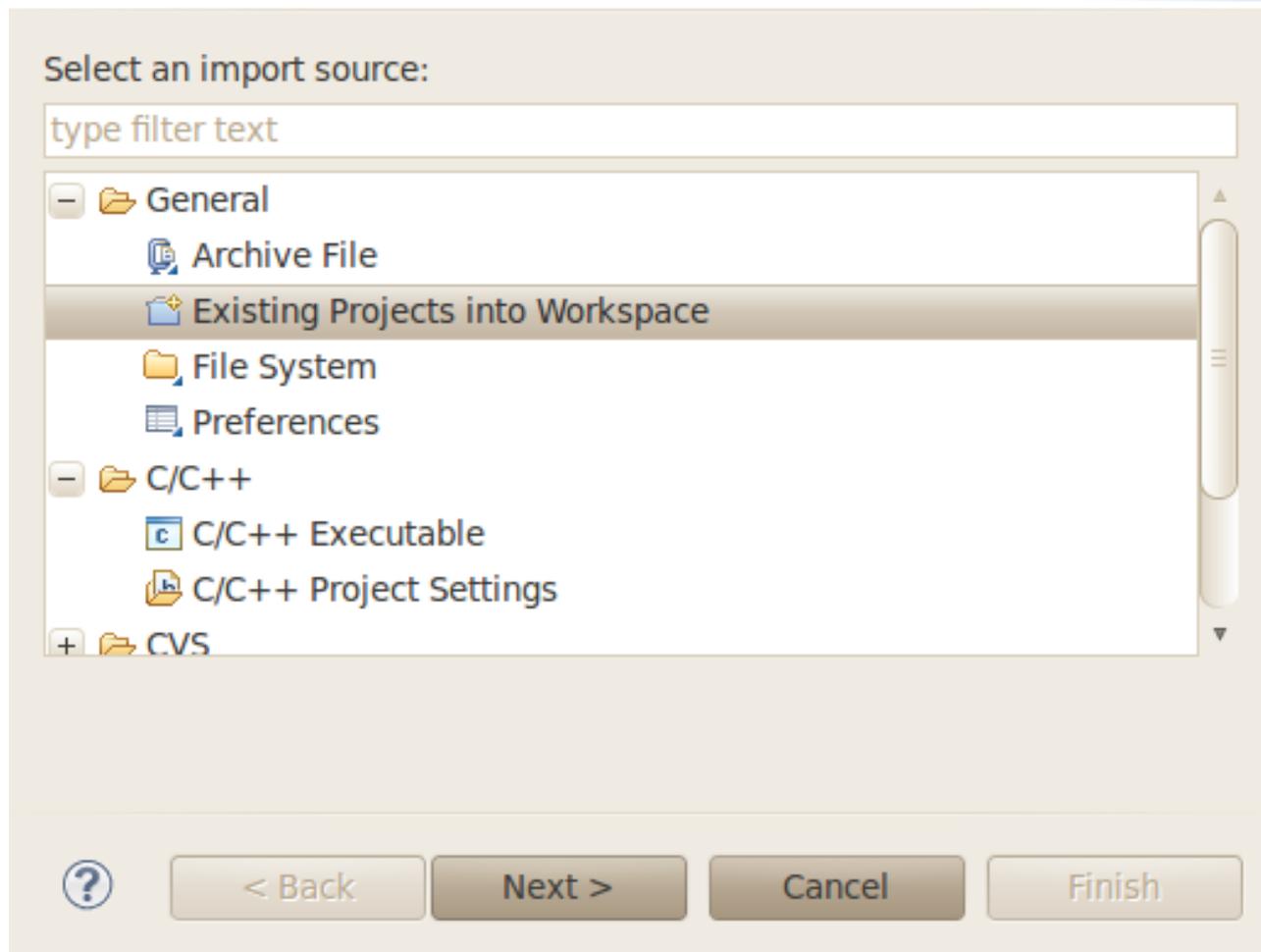
- Now we switch to the Qt C++ perspective. Click *Window* ► *Open Perspective* ► *Other*.



- A dialog opens. Choose *Qt C++* and click **OK**.
- Now we can import the example project. Select *File* ► *Import* from the menu bar.

Select

Create new projects from an archive file or directory.



- Select *Existing Projects into Workspace*
- Click *Next*

Import Projects



Select a directory to search for existing Eclipse projects.

Select root directory:

Select archive file:

Projects:

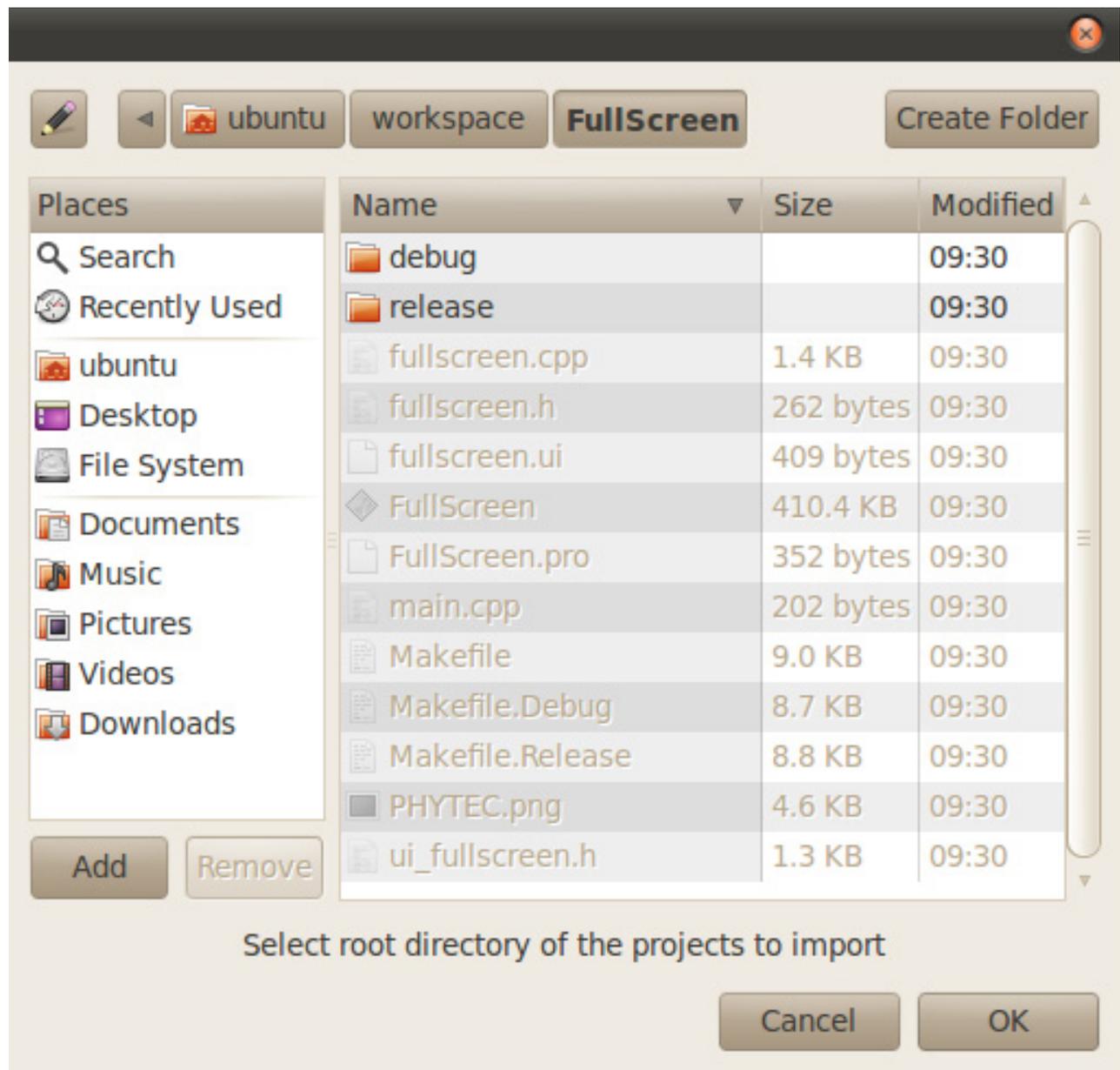
Copy projects into workspace

Working sets

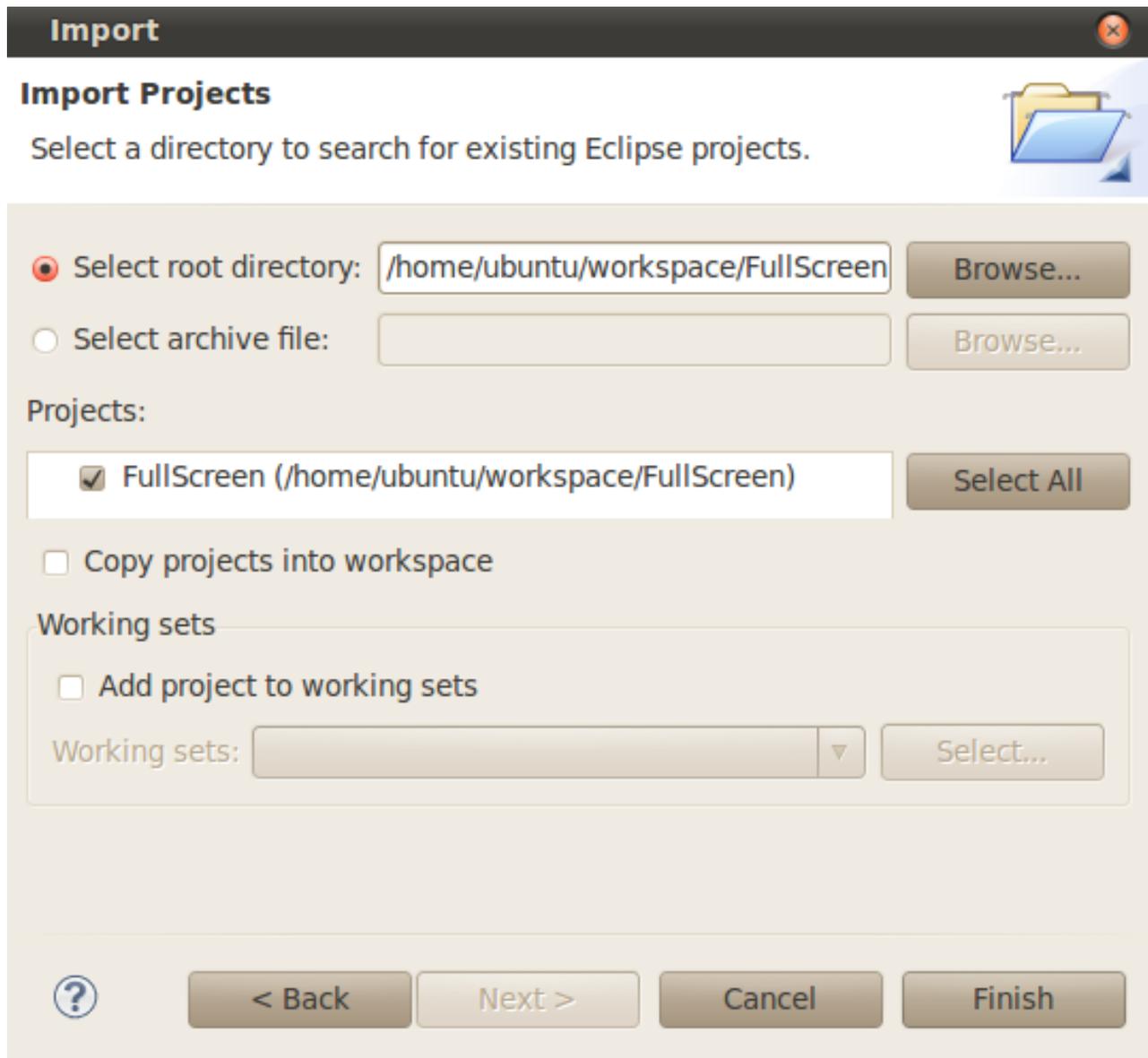
Add project to working sets

Working sets:

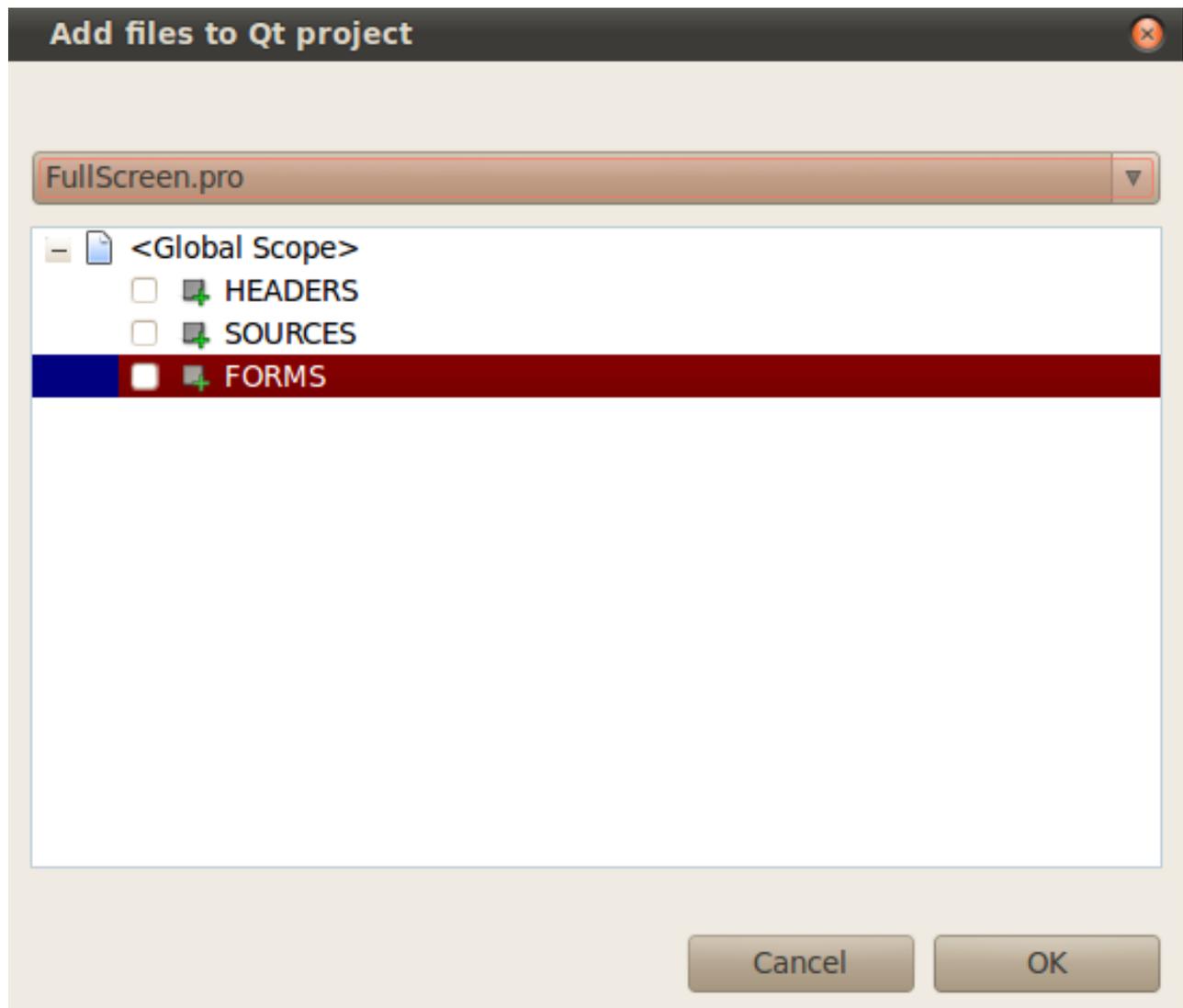
- Select *Browse*
- Double-click the *FullScreen* directory under */home/ubuntu/workspace/*



- Click *OK*



- Select *Finish* to import the project
- On the next window deselect *HEADERS* and *FORMS* and click *OK*



The *FullScreen* program will be compiled and you can find the outputs on the console. After the compilation is finished successfully the *FullScreen* executable for the target is built and can be found under */home/ubuntu/workspace/FullScreen*.

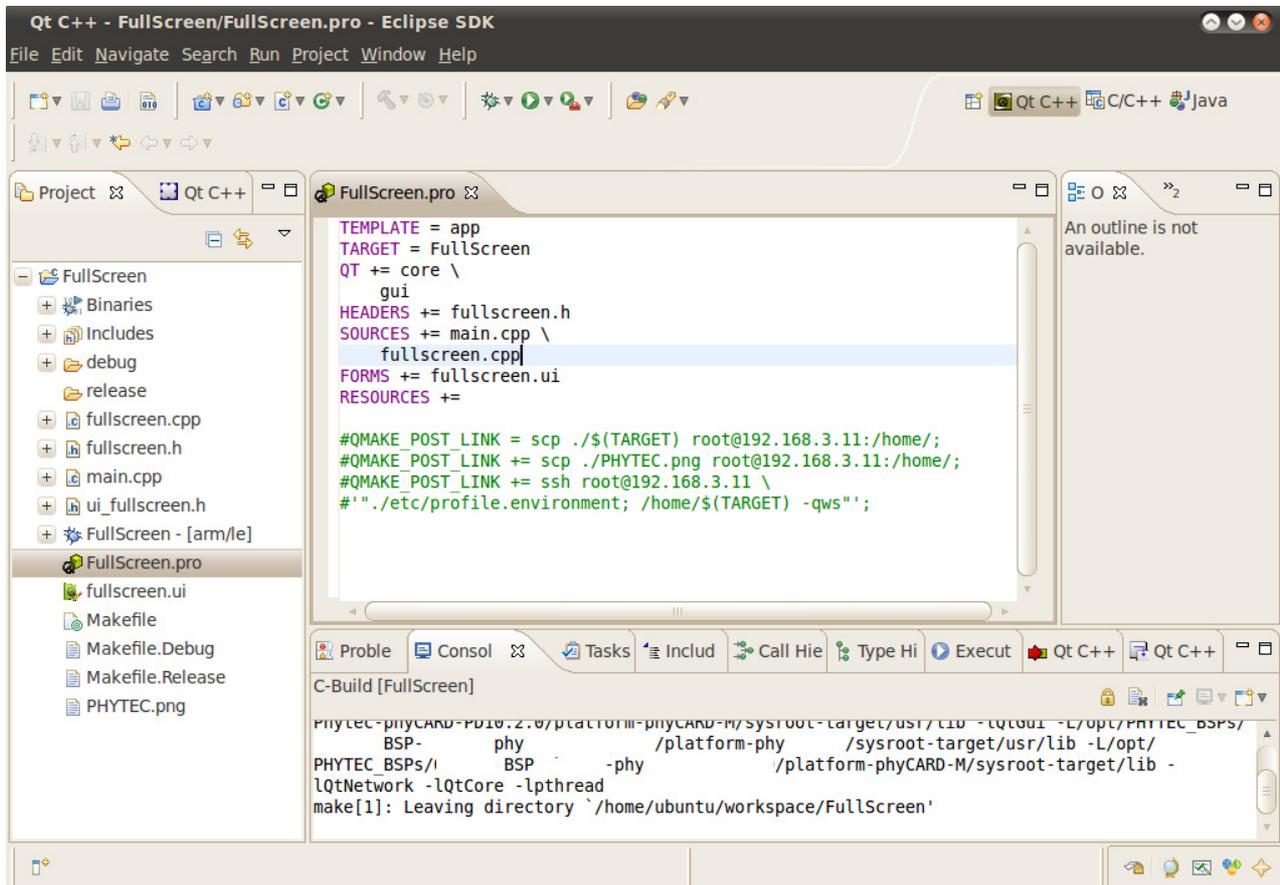


If the project is not built automatically, you will have to check *Project ► Build automatically* in the menu bar. To rebuild select *Project ► Clean... .*

3.2.2 Handle with the demo application

If you want the project to be automatically copied to the target and executed we must make some changes in the *FullScreen.pro* file.

- Double-click the *FullScreen.pro* to open it

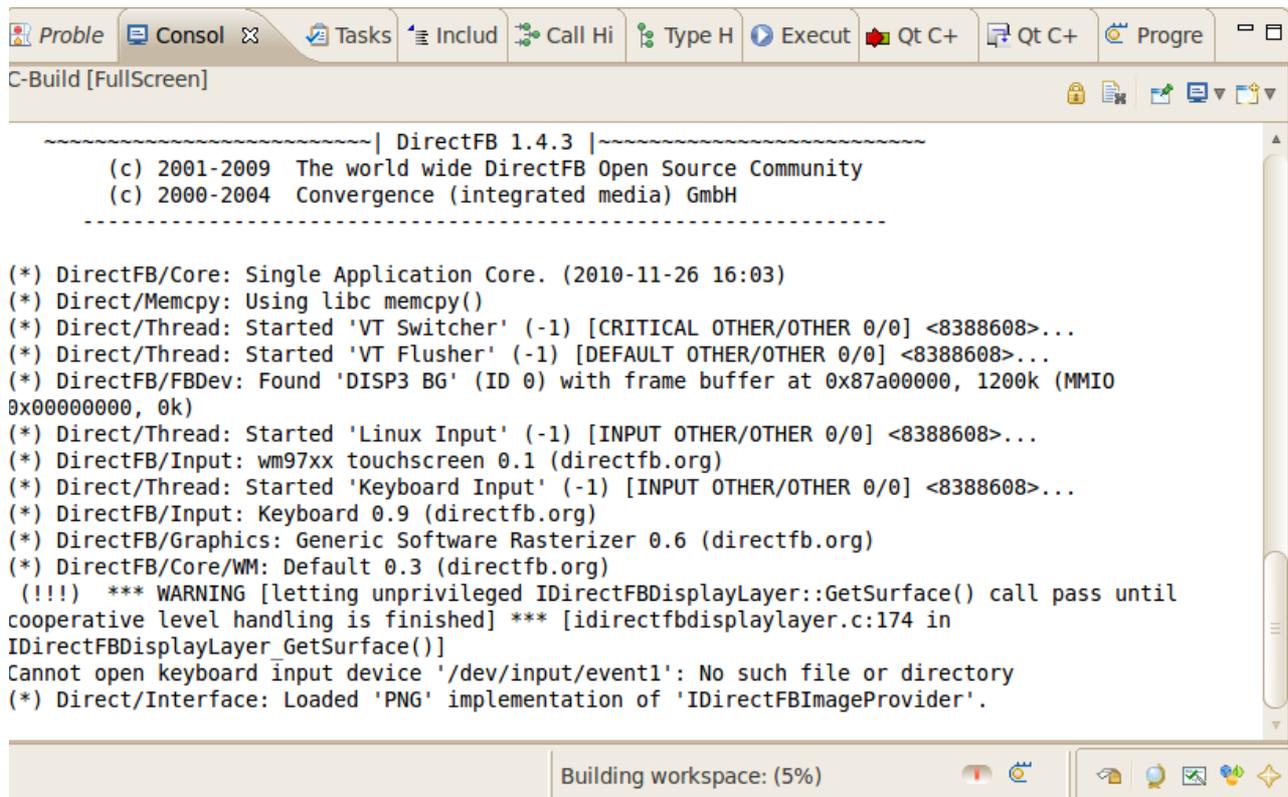


In this file you will find four uncommented rows.

- Remove the comment signs (#) to enable the three `QMAKE_POST_LINK` tags. With this tag you can add post-build commands. In our case we copy the compiled project and the picture to the target and connect via `ssh`, set the environment and execute the application.
- Select *Project* ► *Clean...*
- Confirm the Clean dialog with *OK*

After the *FullScreen* program is compiled the `QMAKE_POST_LINK` is called. The *FullScreen* file is copied to the target using secure copy and executed using *SSH*.

You will see the following content in the *Console* window:



```

-----| DirectFB 1.4.3 |-----
      (c) 2001-2009 The world wide DirectFB Open Source Community
      (c) 2000-2004 Convergence (integrated media) GmbH
-----

(*) DirectFB/Core: Single Application Core. (2010-11-26 16:03)
(*) Direct/Memcpy: Using libc memcpy()
(*) Direct/Thread: Started 'VT Switcher' (-1) [CRITICAL OTHER/OTHER 0/0] <8388608>...
(*) Direct/Thread: Started 'VT Flusher' (-1) [DEFAULT OTHER/OTHER 0/0] <8388608>...
(*) DirectFB/FBDev: Found 'DISP3 BG' (ID 0) with frame buffer at 0x87a00000, 1200k (MMIO
0x00000000, 0k)
(*) Direct/Thread: Started 'Linux Input' (-1) [INPUT OTHER/OTHER 0/0] <8388608>...
(*) DirectFB/Input: wm97xx touchscreen 0.1 (directfb.org)
(*) Direct/Thread: Started 'Keyboard Input' (-1) [INPUT OTHER/OTHER 0/0] <8388608>...
(*) DirectFB/Input: Keyboard 0.9 (directfb.org)
(*) DirectFB/Graphics: Generic Software Rasterizer 0.6 (directfb.org)
(*) DirectFB/Core/WM: Default 0.3 (directfb.org)
(!!!) *** WARNING [letting unprivileged IDirectFBDisplayLayer::GetSurface() call pass until
cooperative level handling is finished] *** [idirectfbdisplaylayer.c:174 in
IDirectFBDisplayLayer_GetSurface()]
Cannot open keyboard input device '/dev/input/event1': No such file or directory
(*) Direct/Interface: Loaded 'PNG' implementation of 'IDirectFBImageProvider'.
  
```

Building workspace: (5%)

On the display of the target the project is started and you can change between windowed and fullscreen mode by clicking the button.



*You have successfully imported and built a Qt project in Eclipse.
You've also learned to run your application on the target.*

4 Debugging an example project

**35 min**

In this chapter you will learn using the GNU debugger GDB on the host for remote debugging in conjunction with the GDB server on the target. GDB is the symbolic debugger of the GNU project and is arguably the most important debugging tool for any Linux system.

First you will start the GDB server on the target. Then you will configure the Eclipse platform and start the GNU debugger out of Eclipse using the Debug view.

The CDT extends the standard Eclipse Debug view with functions for debugging C/C++ code. The Debug view allows you to manage the debugging and running of a program in the workbench. Using the Debug view you will be able to set breakpoints/watchpoints in the code and trace variables and registers. The Debug view displays the stack frame for the threads of each target you are debugging. Each thread in your program appears as a node in the tree, and the Debug view displays the process for each target you are running.

The GDB client is running on the host and is used to control the GDB server on the target, which in turn controls the application running on the target. GDB client and GDB server can communicate over a TCP/IP network connection as well as via a serial interface. In this Quickstart we will only describe debugging via TCP/IP.

4.1 Starting the GDB server on the target

In this passage you will learn how to start the GDB server on the target. The GDB server will be used to start and control the *myHelloWorld* program.

To debug a program with GDB, the program needs extended debugging symbols. This has already been added while building the program.



- Open Microcom
- Type **root** and press **Enter**
- Start the GDB server:
gdbserver 192.168.3.11:10000 myHelloWorld

You have started the GDB server on the target. The GDB server is now waiting for connections on TCP port 10000.

4.2 Configuring and starting the debugger in Eclipse

In this passage you will learn how to configure your project settings to use Eclipse with the GNU debugger. After the configuration of your project settings, the GNU debugger will start and connect to the GDB server on the target.

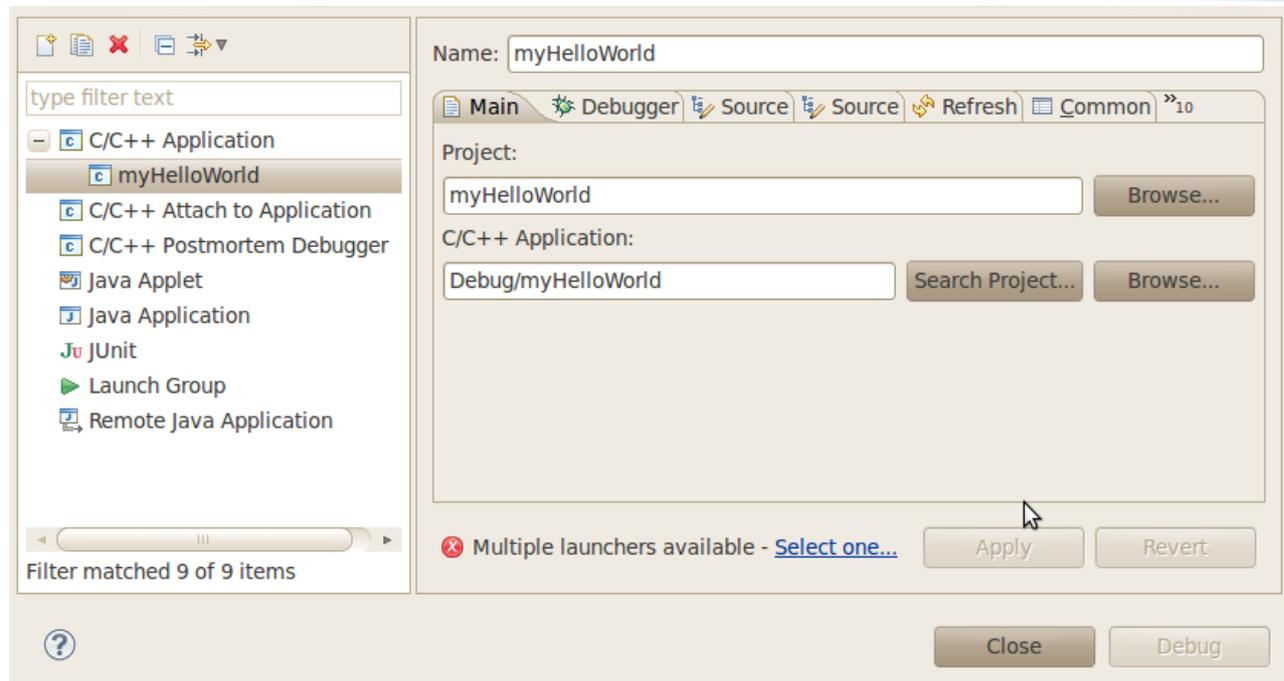
- Start Eclipse if the application is not started yet
- Right-click on the *myHelloWorld* project in the *Navigator* window
- Select *Debug As* ► *Debug Configurations*

A dialog to create, manage and run applications appears.

- Select *myHelloWorld* under *C/C++ Application*

Create, manage, and run configurations

Multiple launchers available - select one to continue



- Select the *Debugger* tab

Create, manage, and run configurations

Multiple launchers available - select one to continue



- Select *gdbserver Debugger* from the *Debugger* drop-down box
- Click the *Browse* button right beside the *GDB debugger* input field.
A new dialog opens to choose the GDB executable.
- Click on *File System*
- Navigate to the directory `/opt/SELAS.Toolchain*/arm-cortexa9-linux-gnueabi/gcc-4.6.2-glibc-2.14.1-binutils-2.21.1a-kernel-2.6.39-sanitized/bin`
- Select the file *arm-cortexa9-linux-gnueabi-gdb*
- Click *OK*

Create, manage, and run configurations

Multiple launchers available - select one to continue



Name: myHelloWorld

Debugger: gdbserver Debugger

Stop on startup at: main

Debugger Options

Main Shared Libraries Connection

GDB debugger: /opt/OSELAS.Toolchain-1.99.3/arm-113 Browse...

GDB command file: Browse...

(Warning: Some commands in this file may interfere with the startup operation of the debugger, for example "run".)

Non-stop mode (Note: Requires non-stop GDB)

Enable Reverse Debugging at startup (Note: Requires Reverse GDB)

Multiple launchers available - [Select one...](#) Apply Revert

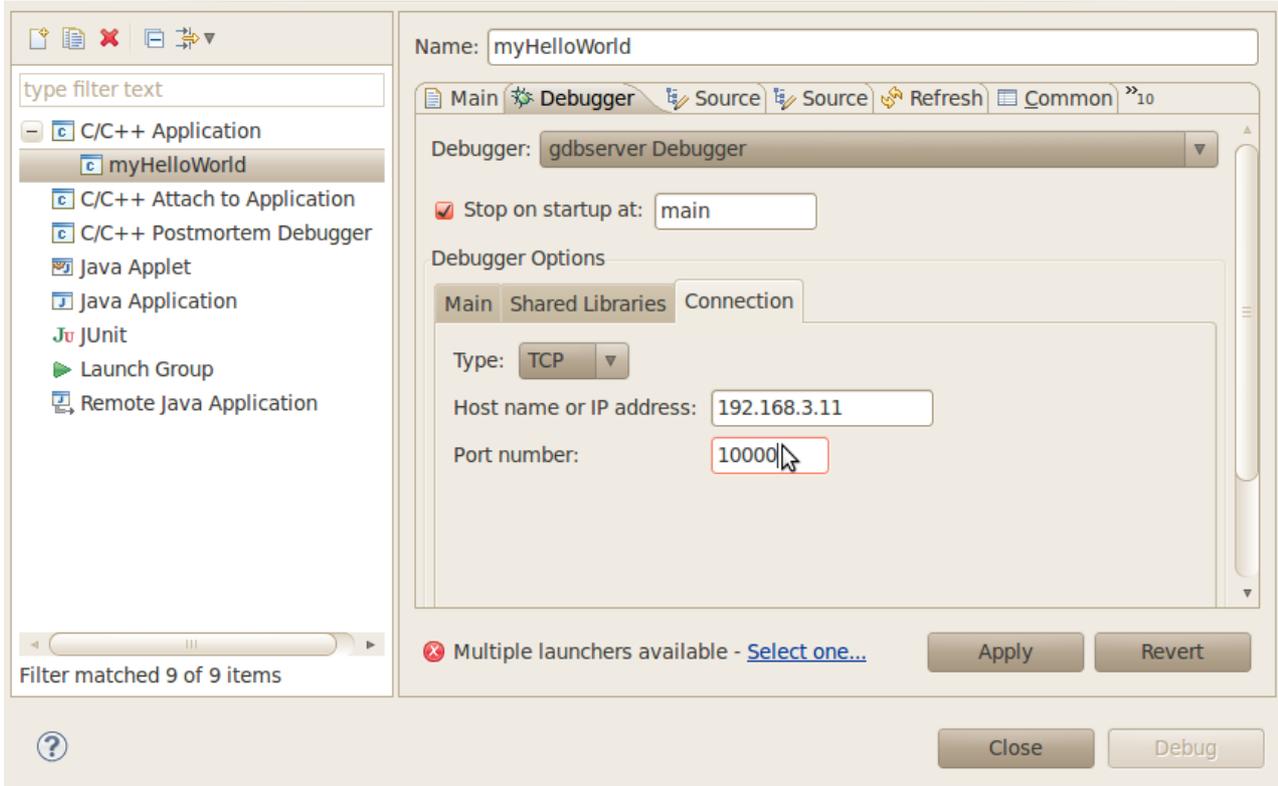
Close Debug

- Keep the *GDB command file* field empty
- Select the *Connection* tab and select *TCP* in the drop-down box
- Enter **192.168.3.11** (the target's IP address) in the *Host name* input field. The host's GDB will connect to this IP address to communicate with the target's GDB-server.

Create, manage, and run configurations

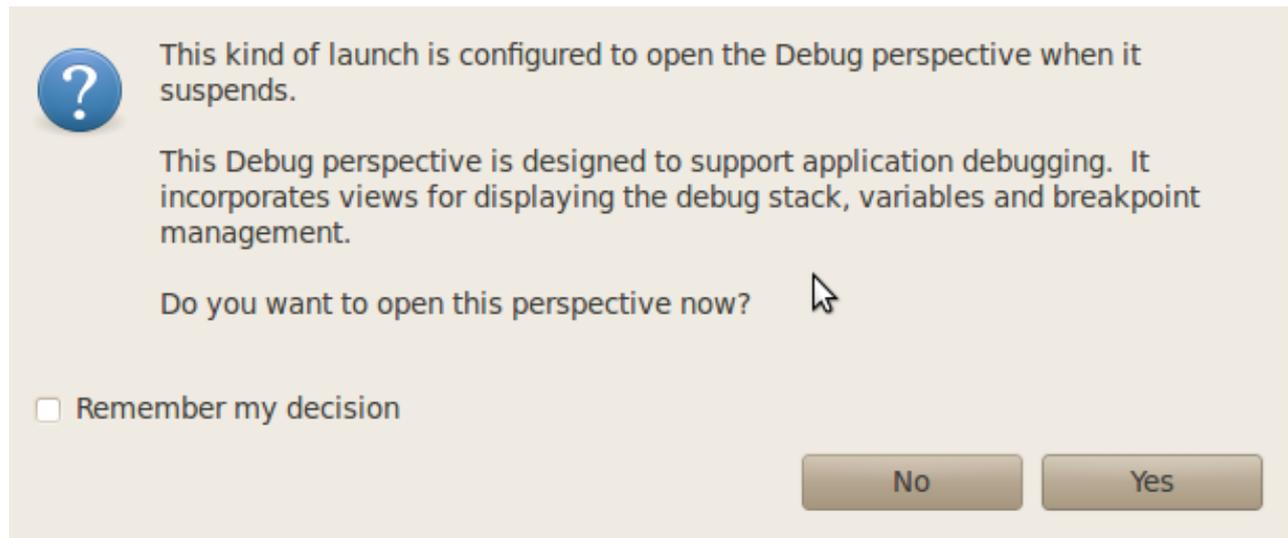


Multiple launchers available - select one to continue



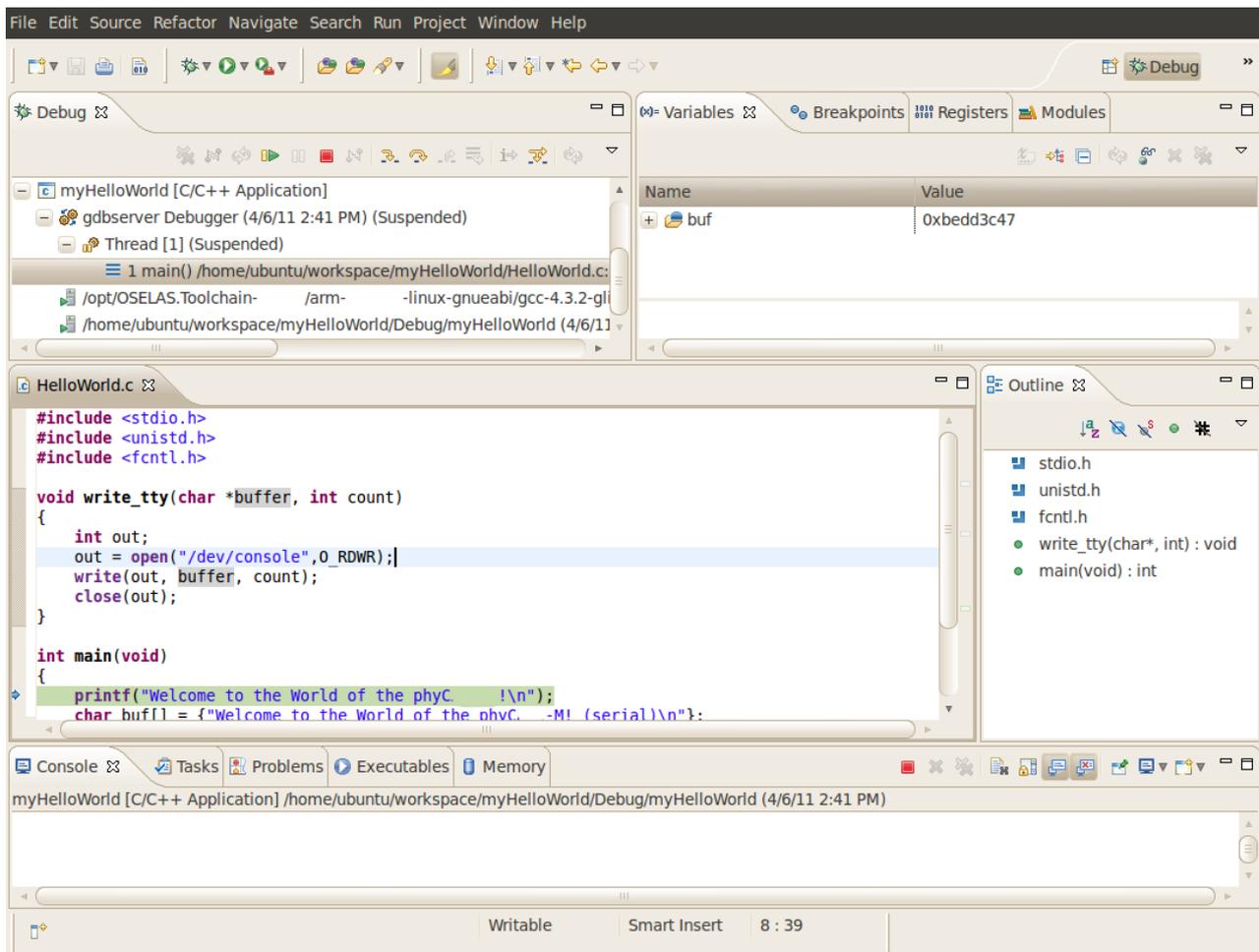
- Click *Apply*
- Click *Debug*

A new dialog appears.



- Select Yes to switch to the Debug perspective

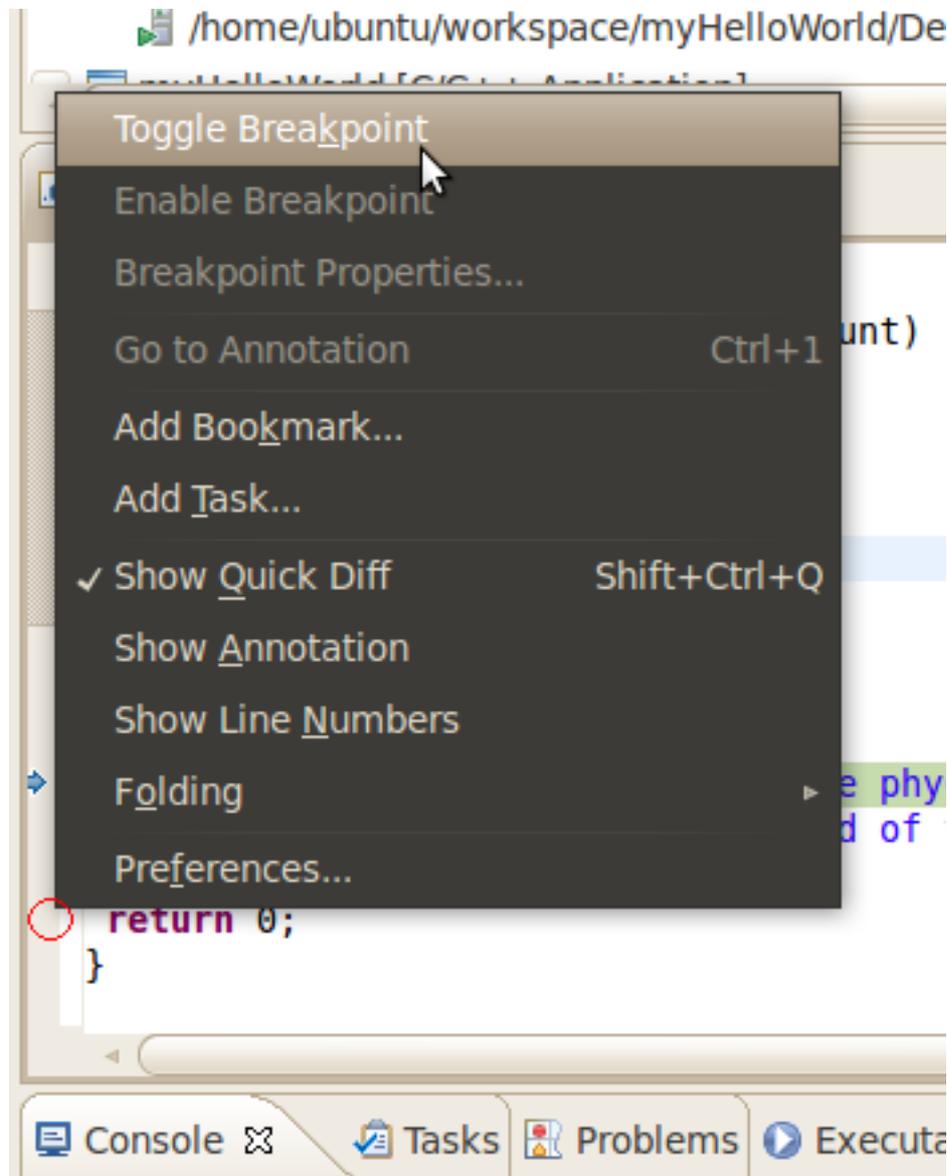
The debug perspective opens and the debugger stops automatically at the first line. The host's GDB is now connected to the GDB server on the target.



You have configured your project for remote debugging. You have started the GNU debugger in Eclipse and connected the host's GDB with the target's GDB server. You can now start to debug the project.

4.3 Setting a Breakpoint

Now you will set a breakpoint in your program. The breakpoint will be set on the last line of the function `main()`. If you resume the application, the debugger will stop on this line.

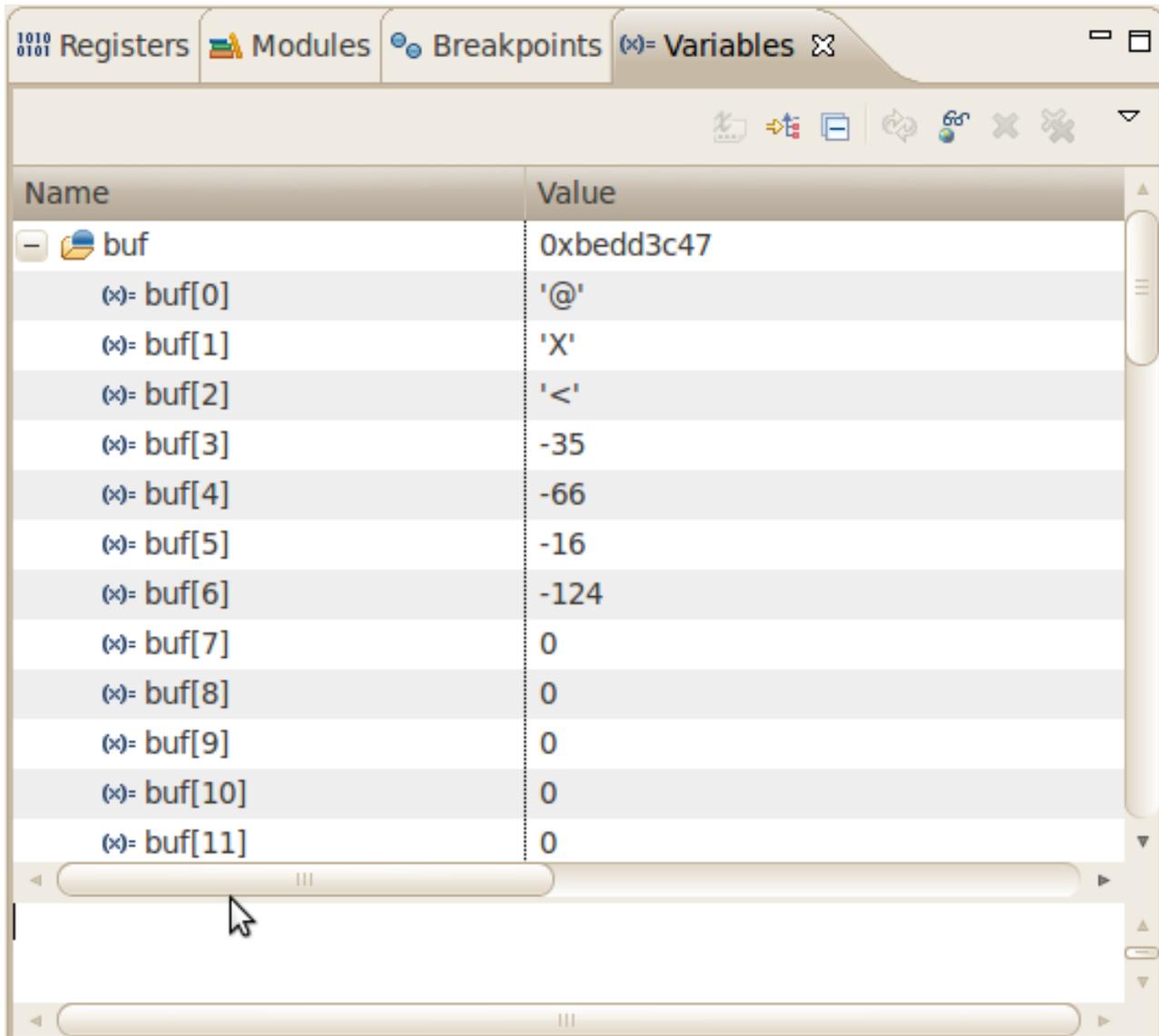


- Select the last line in `main()`
- Right-click into the small grey border on the left-hand side and select *Toggle Breakpoint* to set a new breakpoint

4.4 Stepping and Watching Variable Contents

In this part you will step through the example project with the debugger. You will also learn how to check the content of a variable.

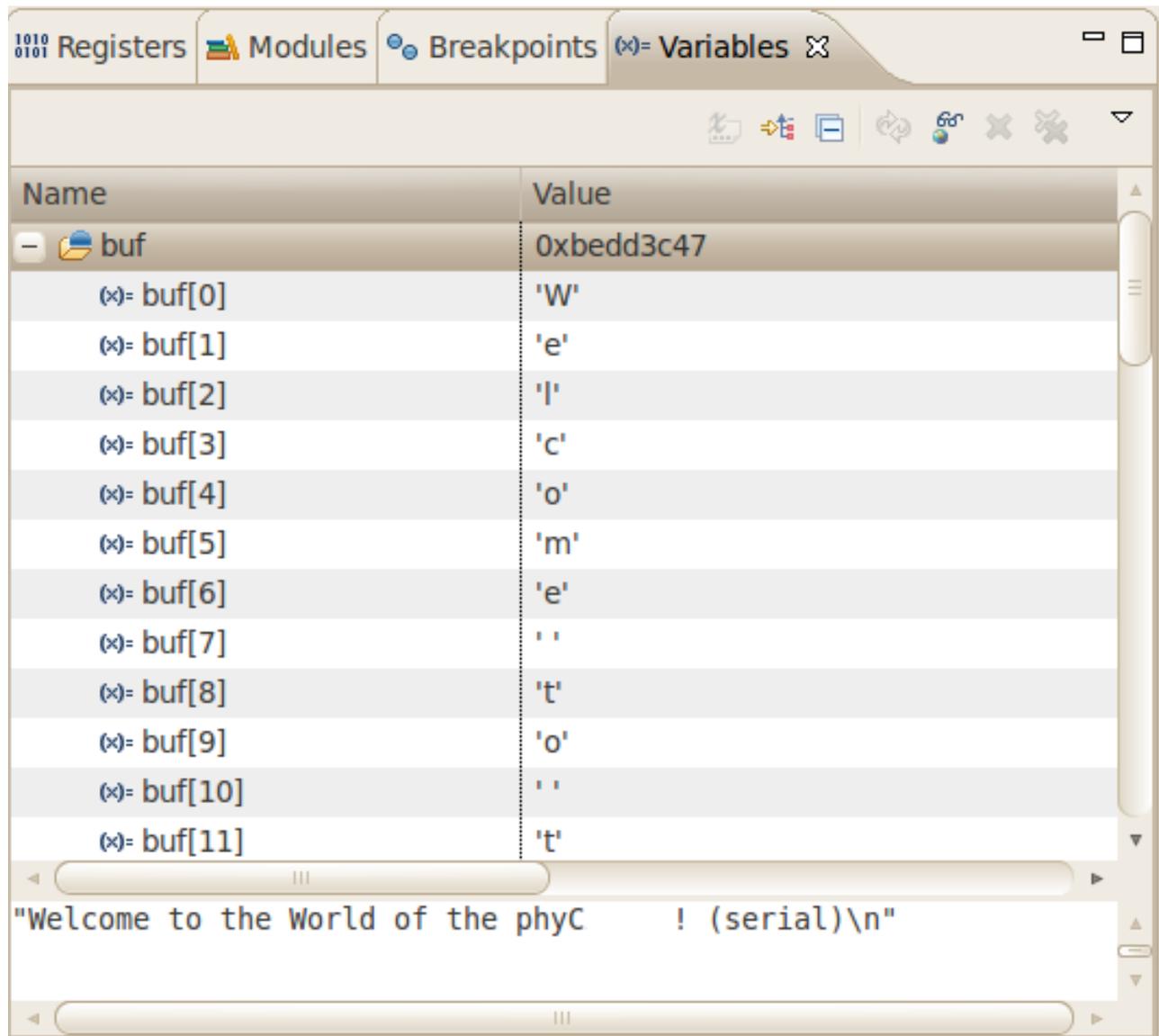
- Expand *buf* in the *Variables* window



- Click the *Step Over* button in the *Debug* window to step to the next line



You will see the content of the *buf* variable in the *Variables* window.



The screenshot shows a debugger window with the 'Variables' tab selected. The variable 'buf' is expanded, showing its memory address '0xbedd3c47' and its contents as a character array. The characters are 'W', 'e', 'l', 'c', 'o', 'm', 'e', ' ', 't', 'o', ' ', 't'. Below the variable list, a scrollable text area displays the string: "Welcome to the World of the phyC ! (serial)\n".

Name	Value
buf	0xbedd3c47
(x)= buf[0]	'W'
(x)= buf[1]	'e'
(x)= buf[2]	'l'
(x)= buf[3]	'c'
(x)= buf[4]	'o'
(x)= buf[5]	'm'
(x)= buf[6]	'e'
(x)= buf[7]	' '
(x)= buf[8]	't'
(x)= buf[9]	'o'
(x)= buf[10]	' '
(x)= buf[11]	't'

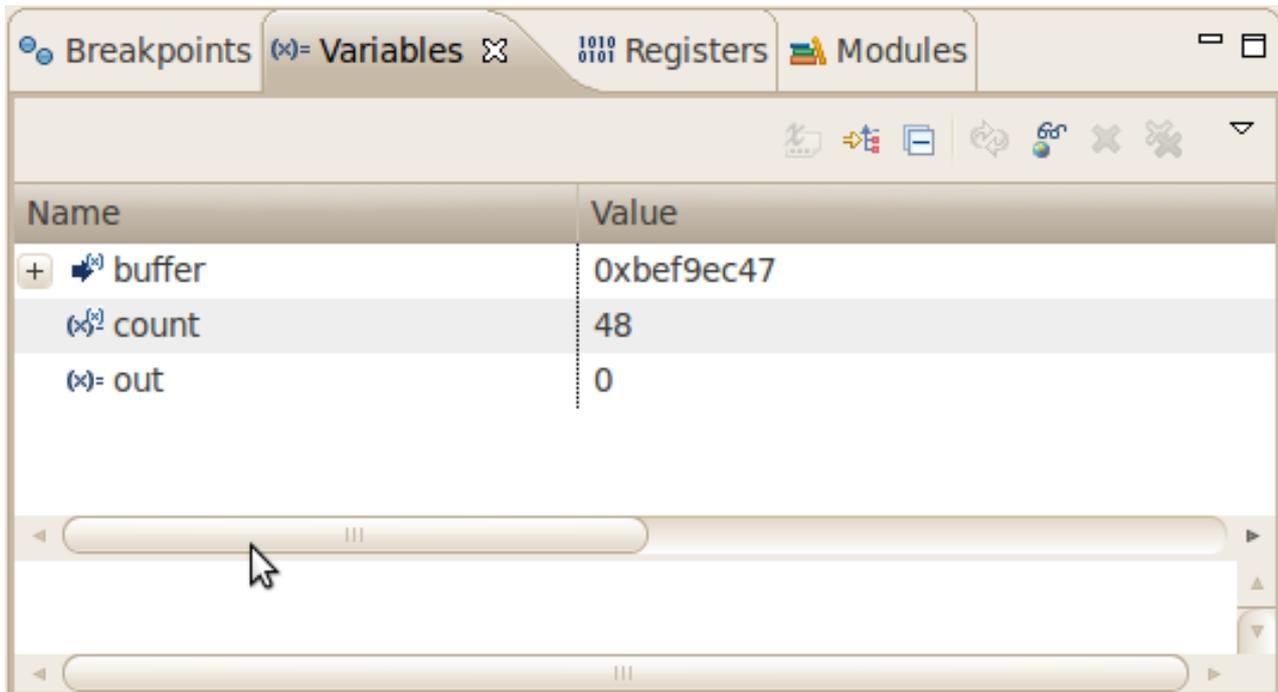
"Welcome to the World of the phyC ! (serial)\n"

- Click on the variable *buf*
- Then click the button *Step into* to enter the function *write_tty()*



The debugger stops in *write_tty()*.

You will see the following variable window:



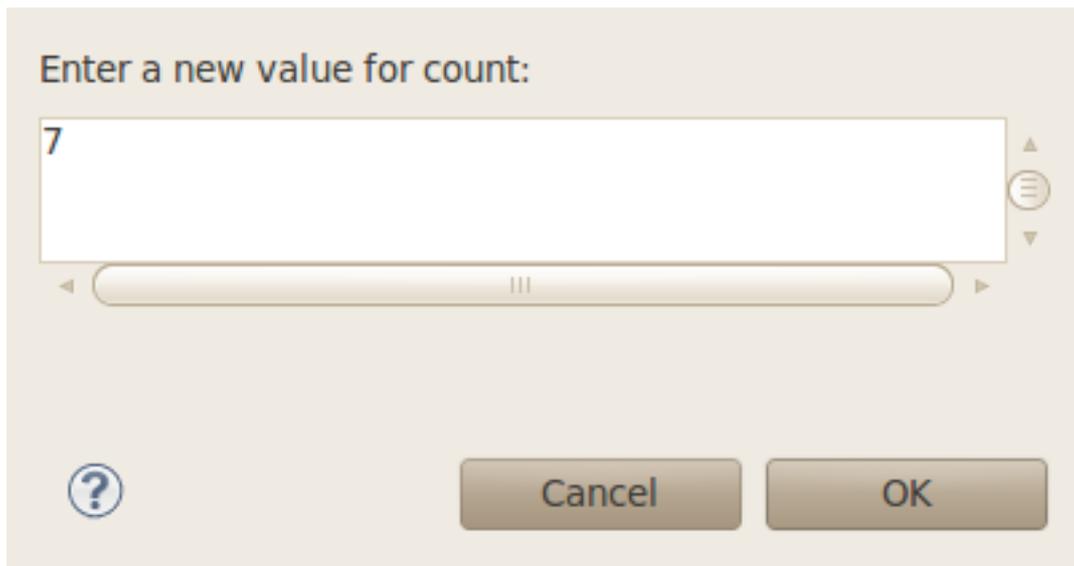
- Click on the variable *buffer*

You will probably see a different address on the buffer pointer. Remember what address is shown in your case; you will need this address later.

4.5 Stepping and Watching Variable Contents

In this section you will change the value of a variable. At the end of this part you will see the effect of this change.

- Select the *count* variable in the *Variables* window
- Right-click on *count* and select *Change Value*



- Change the value of *count* to **7** and click *OK*
- Open *Microcom* if the application is not already opened
- Go back to *Eclipse*
- Click the *Step Over* button **two times**



- Change to *Microcom*

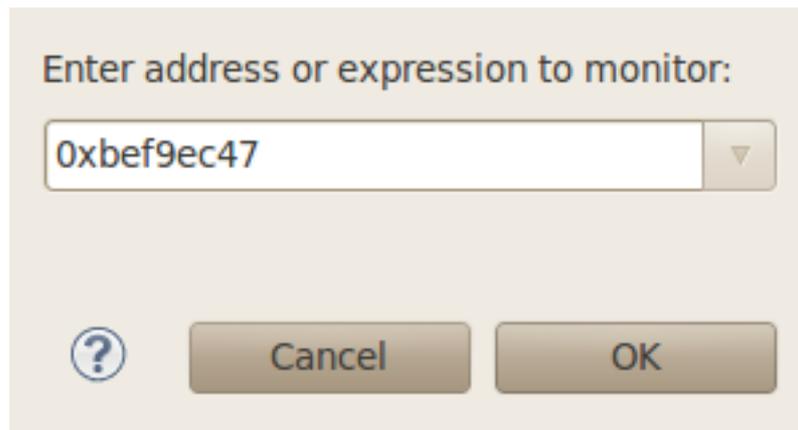
```
root@phyCARD:~# gdbserver 192.168.3.11:10000 myHelloWorld
Process myHelloWorld created; pid = 591
Listening on port 10000
Remote debugging from host 192.168.3.10
Welcome to the World of the phyC.    !
Welcome
```

You will see the output *Welcome* in the Microcom window. This shows when changing the *counter* variable's value to 7 only the first seven characters of the buffer are output, instead of the whole sentence.

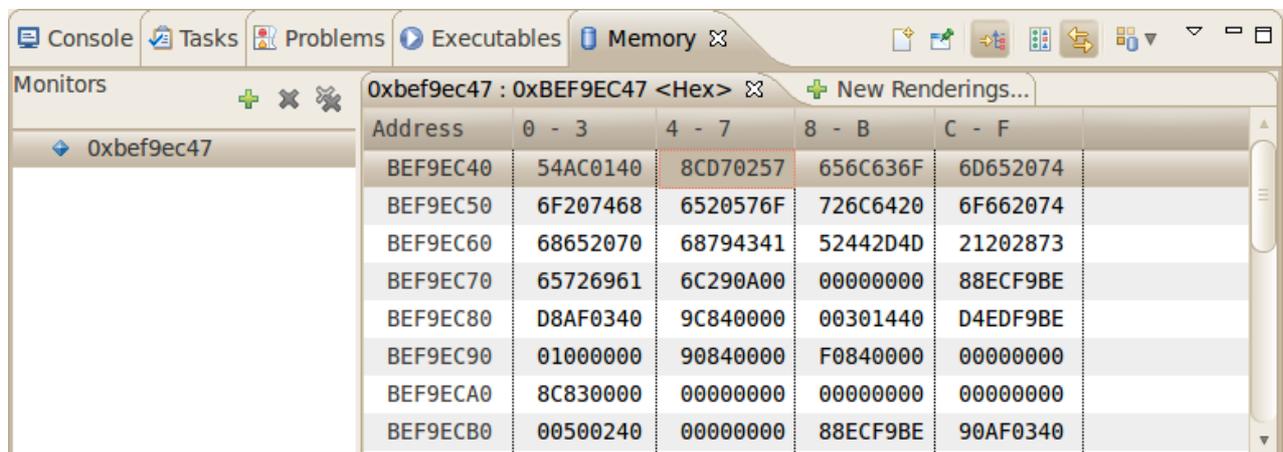
4.6 Using the Memory Monitor

In the last section of this chapter you will use the memory monitor to control the content at a memory address.

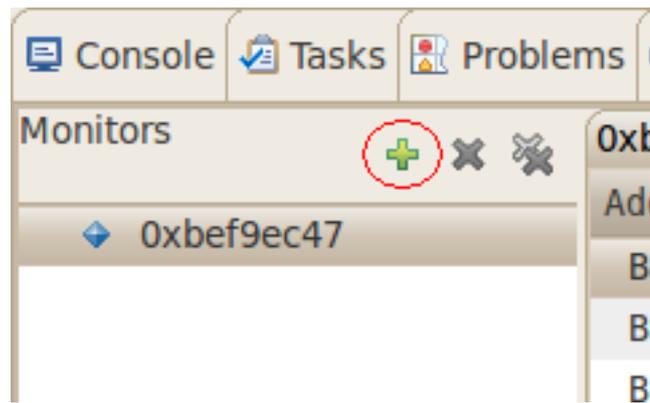
- Select the *Memory* tab
- Click *Add Memory Monitor*



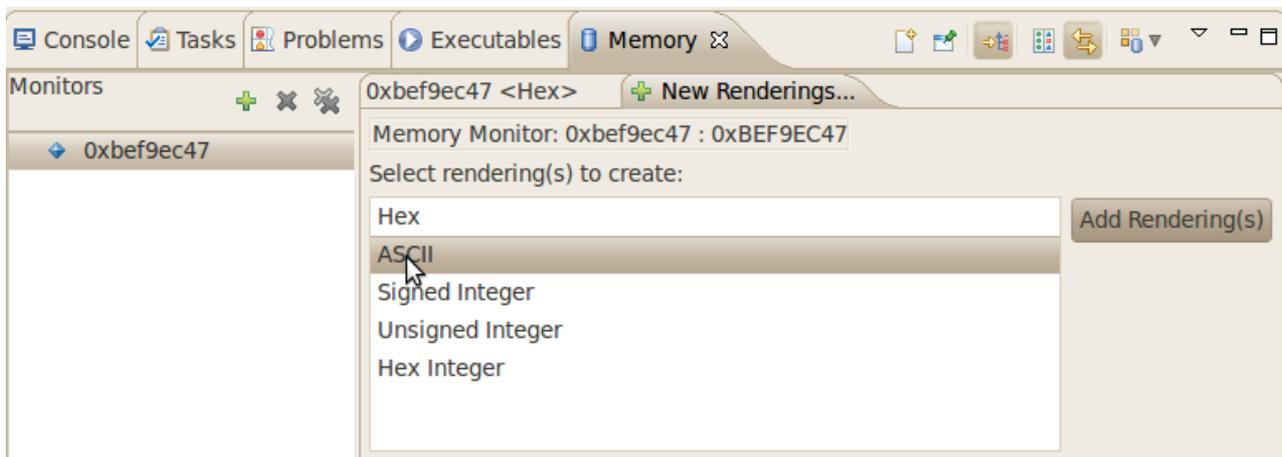
- Enter the address of buffer and click OK. Remember that the variable's address might differ from your system.



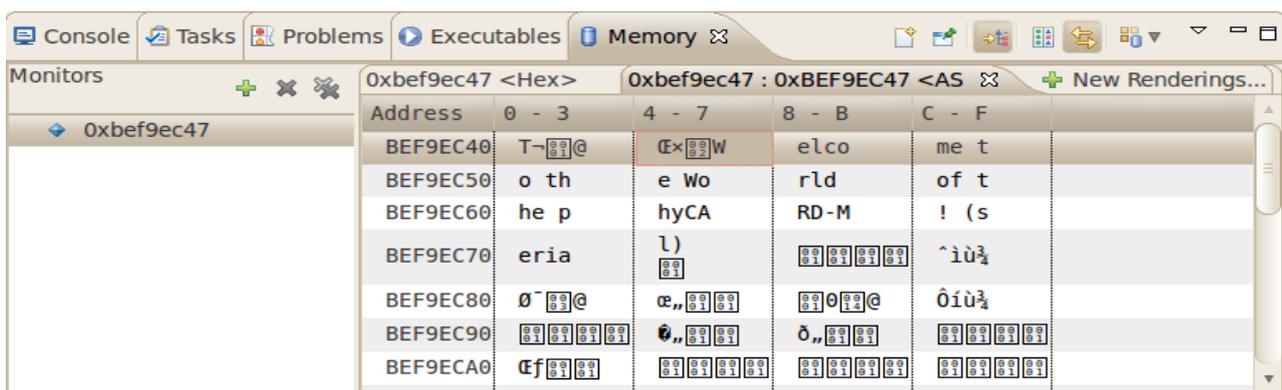
- Change the window size



- Click *Add Rendering*



- Select *ASCII* and click *OK*



You can see the contents of the variable *buffer* at the address *0xbee13ca7* (or whatever address is used on your system).

- Now click the *Resume* button from the menu bar



```
HelloWorld.c ✕
int out;
out = open("/dev/console", O_RDWR);
write(out, buffer, count);
close(out);
}

int main(void)
{
    printf("Welcome to the World of the phyC    !\n");
    char buf[] = {"Welcome to the World of the phyC    ! (serial)\n"};
    write_tty(buf, sizeof(buf) - 1);
    return 0;
}
```

The debugger stops at the breakpoint in the last line of *main()*.

- Click the *Resume* button to end the application



You have successfully passed the debugging chapter. You are now able to configure and use Eclipse for remote debugging. You can step through a project, watch and change the content of variables, and you can use the memory monitor to view the content at a memory address.

5 Summary

This Quick Start manual gave you a general “Rapid Development Kit” description, as well as software installation advice and an example program enabling quick out-of-the-box start-up of the phyFLEX-i.MX6 in conjunction with the Eclipse IDE and GNU C/C++ software tools.

In the *Getting Started* section you learned how to configure your host to provide a basis for working with your target platform. You installed the Rapid Development Kit software and learned how to copy and run a program on the target.

In the *Getting More Involved* section you got step-by-step instructions on how to configure and build a new kernel, modify the example application, create and build new projects, and copy programs to your phyFLEX-i.MX6 using Eclipse.

The *Debugging* part of this Quick Start gave you information on setting up and using the GNU debugger with the Eclipse IDE. You learned how to set breakpoints, watching and changing variable contents and using the memory monitor.

6 Installing Linux on the phyFLEX-i.MX6

This part provides preparations on how to update bootloader (Barebox) on the phyFLEX-i.MX6 and how to write a kernel and/or a root file system image into the target's flash memory.

6.1 Configure Barebox Environments Variables

- Connect the serial cable with the UART0 (connector X51) on the target and the first serial interface on your host
- Connect the cross-over Ethernet cable with the connector (POE, X28) on the target and the correct network card of your host



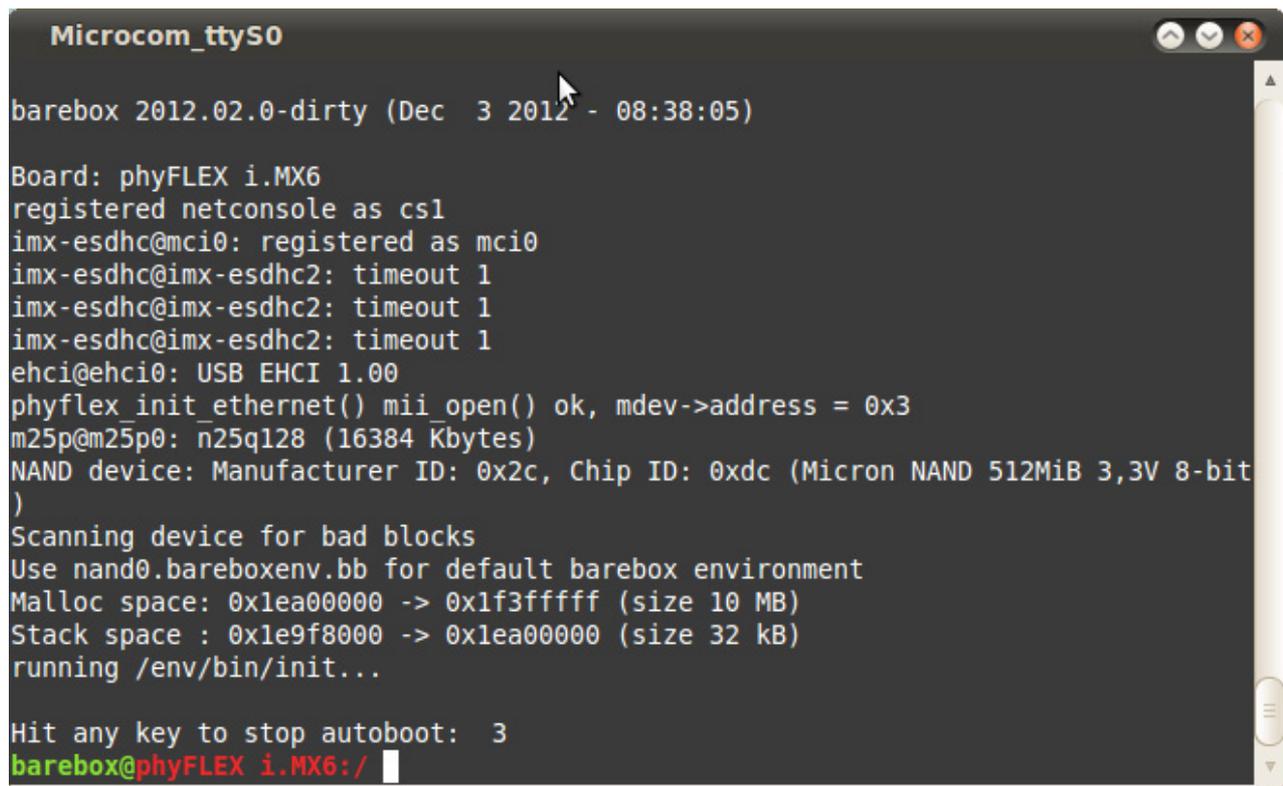
- Click the *Microcom* icon on your desktop

Microcom is configured with the following configuration:

115200 baud, 1 start bit, 8 data bits, 1 stop bit, no parity, no flow control.

If you want to use another program than Microcom for serial communication, you will have to setup that program with these settings.

- Connect the AC adapter with the power supply connector PWR (12V) on your board

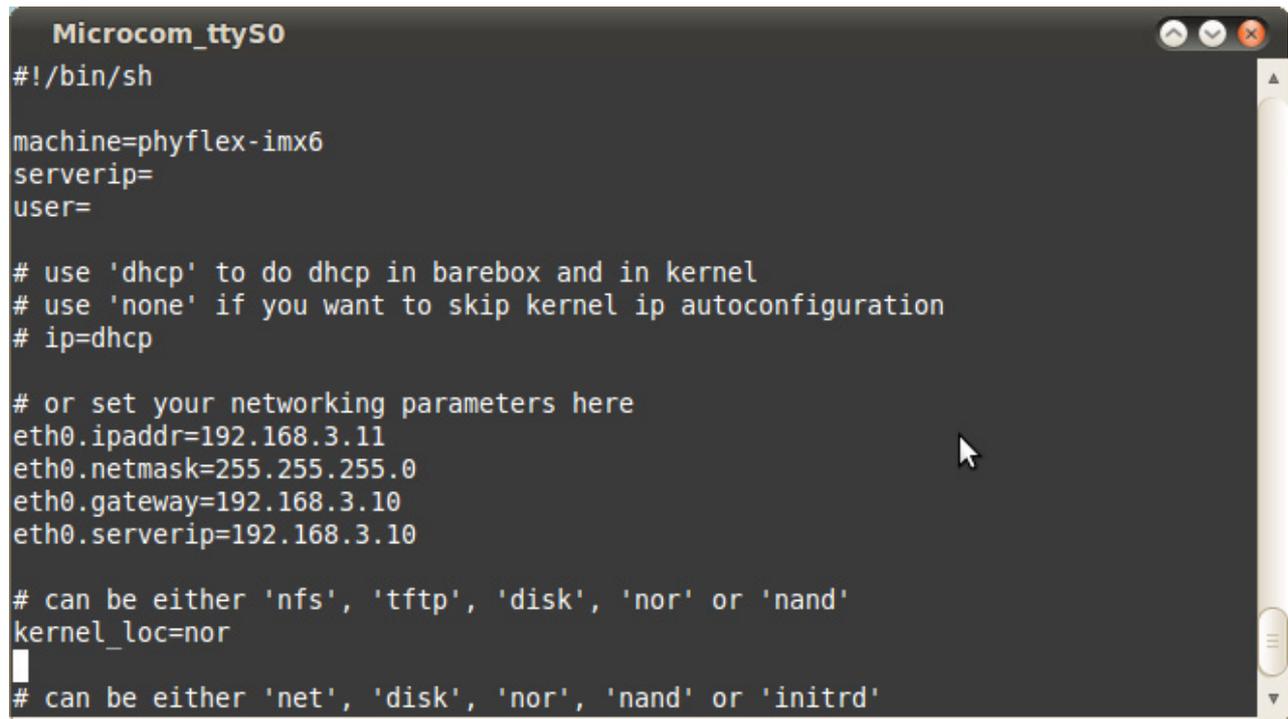


```
Microcom_ttyS0
barebox 2012.02.0-dirty (Dec  3 2012 - 08:38:05)

Board: phyFLEX i.MX6
registered netconsole as cs1
imx-esdhc@mci0: registered as mci0
imx-esdhc@imx-esdhc2: timeout 1
imx-esdhc@imx-esdhc2: timeout 1
imx-esdhc@imx-esdhc2: timeout 1
ehci@ehci0: USB EHCI 1.00
phyflex_init_ethernet() mii_open() ok, mdev->address = 0x3
m25p@m25p0: n25q128 (16384 Kbytes)
NAND device: Manufacturer ID: 0x2c, Chip ID: 0xdc (Micron NAND 512MiB 3,3V 8-bit
)
Scanning device for bad blocks
Use nand0.bareboxenv.bb for default barebox environment
Malloc space: 0x1ea00000 -> 0x1f3ffffff (size 10 MB)
Stack space : 0x1e9f8000 -> 0x1ea00000 (size 32 kB)
running /env/bin/init...

Hit any key to stop autoboot:  3
barebox@phyFLEX i.MX6:/
```

- Press any key to stop autoboot
- Type **edit /env/config** to check and edit the configuration file



```
Microcom_ttyS0
#!/bin/sh

machine=phyflex-imx6
serverip=
user=

# use 'dhcp' to do dhcp in barebox and in kernel
# use 'none' if you want to skip kernel ip autoconfiguration
# ip=dhcp

# or set your networking parameters here
eth0.ipaddr=192.168.3.11
eth0.netmask=255.255.255.0
eth0.gateway=192.168.3.10
eth0.serverip=192.168.3.10

# can be either 'nfs', 'tftp', 'disk', 'nor' or 'nand'
kernel_loc=nor

# can be either 'net', 'disk', 'nor', 'nand' or 'initrd'
```

You will see the configuration file that contains Barebox's environment variables.

The default IP address of the target is 192.168.3.11 and the default server IP address is 192.168.3.10. If you want to set up a different network configuration, you can edit the following lines of the configuration file:

```
eth0.ipaddr=target IP address
eth0.netmask=target netmask
eth0.serverip=server IP address
```

- Type **CTRL-D** to save the settings to the file
- Type **save** to write the settings to the target's flash
- Press the RESET button on your board. The target will restart with the new settings applied.

6.2 Restoring the Barebox Default Configuration

If you want to restore the default Barebox configuration, you can use the following commands to delete the Barebox environment partition:

- **erase /dev/env0**

After pressing the RESET button on your board, the default Bootloader configuration will be used. This also means that you will be asked to enter the MAC address of your board again.

6.3 Update the Bootloader

The bootloader used on the phyFLEX-i.MX6 is *Barebox* which is already preinstalled. If nothing is installed or something went wrong while updating the bootloader please contact our support department. Otherwise you can update the bootloader from the command line of the bootloader.

You can download the bootloader from the TFTP server into the target's RAM, erase the corresponding flash partition, and write the barebox from RAM into flash with one simple command: *update*.

Before executing this command, you should check that your Barebox environment is properly configured. See chapter 6.1 for more information.

In the directory */tftpboot* in your live environment or under */PHYTEC/BSP* on your DVD you can find a file called *barebox-image* - this file is the bootloader.

- Open Microcom if not already open and press the RESET button on the target. You will see the message "*Hit any key to stop autoboot.*"
- Press any key to stop autoboot
- Type **update -t barebox -d nor -f barebox-image** to download and write the barebox into target's flash
- Press the RESET button on the board to restart your target with the written barebox

6.4 Writing the Kernel into Flash

In the directory */tftpboot* in your live environment or under */PHYTEC/BSP* on your DVD you can find a file called *uImage-pfla02* - this file is the Linux kernel image.

You can download the kernel from the TFTP server into the target's RAM, erase the corresponding flash partition, and write the kernel from RAM into flash with one simple command: *update*.

Before executing this command, you should check that your Barebox environment is properly configured. See chapter 6.1 for more information.

- Open Microcom and press the RESET button on the target.
You will see the message "*Hit any key to stop autoboot.*"
- Press any key to stop autoboot
- Type **update -t kernel -d nor -f uImage-pfla02** to download and write the kernel into target's flash
- Press the RESET button on the board to restart your target with the written kernel

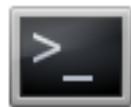
6.5 Writing the root filesystem into Flash

Writing the root filesystem into flash is a little bit more difficult, because we need and must configure a sd card.

In the directory `/tftpboot` in your live environment or under `/PHYTEC/BSP` on your DVD you can find two files called `root-pfla02.tgz` and `root-pfla02.ubi` - the first is the compressed root filesystem the second the root filesystem in ubi format.

We need a sd card with a 512 MB ext3-partition or more capacity.

- First plugin the sd card to your host pc. Then open a new terminal window.

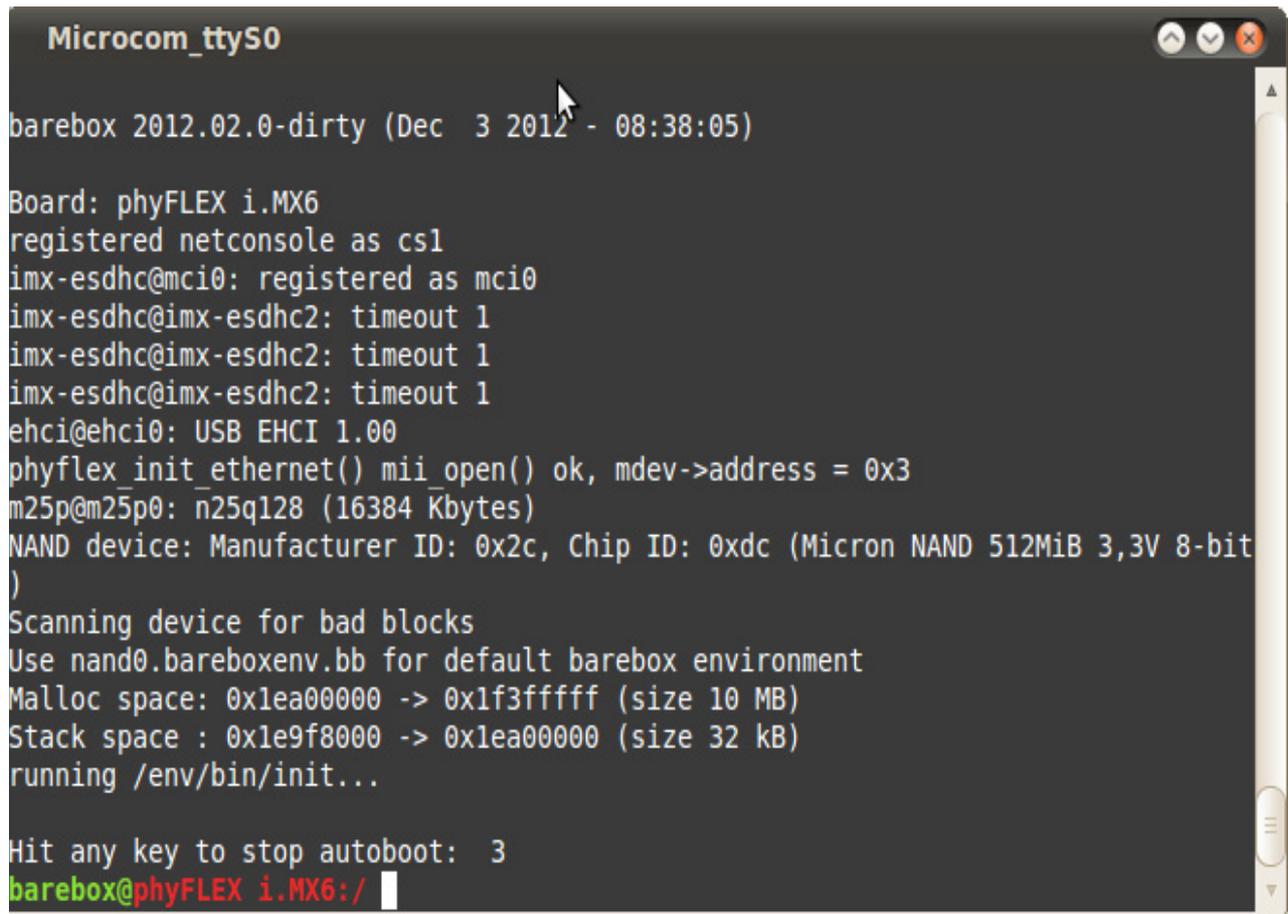


Terminal

- Normally the inserted sd card is mounted automatically under `/media/"DeviceName"`. So we switch to this directory in the terminal:
cd /media/"DeviceName"
- Now we can uncompress the root filesystem to the mounted sd card with following command:
sudo tar xzf /tftpboot/root-pfla02.tgz
- Also we must copy the `root-pfla02.ubi` to the sd card.
sudo cp /tftpboot/root-pfla02.ubi /media/"DeviceName"/home/
- Go to a directory outside the sd card and unmount the sd card with:
cd ...
sudo umount /media/
- Unplug the sd card from the host and insert the sd card to the sd0 slot on the baseboard

Before we can flash the root filesystem, you should check that your barebox environment is properly configured.

- Open Microcom and press the RESET button on the target.
You will see the message "*Hit any key to stop autoboot.*"
- Press any key to stop autoboot



```
Microcom_ttyS0
barebox 2012.02.0-dirty (Dec 3 2012 - 08:38:05)

Board: phyFLEX i.MX6
registered netconsole as cs1
imx-esdhc@mci0: registered as mci0
imx-esdhc@imx-esdhc2: timeout 1
imx-esdhc@imx-esdhc2: timeout 1
imx-esdhc@imx-esdhc2: timeout 1
ehci@ehci0: USB EHCI 1.00
phyflex_init_ethernet() mii_open() ok, mdev->address = 0x3
m25p@m25p0: n25q128 (16384 Kbytes)
NAND device: Manufacturer ID: 0x2c, Chip ID: 0xdc (Micron NAND 512MiB 3,3V 8-bit
)
Scanning device for bad blocks
Use nand0.bareboxenv.bb for default barebox environment
Malloc space: 0x1ea00000 -> 0x1f3fffff (size 10 MB)
Stack space : 0x1e9f8000 -> 0x1ea00000 (size 32 kB)
running /env/bin/init...

Hit any key to stop autoboot: 3
barebox@phyFLEX i.MX6:/
```

- Type the following command to check your Barebox settings:
edit /env/config
You will see the configuration file which contains Barebox's environment variables.
- Change following values in the configuration file if you have only one partition on the sd card:
rootfs_loc=disk
rootfs_type=ext3
rootfs_part_linux_dev=mmcblk0p1
- Type **CTRL-D** to save the settings temporarily to the file
- Type the following command to boot the board with the temporarily configuration:
boot
- After the board is booted and you have successfully logged it is time to flash the root filesystem into the flash with only one command:
ubiformat -y /dev/mtd8 -f root-pfla02.ubi

```
s 293us.  
  
[BDP]  
  
phyFLEX-i.MX6  
  
OSELAS(R)-phyFLEX-i.MX6-master / phyFLEX-i.MX6-master  
ptxdist-2012.03.0/2013-40-11T09:40:31+0100  
  
phyFLEX-i login: root  
root@phyFLEX-i:~# ubiformat -y /dev/mtd8 -f root-pfla02.ubi  
ubiformat: mtd8 (nand), size 1062600704 bytes (1013.4 MiB), 8107 eraseblocks of 131072 by  
28.0 KiB), min. I/O size 2048 bytes  
libscan: scanning eraseblock 8106 -- 100 % complete  
ubiformat: 8093 eraseblocks have valid erase counter, mean value is 0  
ubiformat: 14 bad eraseblocks found, numbers: 1440, 3429, 3694, 3969, 5019, 6064, 7461, 7  
020, 8065, 8103, 8104, 8105, 8106  
ubiformat: flashing eraseblock 891 -- 100 % complete  
ubiformat: formatting eraseblock 8106 -- 100 % complete  
root@phyFLEX-i:~#
```

After the root filesystem is flashed into nand reset the target. The flashed root filesystem from NAND is now used to start the module.

7 Setup your own Linux-Host-PC

In this chapter we give an overview of the modifications which we made to the Ubuntu version on the Linux-phyFLEX-i.MX6-Kit-DVD in comparison to the original Ubuntu. In the following we distinguish between optional and essential modifications. So you can see faster which changes are important to execute this Quickstart in case you don't want to use our modified Ubuntu version. You can find a step-by-step instruction of the essential changes in order to modify your own distribution.



We can't guarantee that the presented changes are compatible to other distributions or versions. If you want to use another distribution, it might take a lot of individual initiative. We do not support other distributions. You should be sure about what you do.

7.1 Essential settings

In the following you see a short instruction of the important settings which are essential to guarantee the execution of this Quickstart.

7.1.1 Installation of software packages

Begin with the installation of the required software packages. Therefore first add the needed repositories so that the package manager *APT* (*Advance Packaging Tool*) knows the packages and can install them.

- Open with an editor and with root privileges `/etc/apt/sources.list`
- Remove the commentary marks on the following universe-repositories:
deb http://de.archive.ubuntu.com/ubuntu/ lucid universe
deb-src http://de.archive.ubuntu.com/ubuntu/ lucid universe
deb http://de.archive.ubuntu.com/ubuntu/ lucid-updates universe
deb-src http://de.archive.ubuntu.com/ubuntu/ lucid-updates universe
- Add two additional repositories:
deb http://archive.canonical.com/ubuntu lucid partner
deb-src http://archive.canonical.com/ubuntu lucid partner
- Save the changes

Then add the repositories and now you can continue using *APT*.

- You must update *APT* to access to the new package sources. That occurs with the following command:
sudo apt-get update

- Now you can install the packages. It is useful to split the commands to keep the overview.

Packages which are needed for compiling and building the Board Support Package:
sudo -y apt-get install libncurses-dev flex bison texinfo expect gettext patch g++ gzip

Wine, Eclipse and their plug ins:

sudo -y apt-get install cabextract sun-java6-jre eclipse-jdt qt4-dev-tools

For setting up the TFTP server:

sudo -y apt-get install xinetd tftpd tftp

During installation it can happen that some programs ask for a license agreement. Follow the shown steps.

The first preparations are finished and you will find the instructions to setup the bigger software packages in the next chapters.

7.1.2 Setup of toolchain, PTXdist and BSP

The important packages are installed now and we begin with the setup of *PTXdist* which we need to modify the *Board Support Package* by your own.

- First we create a local directory in our */home/* folder and change into this directory:
mkdir /home/\$USER/local
cd /home/\$USER/local
- In the second step we unpack the compressed *PTXdist* which you will find on our Linux-phyFLEX-i.MX6-Kit-DVD under */PHYTEC/BSP* in the folder *local*:
tar -xjf ...PHYTEC/BSP/ptxdist-XXX.tar.bz2
- After this step a *PTXdist* folder is created in which we change by typing:
cd ptxdist*
- Afterwards we can configure, compile and install *PTXdist* via the standardized *GNU-Tools*. For this we use the following commands:
./configure
make
sudo make install
- The installation is finished and the temporary folder *local* can be removed:
cd ../../
rm -fr local

PTXdist is now ready for use and can be accessed through the **ptxdist** command. But before we can start cross-compiling, we need a toolchain which provides tools to build the *Board Support Package*.

On our Linux-phyFLEX-i.MX6-Kit-DVD you will find under `/PHYTEC/BSP` an already prebuilt Toolchain for the phyFLEX-i.MX6 which we will use.

- Therefore we first decompress the toolchain using the following command:
tar -xvPpjf ...PHYTEC/BSP/arm-cortexa9-linux-gnueabi.tar.bz2

By this operation the toolchain is directly archived under `/opt/OSELAS*`.



If you want to build your own toolchain or if you want further information please read our "BSP-Quickstart phyFLEX-i.MX6" at our FTP under <ftp://ftp.phytec.de>.

Finally our attention is focused on the setup of the *Board Support Package*.

In our modified Ubuntu version our BSPs are located under `/opt/PHYTEC_BSPs/`. So let's use the following procedure:

- **sudo mkdir /opt/PHYTEC_BSPs**
- Now only root has the privilege to access this folder. Therefore we must change the permissions:
sudo chmod 777 /opt/PHYTEC_BSPs
- After this we switch over to the folder and use our compressed but not yet built PHYTEC-BSP on the DVD under `/PHYTEC/BSP/`
cd /opt/PHYTEC_BSPs
tar -xzf ...PHYTEC/BSP/BSP-Phytec-phyFLEX-i.MX6*.tar.gz
- The BSP is unpacked and we change into the created folder:
cd BSP-Phytec-phyFLEX-i.MX6*
- The time has come to handle with *PTXdist*. First of all we must specify our platform with which we will work:
ptxdist platform configs/phyFLEX-i.MX6/platformconfig
- The required toolchain should be found automatically and we can use *PTXdist* to build the BSP:
ptxdist go
Depending on your system hardware this procedure can take up to several hours
- Finally we must create the images which we can flash to the target
ptxdist images



Congratulations - you have successfully completed the most time-consuming pre-requirements. In the next chapter you will be informed about everything concerning the set up of Eclipse.

7.1.3 Setting up Eclipse and integrate plug ins

In this chapter we setup *Eclipse* and integrate the plug ins for *QT* and *C/C++*. Thereby you can assign own programs, written in *Eclipse*, to the target.

- First we must create a *workspace* folder, in which we can save the *Eclipse* projects. In our example we make this in our */home/* folder:
mkdir /home/\$USER/workspace
- Thereafter we can open *Eclipse* and insert the path to our created workspace in the pop-up window:
eclipse
- Eclipse is started and now we click at *Help* in the menu bar ► *Install new Software*
- Thereupon a window was opened and in the text field "*Work with*" we enter the following address: **http://download.eclipse.org/tools/cdt/releases/galileo** and click on "**Add**"
- After a while the software which we can integrate appears in the area with a scrollbar. We check "**CDT Main Features**" and click at "**Next**".
- Now the system shows us an overview of the installation details, which we skip with a click on **Next**
- At last the system shows us the licensing agreement, which we accept and after we have clicked on **Finish** it begins to install the required software

The *CDT* plug in is installed. We close *Eclipse* and go on with the *QT* integration.

- For the *QT* integration we must use an external package from the *Nokia* website. This package is already available on our DVD under */PHYTEC/Applications*. We unpack the compressed archive in the */usr/lib* folder:
cd /usr/lib
tar xzf ...PHYTEC/Applications/qt-eclipse-integration-linux.x86-1.6.1.tar.gz
- After that we start *Eclipse* with the option *clean*, which cleans any cached data:
eclipse --clean
- *Eclipse* is started and now we must setup one small thing:
In the menu bar we click on "**Window**" ► "**Preferences**" ► "**QT**" and enter the following *QT* specific information **Name:** *QT_phyFLEX-i.MX6* ► **Bin Path:**

`/opt/PHYTEC_BSPs/BSP-Phytec-phyFLEX-i.MX6*/platform-phyFLEX-i.MX6/sysroot-cross/bin` ► **Include Path:** `/opt/PHYTEC_BSPs/BSP-Phytec-phyFLEX-i.MX6*/platform-phyFLEX-i.MX6/sysroot-host/include`)

- Finally we close *Eclipse* and start it again with the *clean* option



Congratulations! You have successfully integrated two new plugins in Eclipse and now you can start programming in C/C++ and QT in an Eclipse environment. In the next sub-section you will find a short introduction on how to setup a TFTP server.

7.1.4 Setting up a TFTP server

In the chapter "*Installation of software packages*" we have installed the required packages to set up a TFTP server. Now we must change some short settings.

- First we create (or change) the file `/etc/xinet.d/tftp` as follows:

```
service tftp
{
protocol = udp
port = 69
socket_type = dgram
wait = yes
user = nobody
server = /usr/sbin/in.tftpd
server_args = -s /tftpboot
disable = no
}
```

- Then we must create a folder called `/tftpboot`. The TFTP server accesses this folder later.

mkdir /tftpboot

- At last we must set the right permissions:

chmod 777 /tftpboot



You have successfully set up the TFTP server. In the future the phyFLEX-i.MX6 can access to the /tftpboot/ folder to load the images.

7.2 Optional settings

In the following we show the optional settings. These settings are not needed for a successful operation of this Quickstart. They only simplify the handling and the look of the system. For this reason we show the modifications without big explanation.

- Installation of *vim*, the improved *vi* editor
- Creation of desktop-icons for faster and easier start of required programs
- The following modifications were made to the look of Ubuntu:
 - Other wallpaper and associated options were adjusted with the help of **gconftool-2**
 - Window buttons for Maximize, Minimize and Close were moved to the right side
 - Changing the color of the *Gnome* terminal
 - Installation of **gnome-color-chooser** to change the look of the desktop icons more easily
- Alias for the commands `..` and **wine** added in `/home/$USER/.bashrc`
- **History-search-backward** and **history-search-forward** in `/etc/inputrc` is activated. This allows you to search through your history with the entered string.
- Also there are some scripts that will be executed at the first start after the installation. These scripts ensure that the right permissions are set for the created user.

8 Installation of the modified Ubuntu

In this chapter you will find an instruction on how to install the modified Ubuntu on the Linux-phyFLEX-i.MX6-Kit-DVD.

If another operating system is installed on your computer, you should first make a backup of your important files. Before we can start, make sure that your computer is set to boot from DVD before it boots from a hard disk drive.

- Insert the Linux-phyFLEX-i.MX6-Kit-DVD into your DVD drive
- Start or restart your computer. Your system finds a bootable DVD and starts from it. After a while a language screen appears.
- Select your desired language and click **Install Ubuntu...**
- The **Where are you?** window appears. Select the location closest to your location and click **Forward**.
- The **Keyboard layout** window appears. If the suggested option is not correct, select the correct keyboard layout and click **Forward**.
- The **Prepare disk space** window appears. Now you have different options to install Ubuntu:
 1. Install it side by side (only if another OS was found)
If you want to install Ubuntu with dual booting: in the new partition size area, drag the area between the two partitions to create your desired partition sizes.
 2. Erase and use the entire disk
Installing Ubuntu on your entire hard disk will erase all data that is currently on the drive.
 3. Specify partitions manually
An advanced mode to set the partitions manually. Only for more involved users.
After you have made your decision and adapted it to your wishes click on **Forward**.
- The **Who are you?** window appears. Enter the requested information and click **Forward**.
- The **Ready to install** window appears. Verify that the language, layout, location, and personal information are correct and click **Install**.
- The installation wizard begins and when the installation wizard has finished, the "Installation complete" window appears. Click **Restart now** to restart your computer. Ubuntu is now installed.

9 Revision History

Date	Version numbers	Changes in this manual
09-21-2012	Quickstart L-781e_1	First draft
12-05-2012	Quickstart L-781e_2	Remove "Getting more involved" chapter Fix some mistakes
03-12-2013	Quickstart L-781e_3	Add description to flash barebox Add description to flash root filesystem Fix some mistakes