PHYTEC

Linux-Kit

phyCORE-OMAP4

Quickstart Instructions

Using Eclipse and the GNU Cross Development Toolchain

Note: The PHYTEC Linux-phyCORE-OMAP4-Disc includes the electronic version of the English phyCORE-OMAP4 Hardware Manual

Edition December 2011

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.

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.

	EUROPE	NORTH AMERICA
Address:	PHYTEC Technologie Holding AG 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 (800) 0749832 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 December 2011

Table of Contents

1 Introduction

5 min

In this QuickStart Instructions Manual you will find general information on the PHYTEC phyCORE-OMAP4-Kit and instructions how to start-up with the phyCORE-OMAP4. You will learn how to...

- *... connect to the target in different ways.*
- *... handle with PTXdist and the OSELAS.BSP-Phytec-phyCORE-PD11.1.0 to build your own images*
- *... working with Eclipse and running example programs on the target with the GNU GCC C/C++ Cross-Development Toolchain*
- *... and much more "nice-to-know" things.*

This first Chapter gives a short introduction about the PHYTEC phyCORE-OMAP4-Kit and this Quickstart. Also you will find general requirements and information to successfully pass the Quickstart.

Please refer to the phyCORE-OMAP4 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 OSELAS.phyCORE-OMAP4-BSP Quickstart manual in which you will find more detailed instructions on how to handle the phyCORE-OMAP4

1.1 Rapid Development Kit Documentation

This "Rapid Development Kit" (RDK) includes the following electronic documentation on the enclosed "PHYTEC Linux-phyCORE-OMAP4-Disc" under *PHYTEC/Documentations* or if you are in the Live Environment under */opt/PHYTEC_Tools/Documentation*.

- The PHYTEC phyCORE-OMAP4 Hardware Manual
- The OSELAS.phyCORE-BSP Quickstart Manual
- These Quickstart Instructions

1.2 Professional Support Packages Available

This Kit comes with free installation support. 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 to 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 an overview in how make an exposure with the phyCORE-OMAP4. 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 phyCORE-OMAP4 platform.
2. The *"Getting More Involved"* section provides step-by-step instructions on how to configure, build and install a new kernel and file system.
3. The *"Programming with Eclipse"* section explains how to modify an example application, create and build a new project, and copy programs to the phyCORE-OMAP4 using Eclipse with the C/C++ and QT-Plug in.
4. The *"Debugging"* section provides information on how to debug an application with the Eclipse debugging interface.
5. In the Appendix you will find how to install Linux on the target, install the Live DVD on your system and 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:

italic	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:

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

1.5 System Requirements

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

- The PHYTEC phyCORE-OMAP4 (OMAP4)
- 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-phyCORE-OMAP4-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-phyCORE-OMAP4-Kit-DVD

There is a bootable modified *Ubuntu* distribution on the Linux-phyCORE-OMAP4-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-phyCORE-OMAP4-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 *10.4.3* which 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 phyCORE-OMAP4 platform for the first time.

2.1 Some notes before starting

As mentioned in the beginning of this Quickstart the Linux-phyCORE-OMAP4-Kit-DVD is an Ubuntu Live DVD with some modifications. This means that you have the freedom to choose if you will 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 in the view of the live-environment to simplify your decision:

Live-environment vs. Installation	
Pro	**Contra**
• 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	• 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. Whatever 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-phyCORE-OMAP4-Kit-DVD and boot from the DVD-drive.

When the system has completely booted you will see the following screen:

In this window you can choose your desired language and you can also choose between installing the operating system and 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 10.4.3" 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.

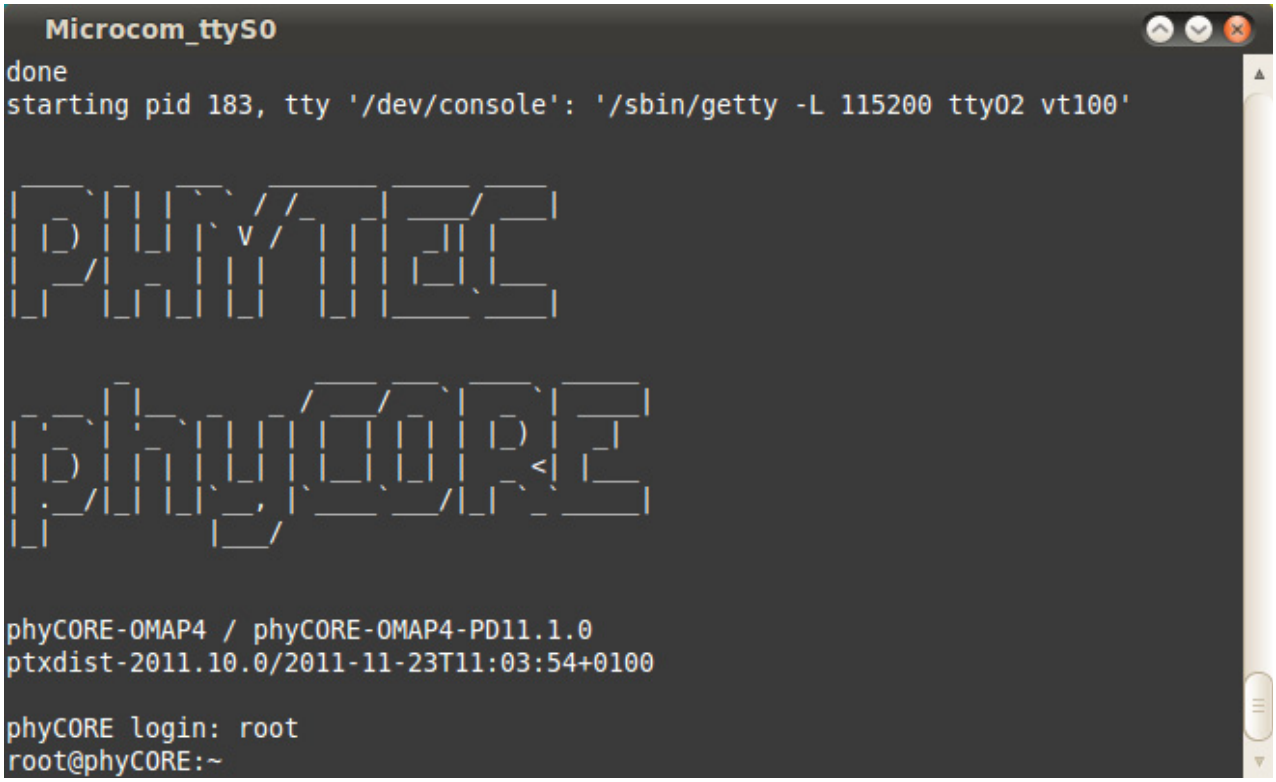
- Connect the serial cable to the UART1 (connector P1, TOP) port on the target and the first serial interface on your host.

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 on the target and to the appropriate network card of your host.

- Click the *Microcom* 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 similiar to the following:



- 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 phyCORE-OMAP4" 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. The kernel will then install the root file system, which is also located in the target's flash. The root file system uses the *Journaling Flash File System, Version 2*.
JFFS2 is the successor and a complete rewrite of the original JFFS by Red Hat. As its name implies, the JFFS2 implements a journaling file system on the memory technology device (MTD) it manages. JFFS2 does not attempt to provide a translation layer that enables the use of a traditional file system with the device. Instead, it implements a log-structured file system directly on the MTD. The file system structure itself is recreated in RAM at installation time by JFFS2 through a scan of the MTD's log content.
In addition to it's log-structured file system, JFFS2 implements wear leveling and data compression on the MTD it manages, while providing power-down reliability. JFFS2 can gracefully restart, and is capable of restoring a file system's content, without requiring outside intervention regardless of power failures.

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 ports 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.

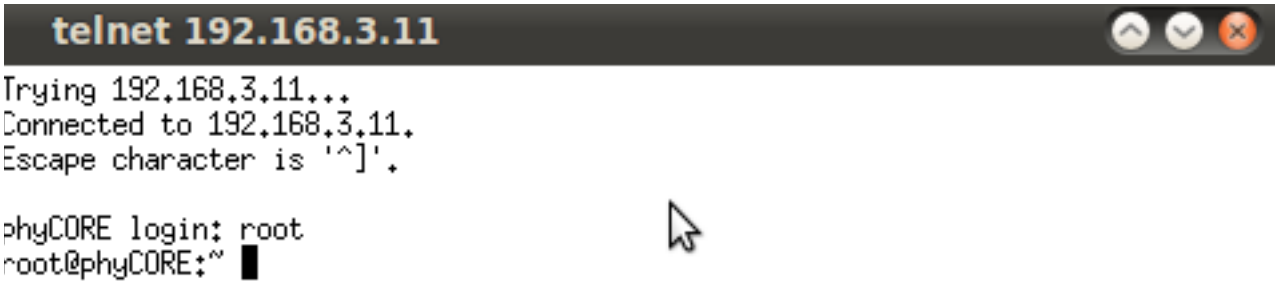
We recommend that the host PC is not connected to any other network. The target and host will be connected with a cross-over cable via a peer-to-peer connection. If your host is part of a company's network, we recommend disconnecting your host from such a network.

- 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.

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

A new window with a connection to the target opens.



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

- Close the window.

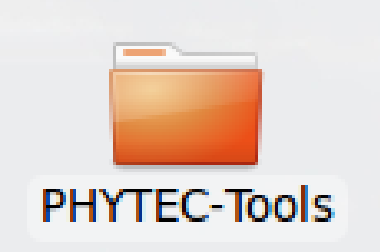
Troubleshooting:
If you don't see the user login, check the Ethernet connection between the target and the host. If you have installed more than one network card on your host, be sure to connect the cable to the network card which you have configured with the IP address **192.168.3.10**.
If you do not 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 phyCORE-OMAP4 from your host.

2.4 Copying an Example to the Target

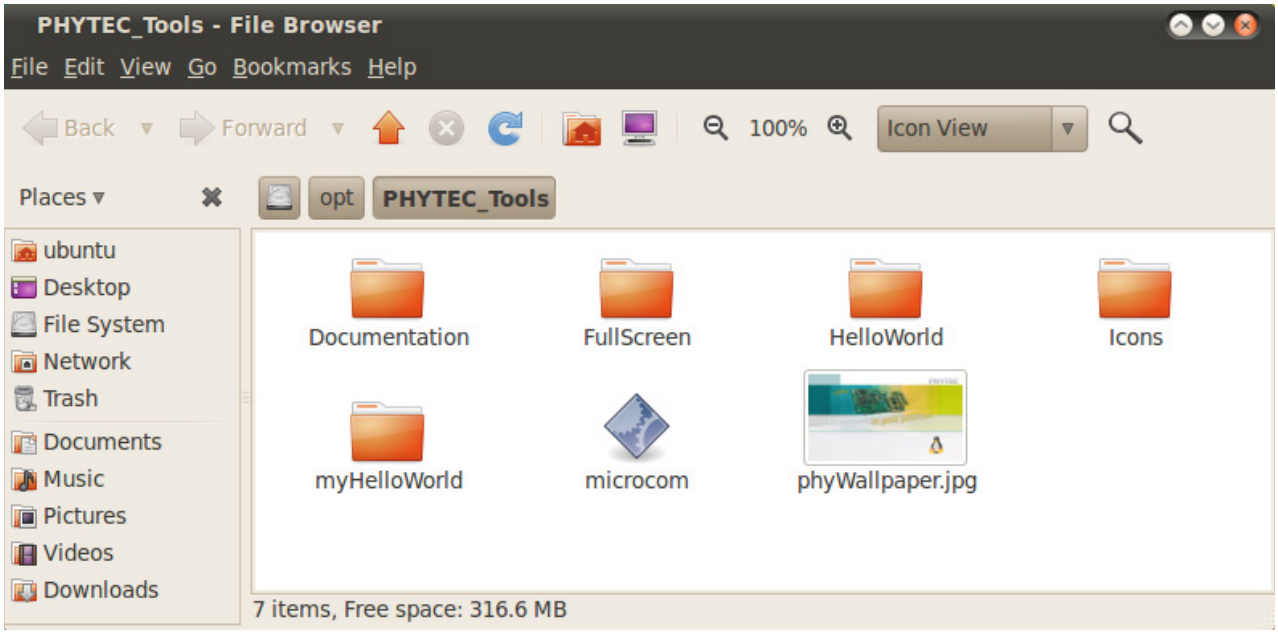
In this section you will learn how to copy an example program to the target using the FTP protocol with the *Nautilus* file browser. After that you will execute an example on the target. At the end of this passage you can find some information on how to copy and execute a file on the target using the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

2.4.1 Copying a Program to the Target



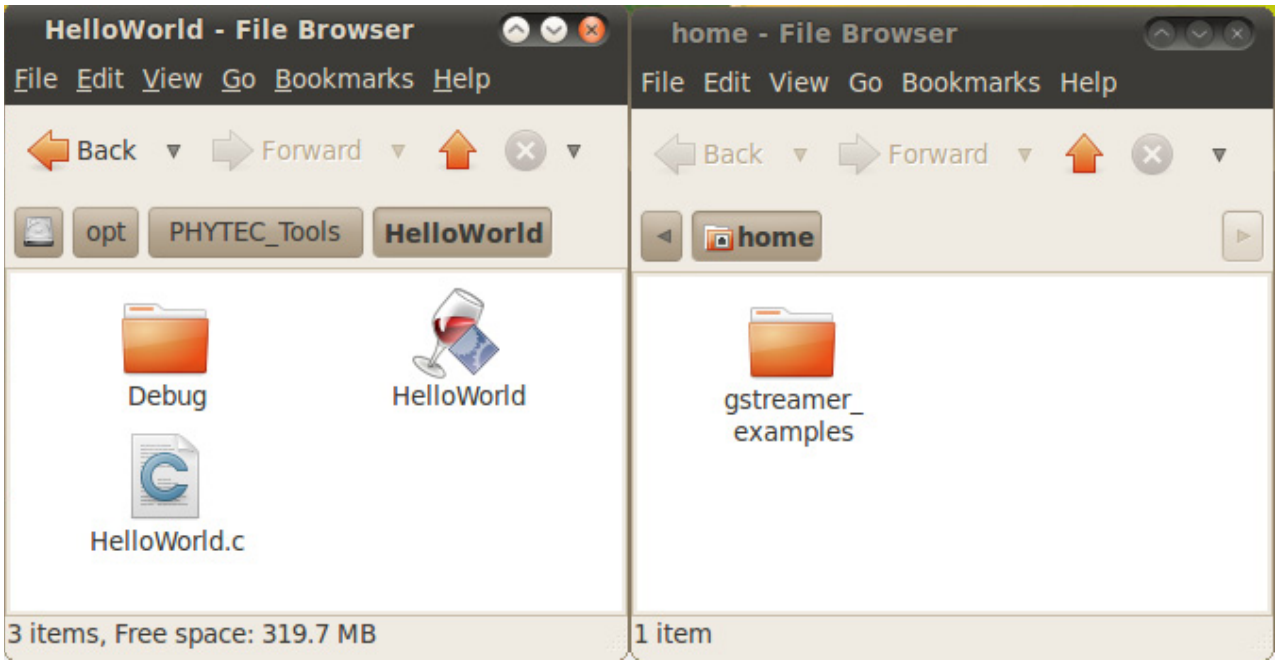
- First click the *PHYTEC-Tools* icon on your *GNOME* desktop.

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

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

- 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 phyCORE-OMAP4!

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 phyCORE-OMAP4 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@phyCORE:~>* 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 phyCORE-OMAP4 before.

The program starts and you should see the following output:

Welcome to the World of the phyCORE-OMAP4!

- 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 root@192.168.3.11/ HelloWorld
  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 Getting more involved

70 min

In this chapter you will go through some continuative topics. First you will configure and compile your own kernel and root file system. With the help of PTXdist you can add additional features, or disable them if they are not needed. After compiling the new images, you will learn how to write the newly created kernel and root file system into target's flash memory and how to start from.

Then you will start working with the Eclipse platform using the C/C++ Development Tools (CDT) in conjunction with the GCC C/C++ tool chain. You will learn how to configure the Eclipse platform and how to open an existing project. After that you will create your first own project and modify the example's source code.

At the end of this chapter you will execute the program as an external application out of Eclipse. Additionally, you will add your application to the startup configuration of the target so it is automatically started when the phyCORE-OMAP4 boots.

3.1 Working with the Kernel

In this part you will learn how to configure and build a new Linux kernel and a root file system. Then you will configure the kernel and the root file system with the help of *PTXdist*, a tool to build the Board Support Package and to create your own image. After the configuration you will create your own image.

To configure your images the following files are already pre-installed:

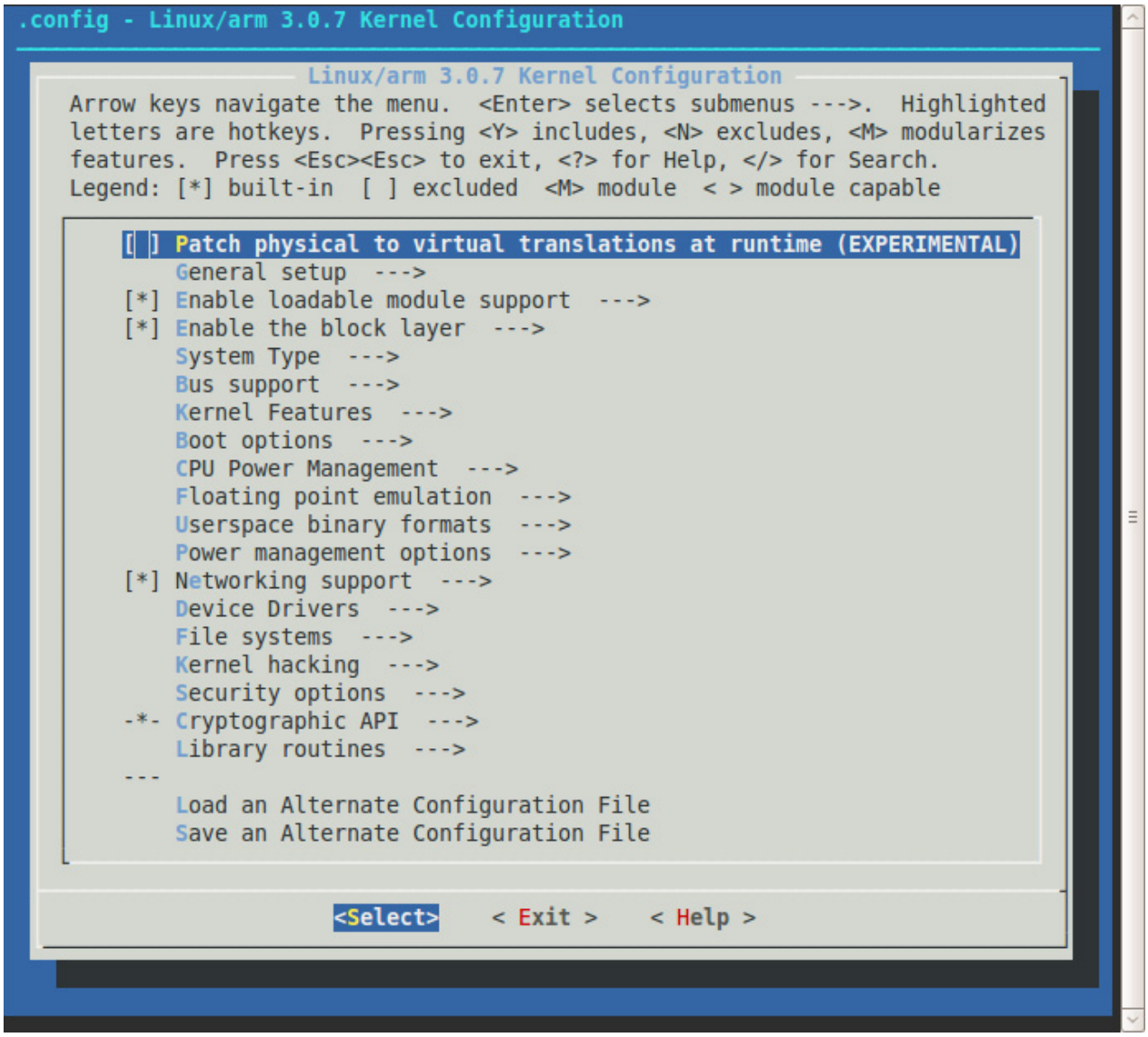
- ptxdist-XXXX.XX.X.tgz (Tool from our partner *Pengutronix*, that configures and compiles BSPs)
- OSELAS.BSP-Phytec-phyCORE-PDXX.X.X.tar.gz (BSP for the phyCORE-OMAP4)
- arm-cortexa9-linux-gnueabi.tar.gz (ready-to-use toolchain)

All files are also downloadable from our ftp server or you can find it on our Linux-phyCORE-OMAP4-Kit-DVD under */PHYTEC/BSP/* . If you want other versions check our ftp server too.
ftp://ftp.phytec.de/

3.2 Working with the filesystem

Let's start by opening a new terminal if not already open and change into the directory of the pre-installed BSP:

- Click the terminal icon on your desktop.
- Type the following command to change to the BSP-directory:
  cd /opt/PHYTEC_BSPs/OSELAS.BSP-Phytec-phyCORE-PD*
- Type the following to show the deselected/selected kernel features:
  ptxdist kernelconfig
  You should see the following output on the terminal:

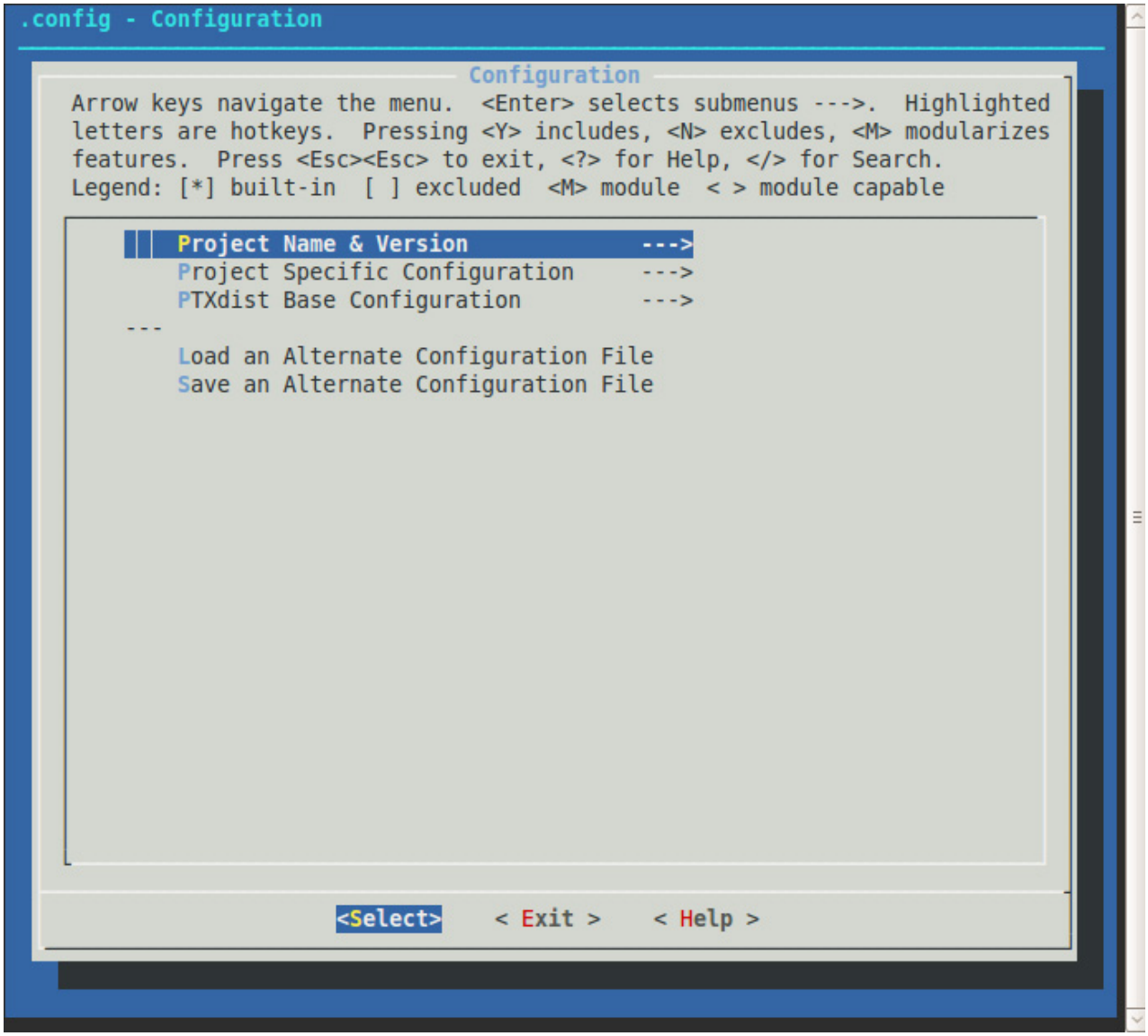


- Leave the menu by selecting < *Exit* > at the bottom with the arrow right key and press **Enter**.

You can find more information about *PTXdist* and the toolchain in our "OSELAS.phyCORE-BSP_Documentation" under /opt/PHYTEC_Tools/Documentation/ or on the Linux-phyCORE-OMAP4-Kit-DVD under */PHYTEC/Documentation*.

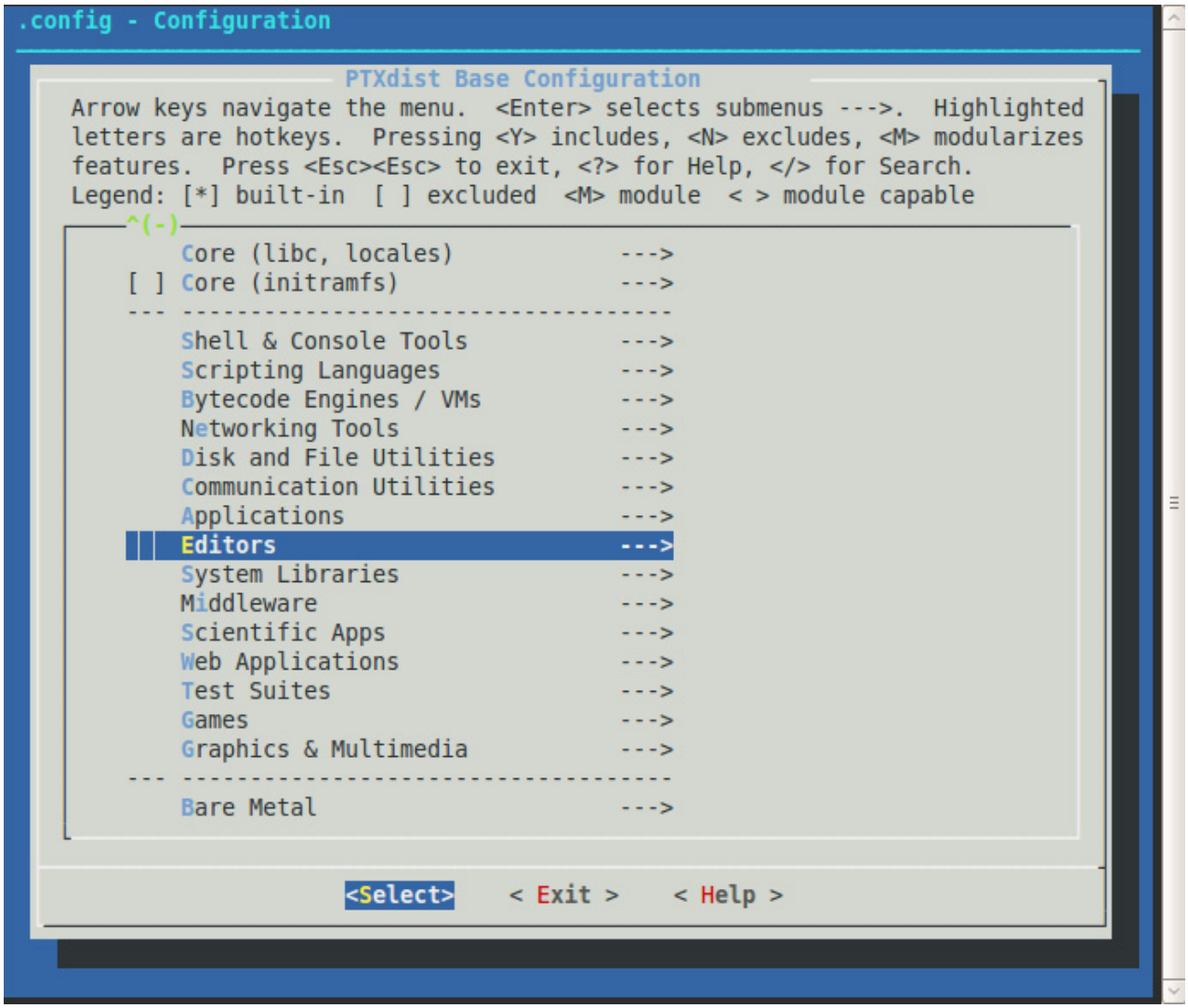
- Now let's take a look into the user space and add a new program in the root file system. Type the following:
  ptxdist menuconfig

You should see the following output:

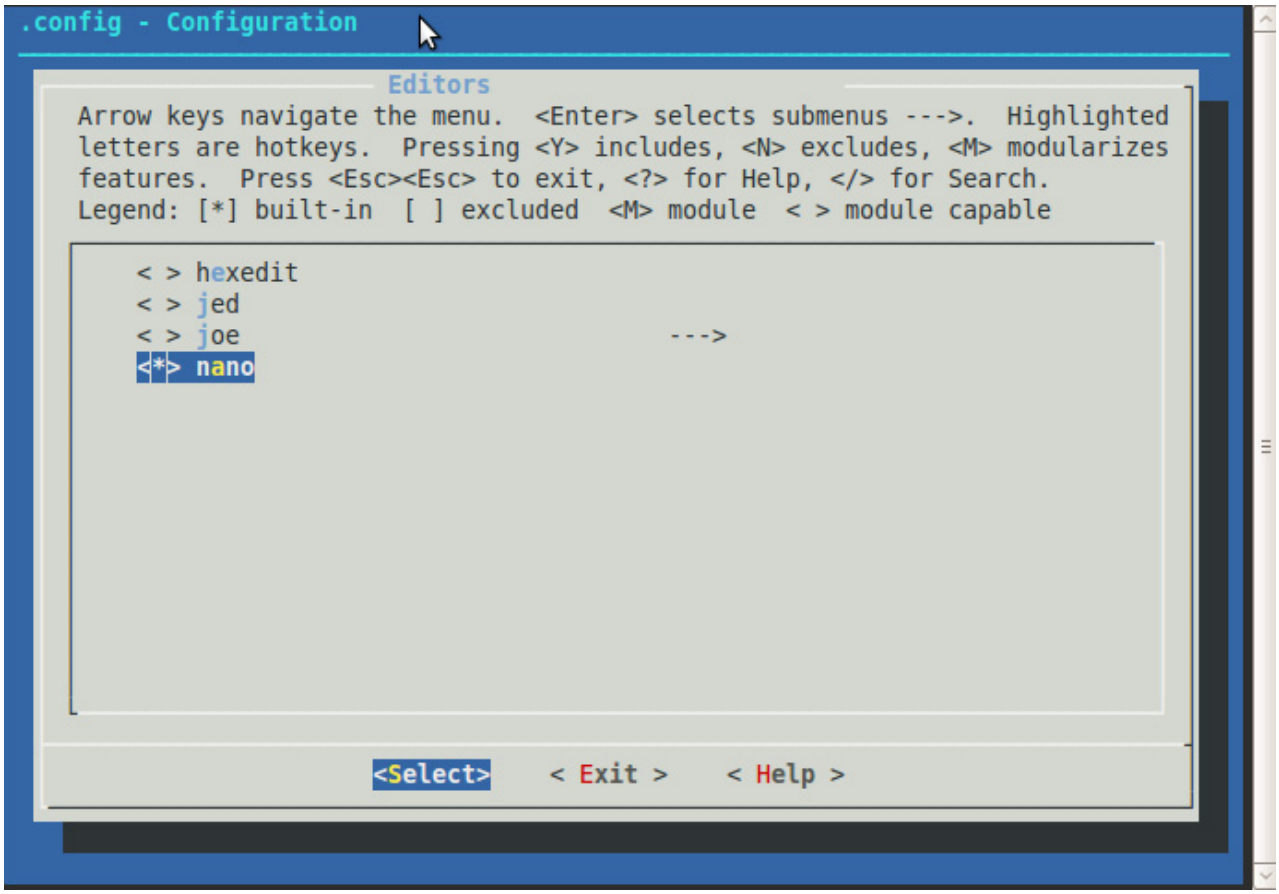


- For example we add *nano* to the user space which is another editor. With the help of the up and down arrow key we select *PTXdist Base Configuration* and press **Enter**.

- Now navigate to *Editors* and press **Enter** again.

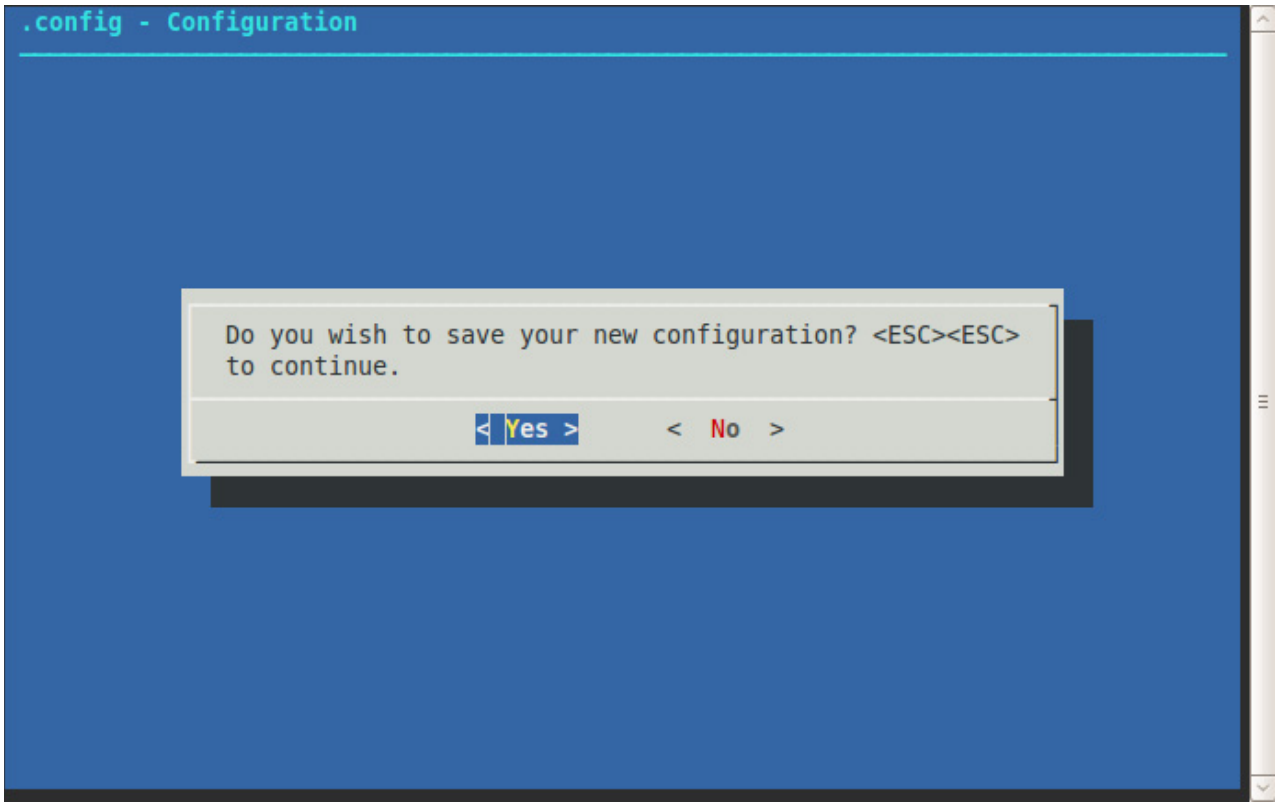


- After that, we navigate to *nano* in the list of editors. By pressing **Y** we selected nano as "*built-in*".



- Leave the menu by selecting < *Exit* > at the bottom with the arrow right key and press **Enter**. Repeat this until a question appears to save the new configuration.

- Select < *Yes* > and press **Enter** to save the configuration.



- Now we can start building the configured BSP. In the terminal type:
  ptxdist go
- After it is finished we can finally you can create the root file system by calling:
  ptxdist images

You will find the kernel named *linuximage* and the root file system named *root.jffs2* in the directory *platform-phyCORE-OMAP4/images/*.

In this section you will build your own kernel and root file system and learn how to configure and compile it. Now you can add new features to your kernel or remove features you do not need.

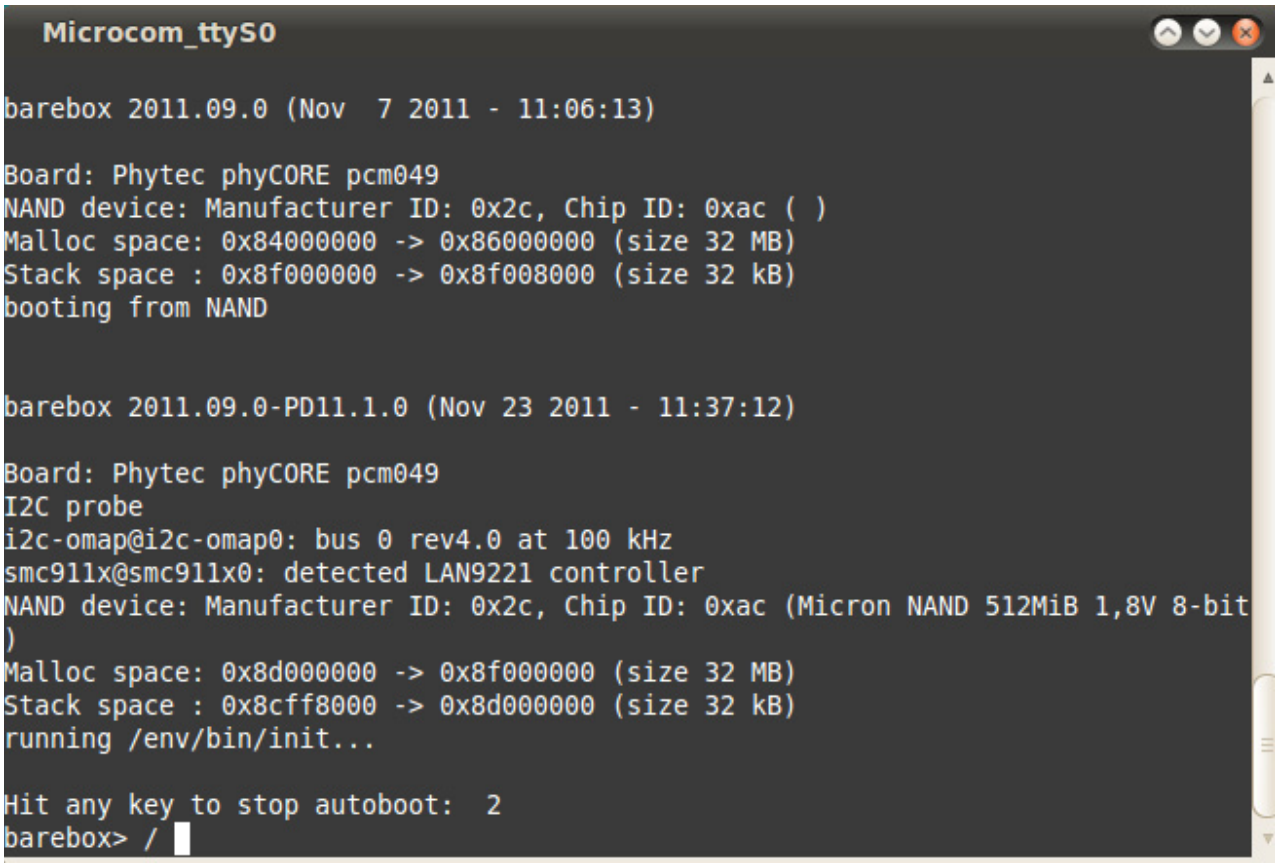
3.3 Writing the Images into the Target´s Flash

In this section you will find a description on how to write the newly created images into the phyCORE-OMAP4's flash memory. Before the images can be written into the flash, the target will have to download them from a TFTP server. This will be done from the command line of the boot loader. The images will be copied into target's RAM. Then you will have to erase the part of the flash where you want to copy the images to. Finally the images are written from the RAM to the flash.

In the default configuration you will find four partitions on the target: The first partition contains the boot loader, the second is used to store the boot loader settings, the third partition stores the Linux kernel, and the fourth contains the root file system.

You should never erase the Barebox partition. If this partition is erased, you won't be able to start your target anymore. In such a case please contact our support department.

- First open a new terminal window if it is not opened yet. Then change to the directory /opt/PHYTEC_BSPs/OSELAS.BSP-Phytec-phyCORE-PD*.
- Copy the new images to the */tftpboot* directory and exit:
  cd platform-phyCORE-OMAP4/images
  cp linuximage /tftpboot
  cp root.jffs2 /tftpboot
  exit
- Open Microcom and press the RESET button on the target.
  You will see the output *"Hit any key to stop autoboot"*.
- Press any key to stop autoboot.

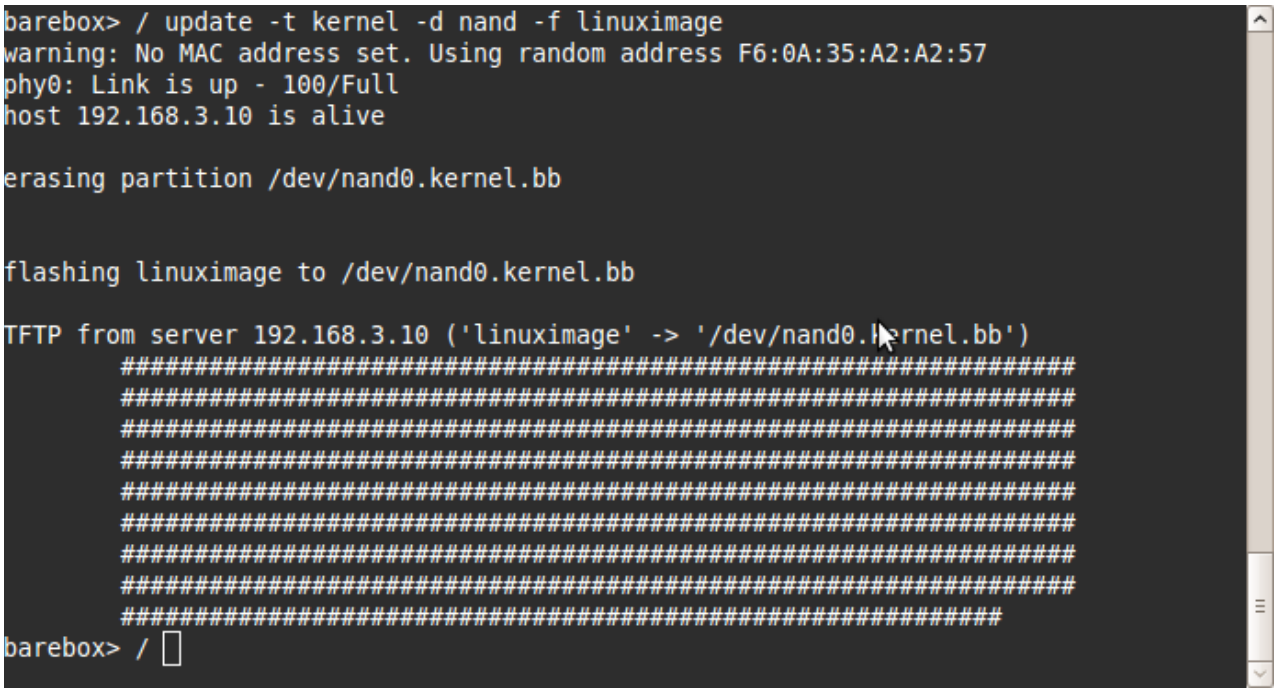


- Type the following command to check your Barebox settings:
  edit /env/config
  You will see the configuration file which contains Barebox's environment variables.
- Make sure that the following values are set in the configuration file:
  eth0.ipaddr=192.168.3.11
  eth0.netmask=255.255.255.0
  eth0.serverip=192.168.3.10

- Type **CTRL-D** to save the settings temporarily to the file.
- If you made any changes to the Barebox environment, type **save** to write these changes to the Barebox environment partition, and then press the target's RESET button. The phyCORE-OMAP4 reboots with the new settings applied. Again, press any key to stop autoboot.

Now we download the images from the TFTP server to the target's RAM, then we erase the required flash and write the images from the RAM into the flash. It sounds like a lot of work but we can do this with one command: *update*.

- Type **update -t kernel -d nand -f linuximage** to download the kernel using TFTP and write it into the target's flash. The copy process can take up to a minute.



- Type **update -t rootfs -d nand -f root.jffs2** to download the root file system using TFTP and write it into the target's flash. The copy process can take up to several minutes.
- Type **boot** to boot the phyCORE-OMAP4 with the new kernel and root file system.
- After the target has successfully finished booting, type **root** to log in.
- Now we can test our new editor *nano* by trying to open a file with it.
  nano /etc/profile.environment
- Close *nano* by pressing **CTRL+X**.
- Close Microcom when nano is closed.

Troubleshooting:
If any problem occurs after writing the kernel or the root file system into the flash memory, you can restore the original kernel (*uImage-pcm049*) and root file system (*root-pcm049.jffs2*) from the */tftpboot* directory.
Type **update -t kernel -d nand -f uImage-pcm049** to restore the kernel and **update -t rootfs -d nand -f root-pcm049.jffs2** to restore the root file system.

In this section you learned how to download images from a tftp server into the RAM of the target. The images have been written from RAM to flash and finally the target was started with the new images.

4 Working with Eclipse

35 min

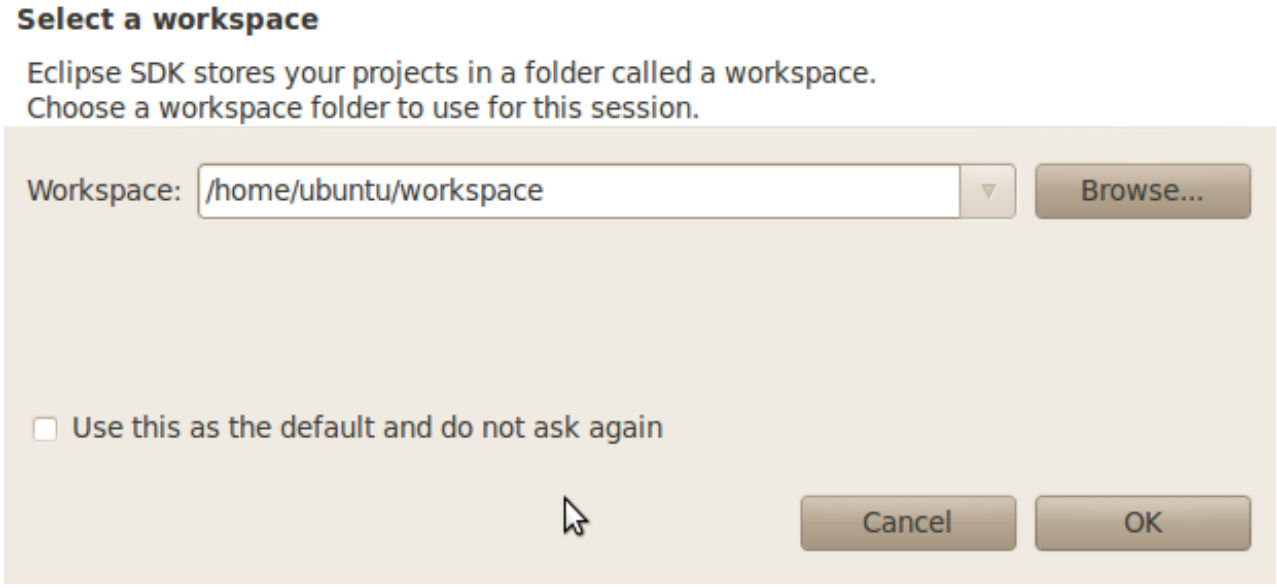
With the help of example projects, we will teach you how to work with eclipse during this chapter. First we take a look on the C programming language and then we show you the QT C++ programming environment. At the end of this chapter we explain how to execute your written programs automatically when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

4.1 Programming in the C/C++ perspective

We are starting with the C/C++ workbench. Therefor 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.

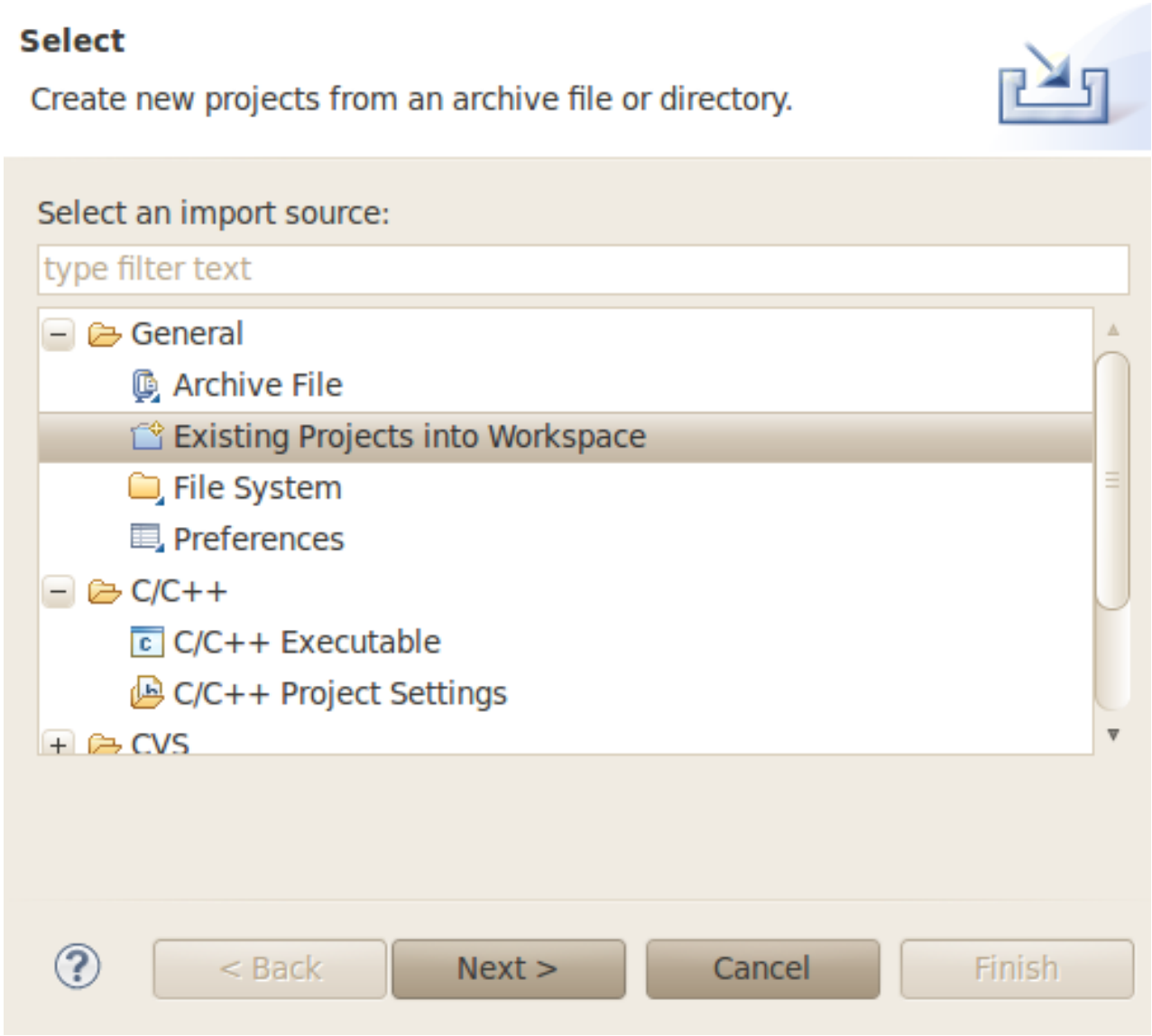
4.1.1 Handle with the demo project

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

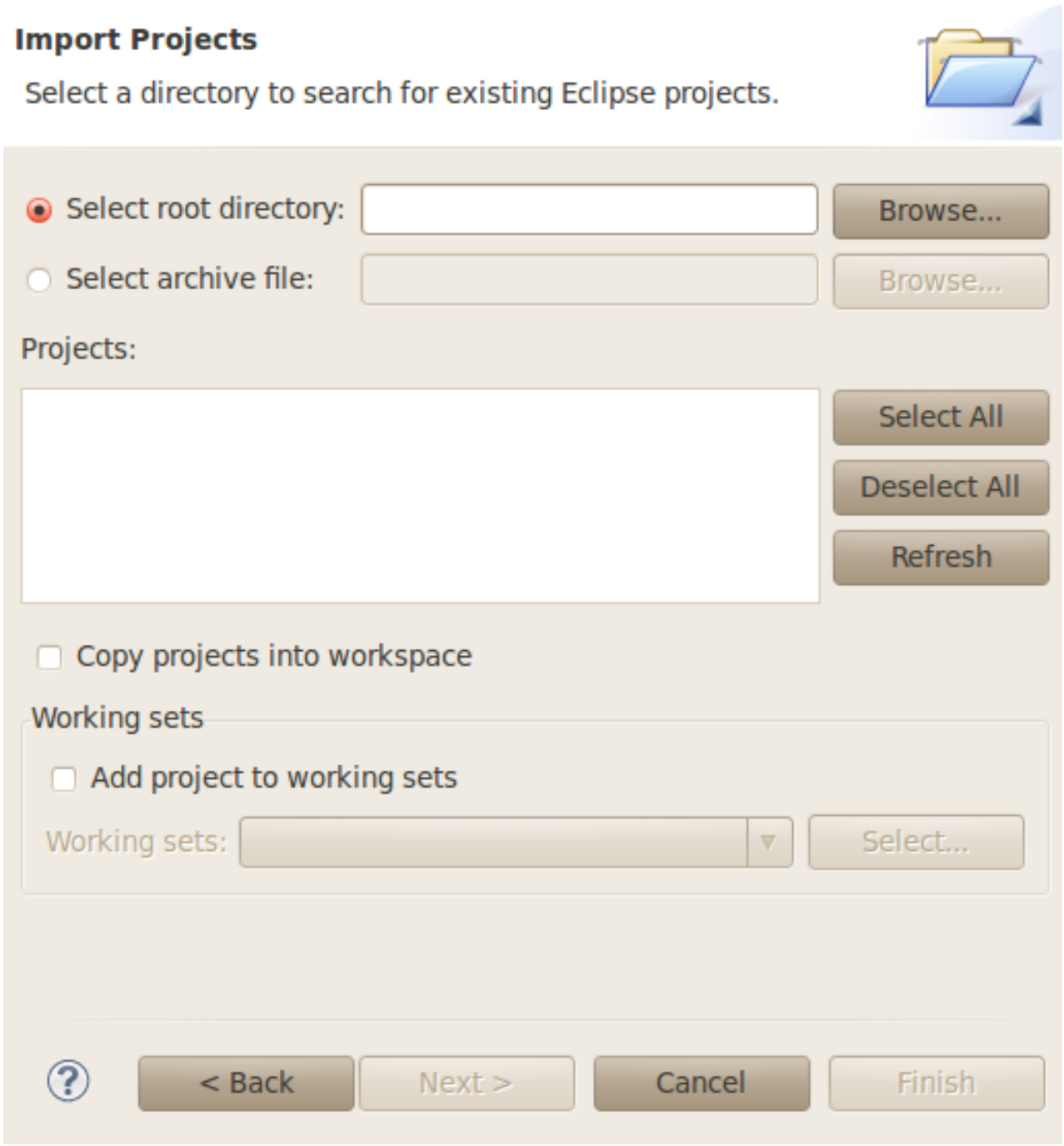


- Confirm the workspace directory with OK.
- Close the "*Welcome to Eclipse*" screen by clicking on the "*Go to the workbench*" - button.

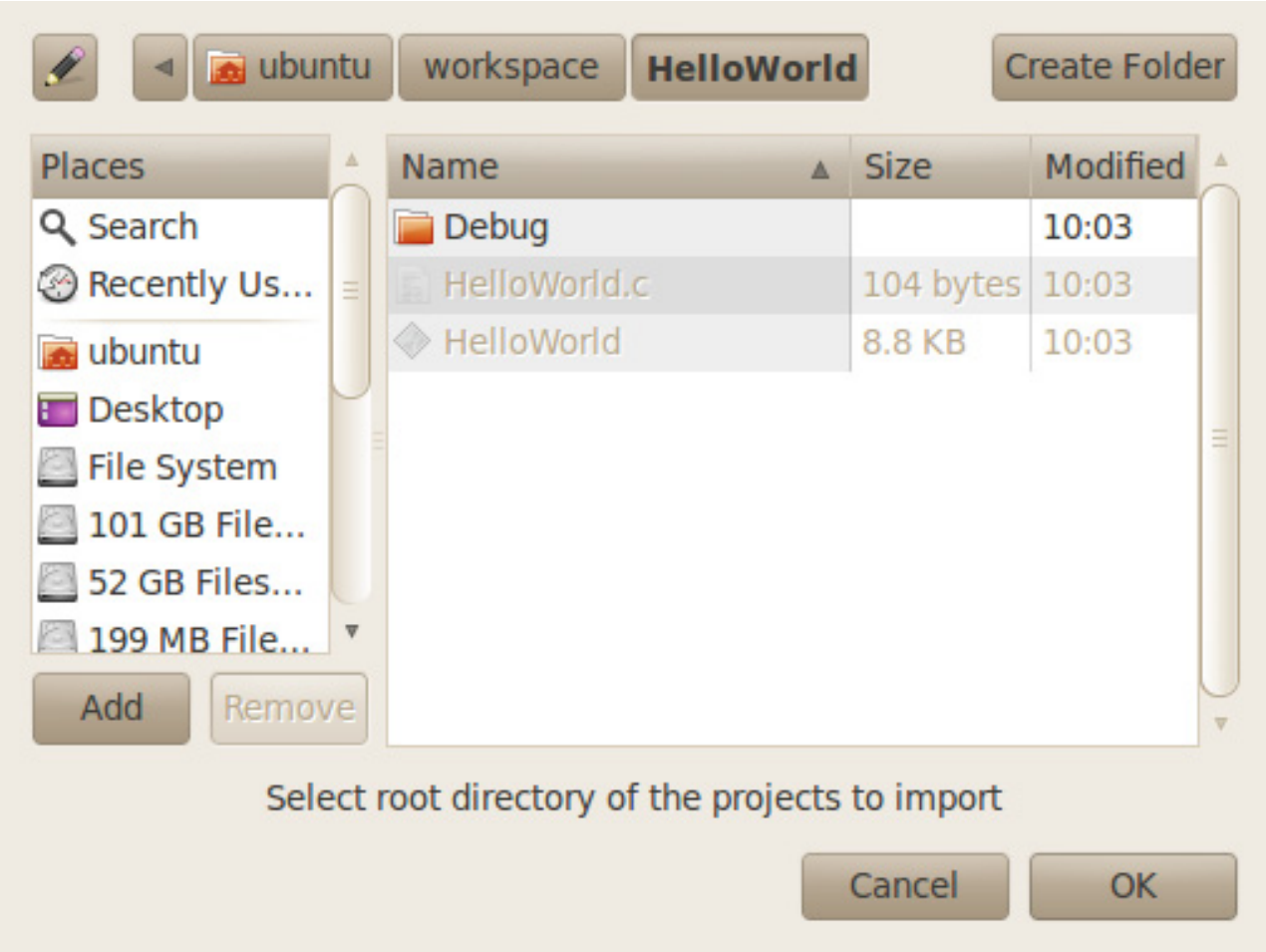
- Select File ► Import from the menu bar.



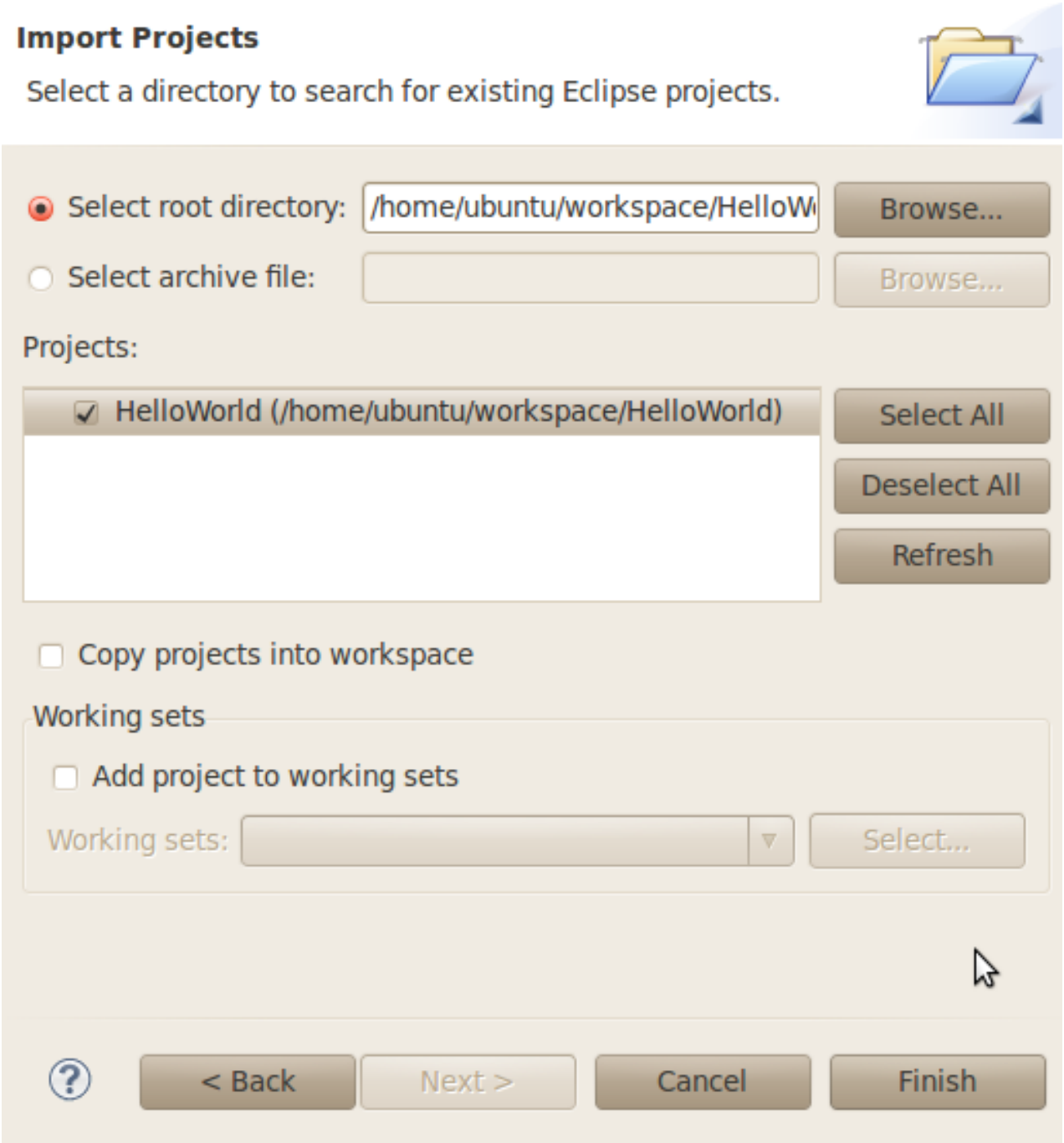
- Select *Existing Projects into Workspace*.
- Click *Next*.



- Select *Browse*.



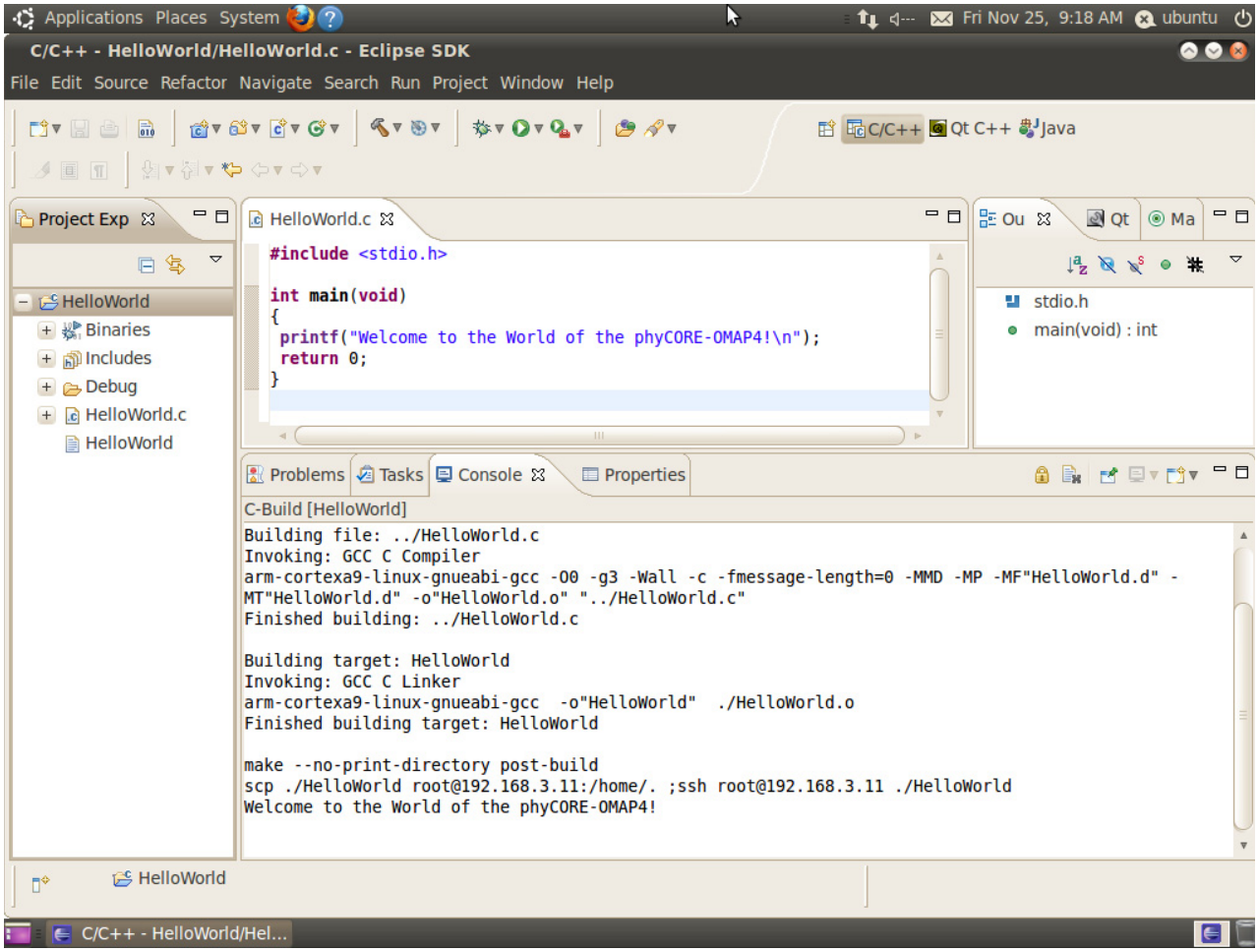
- Double-click the *HelloWorld* directory under */home/ubuntu/workspace/* .
- Click *OK*.



- 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:



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.

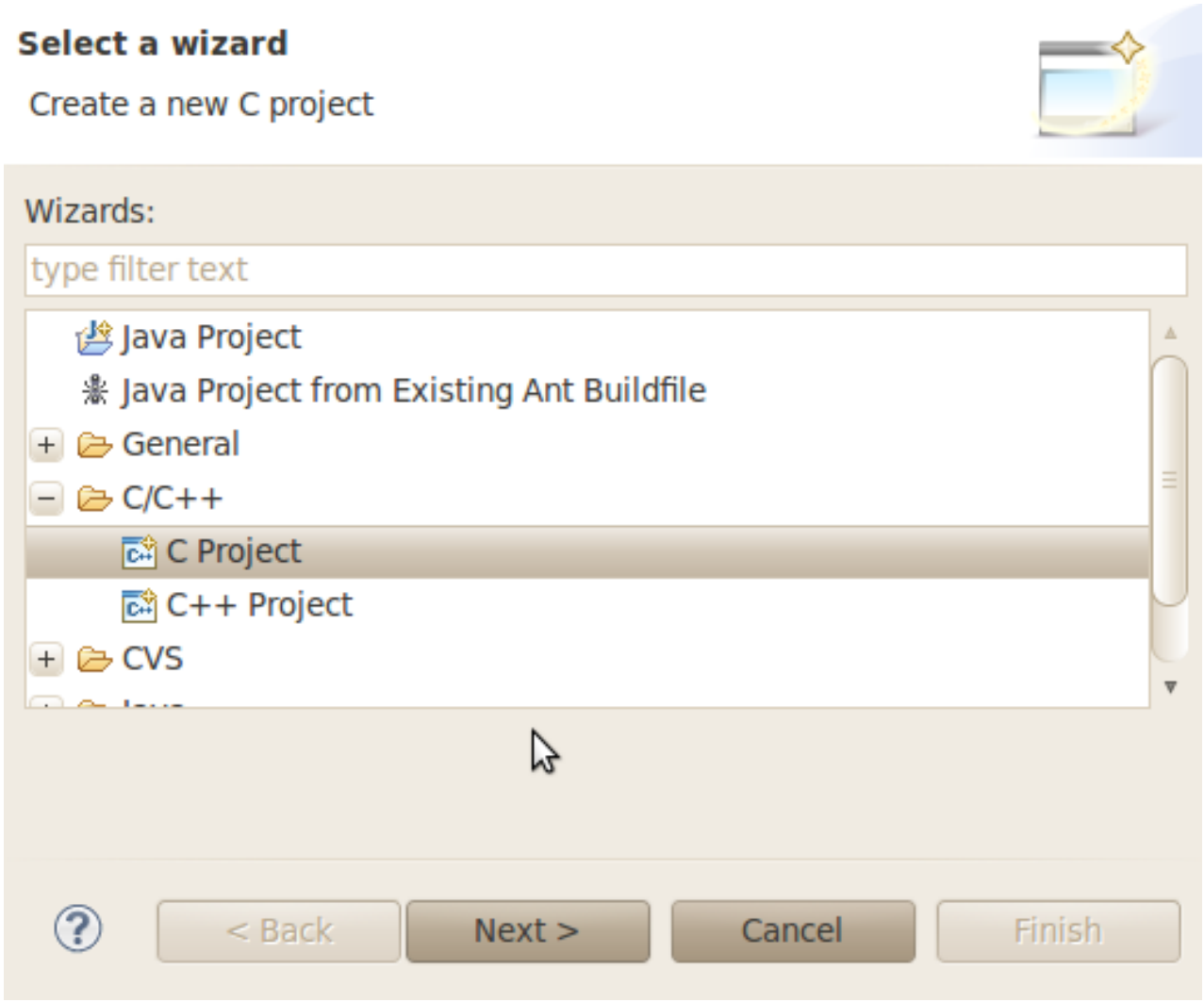
4.1.2 Creating a New Project

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.

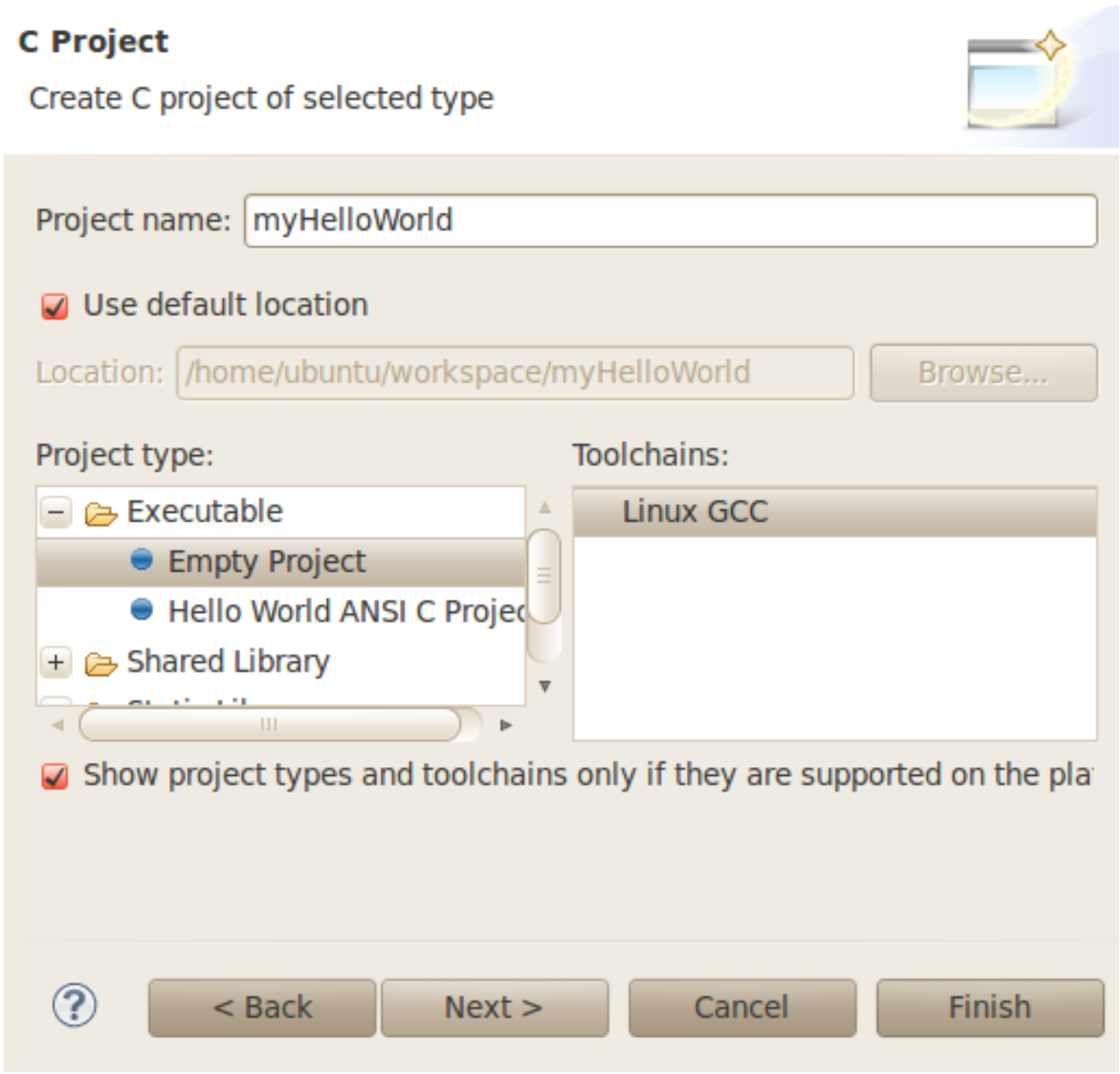
- Open Eclipse if it isn't already opened.

- Select *File* ▶ *New* ▶ *Project* from the menu bar.

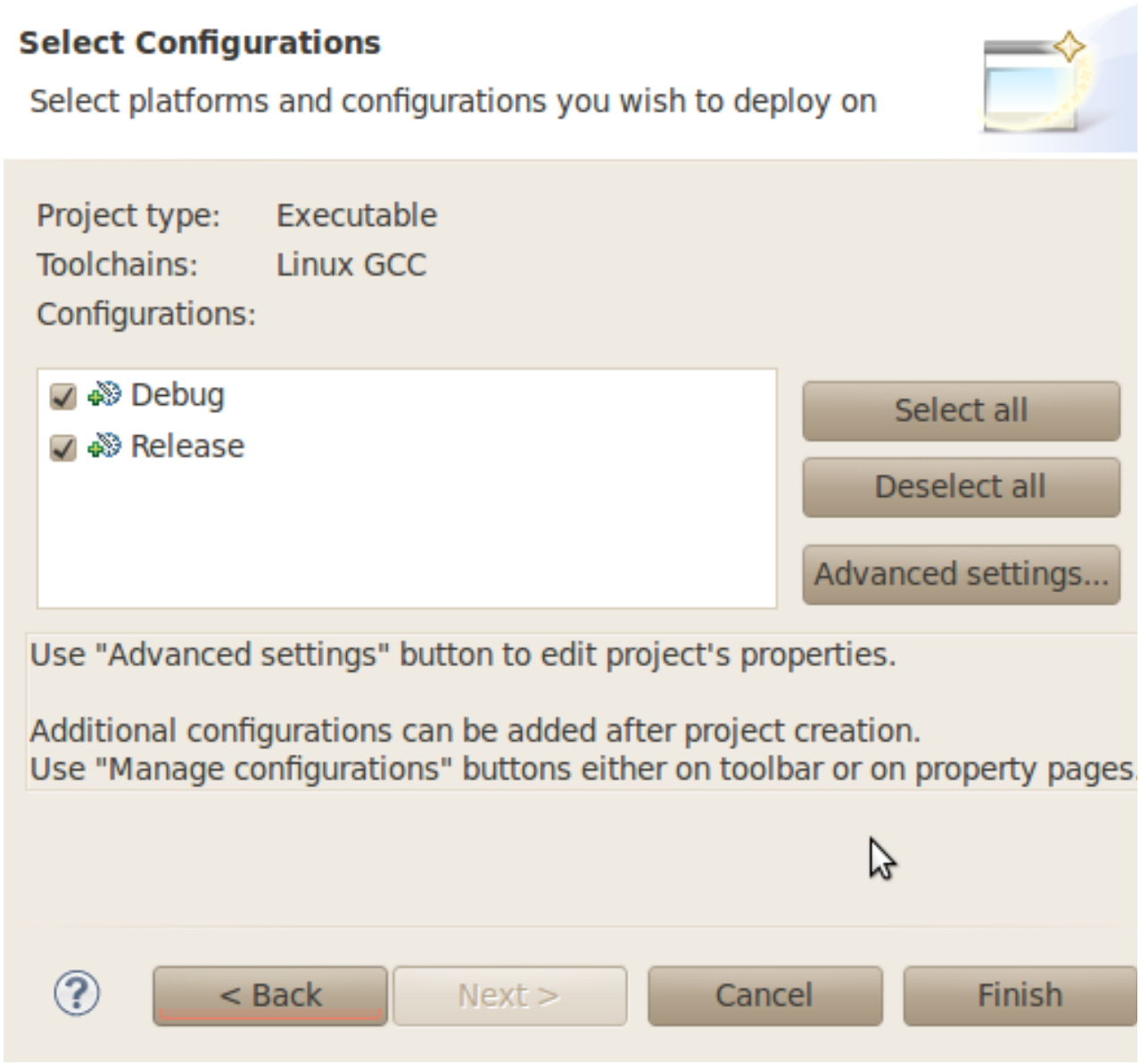
A new dialog opens.



- Select *C Project* and click *Next*.

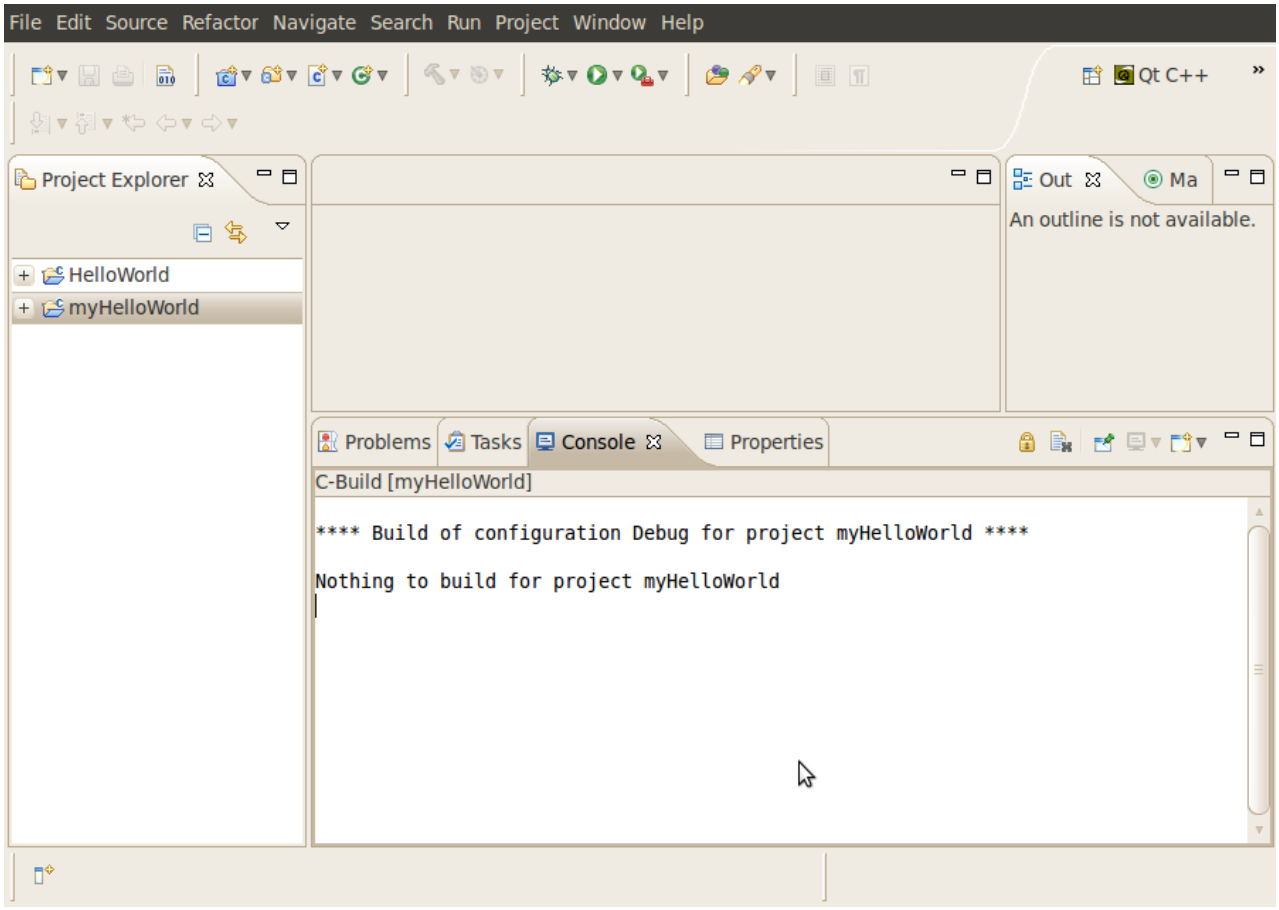


- Enter the project name *myHelloWorld* and click *Next*.

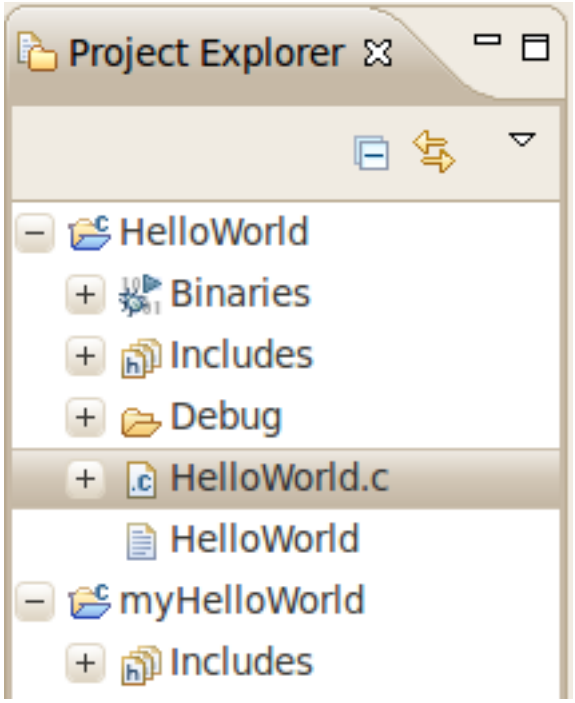


- Click *Finish*.

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.

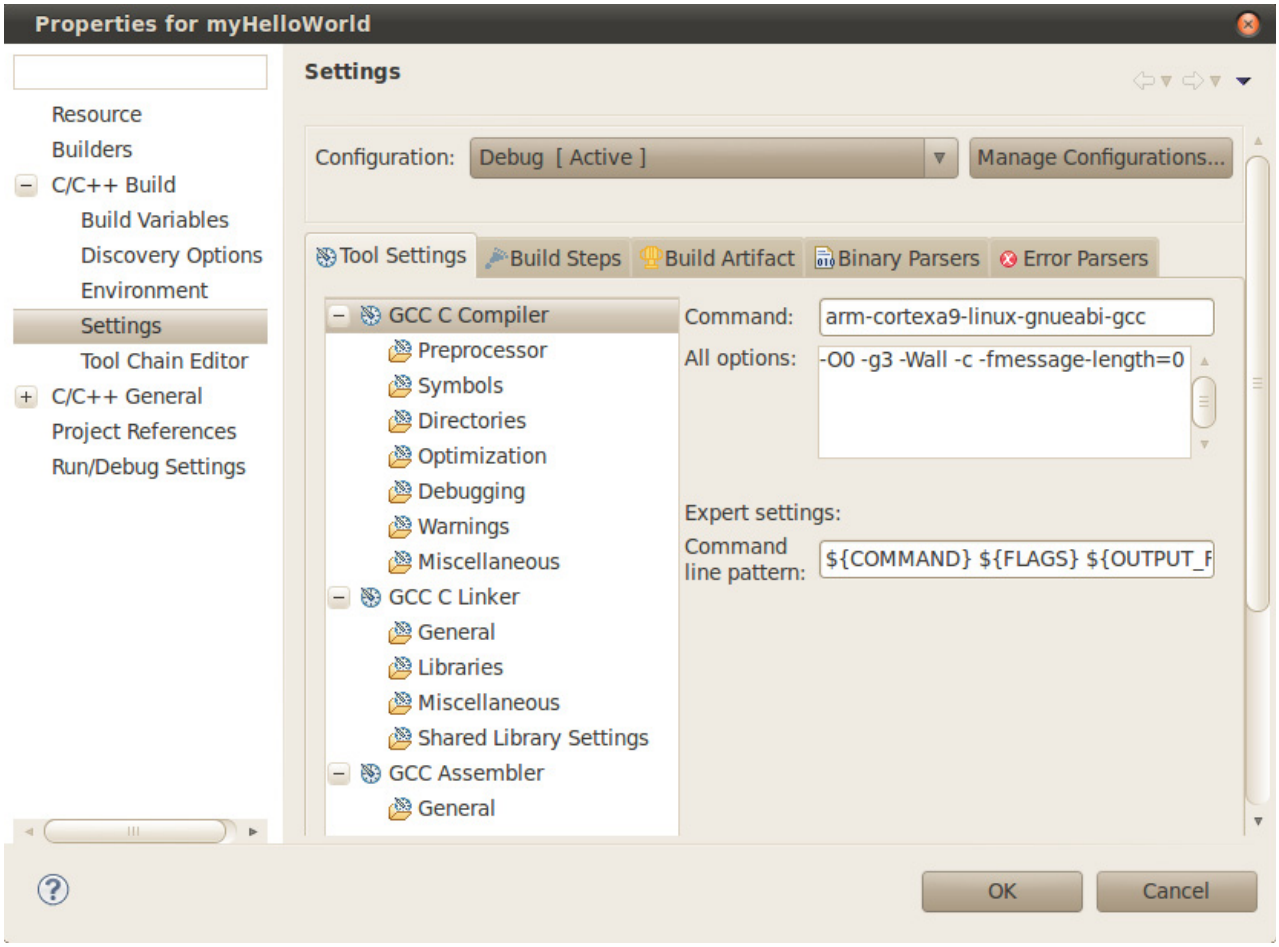
If *Build Automatically* from the *Project* menu is selected, the *HelloWorld* application will now be compiled and created with the standard GCC C/C++ compiler suitable for your host machine. You will find the executable file, which can only be run on your host system, in the *workspace/myHelloWorld/Debug*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

To compile your project for the phyCORE-OMAP4 instead, you will have to use the GNU C/C++ cross compiler.

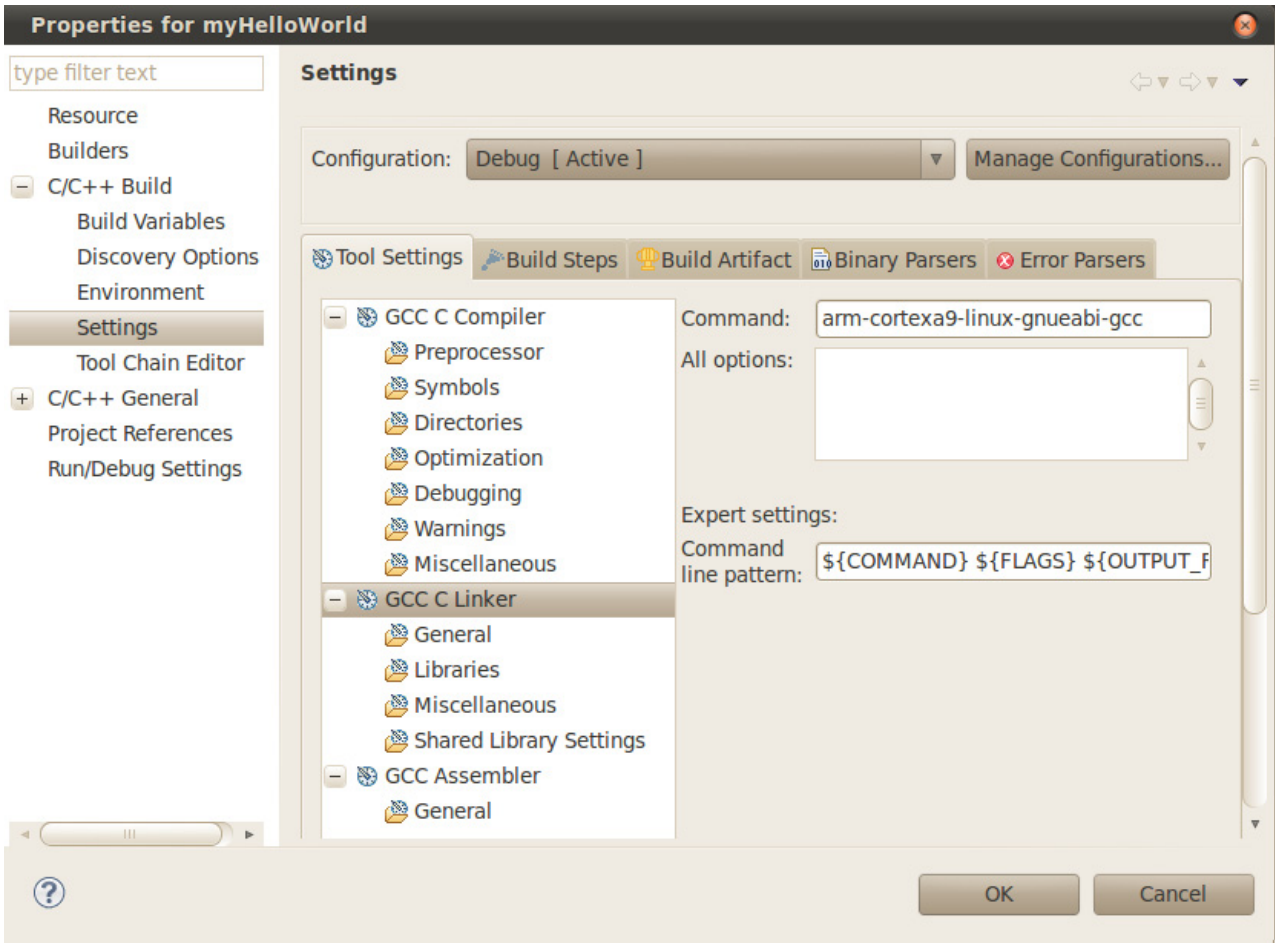
- Right-click the *myHelloWorld* project and choose *Properties*.

The *Properties* dialog appears.

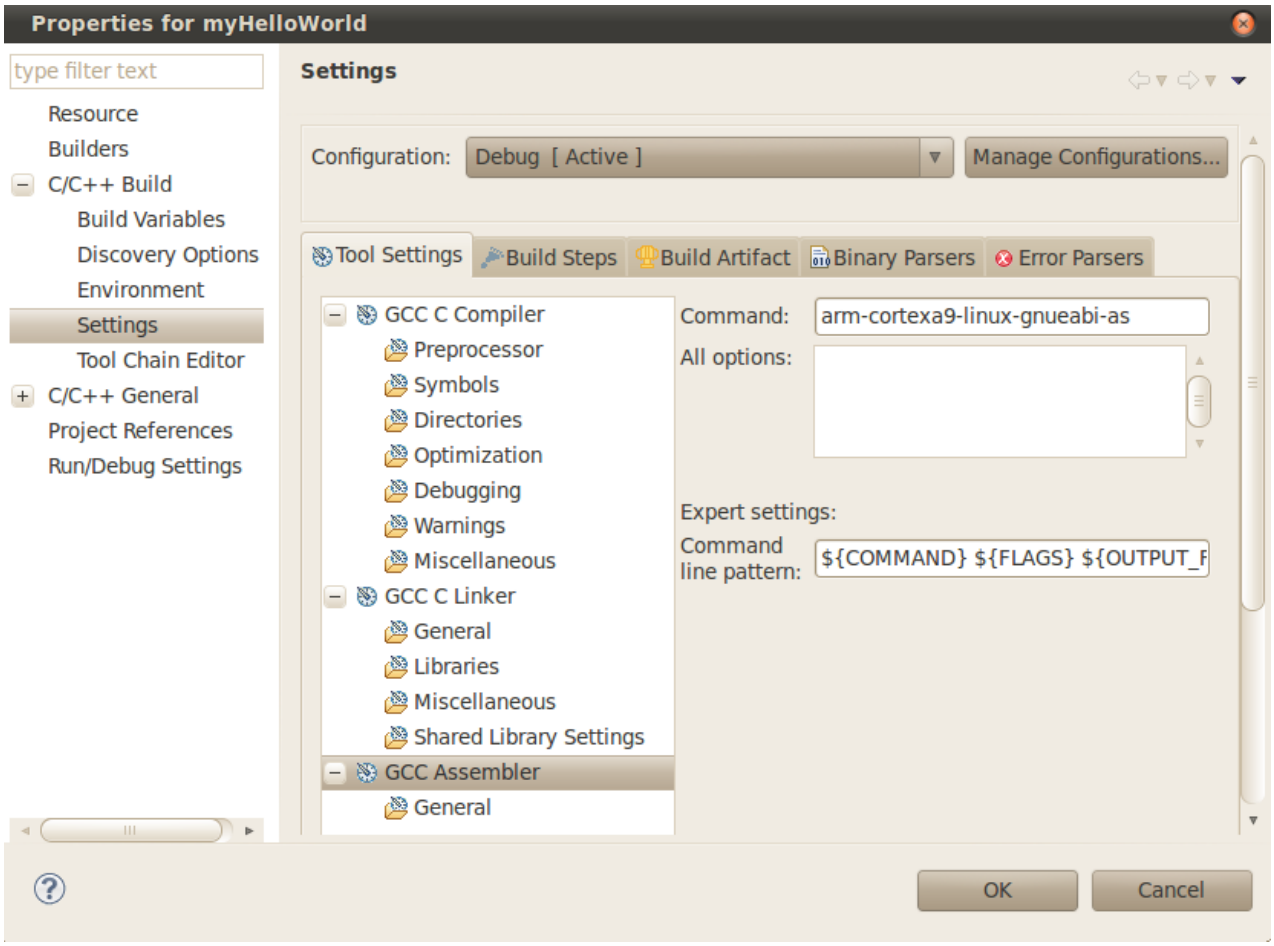
- Select *C/C++ Build*.
- Enter **arm-cortexa9-linux-gnueabi-gcc** into the *Command* input field.



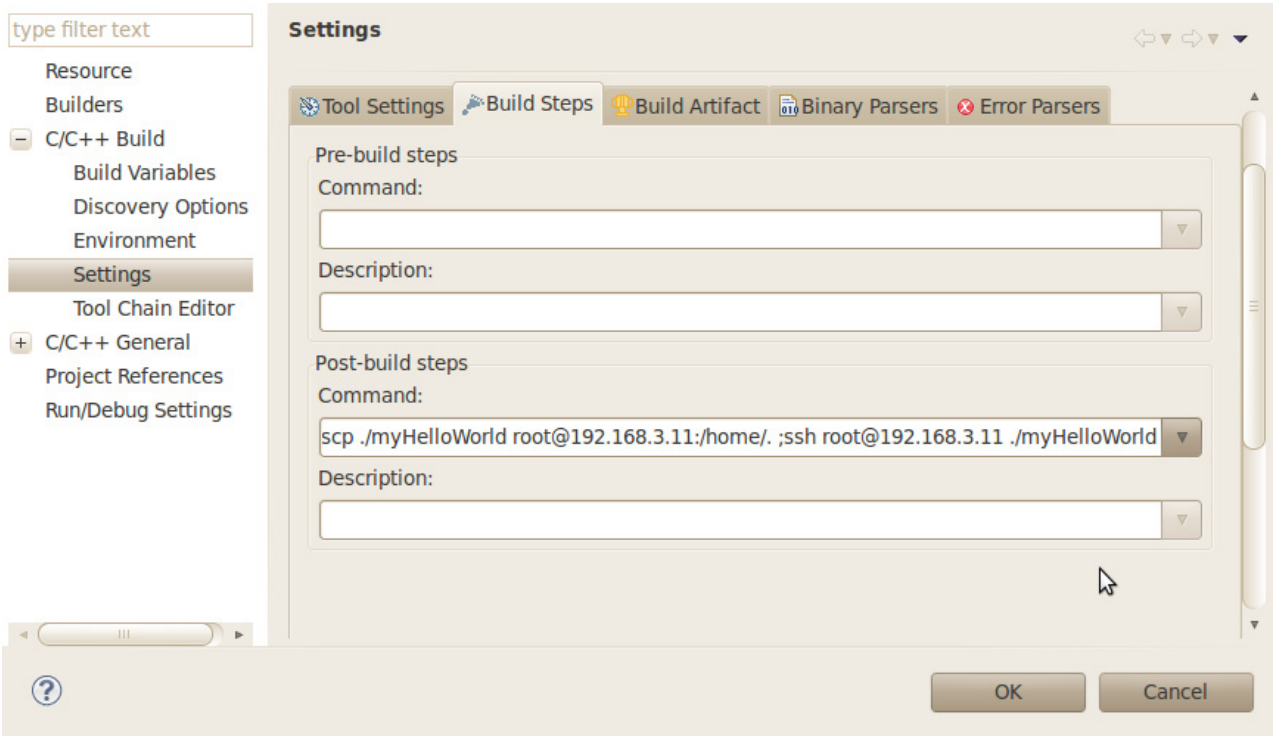
- Select *GCC C Linker*.
- Enter **arm-cortexa9-linux-gnueabi-gcc** into the *Command* input field.



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

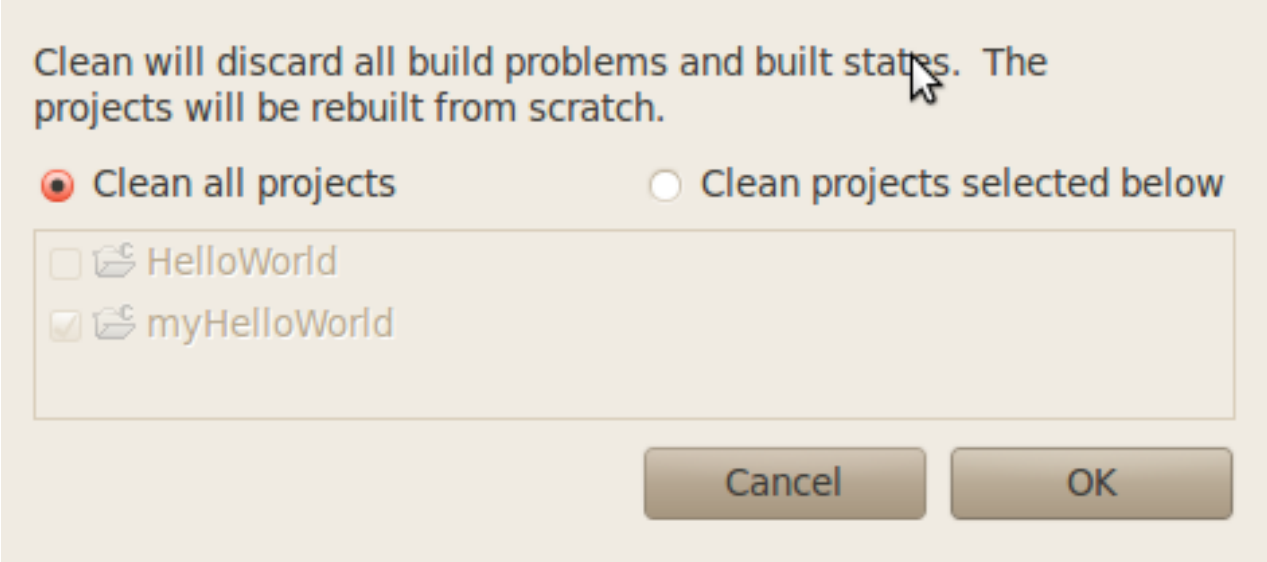


- Select the *Build Steps* tab.
- Enter the following command in the Post-build steps *Command* input field:
  scp ./myHelloWorld root@192.168.3.11:/home/. ;ssh root@192.168.3.11 ./myHelloWorld

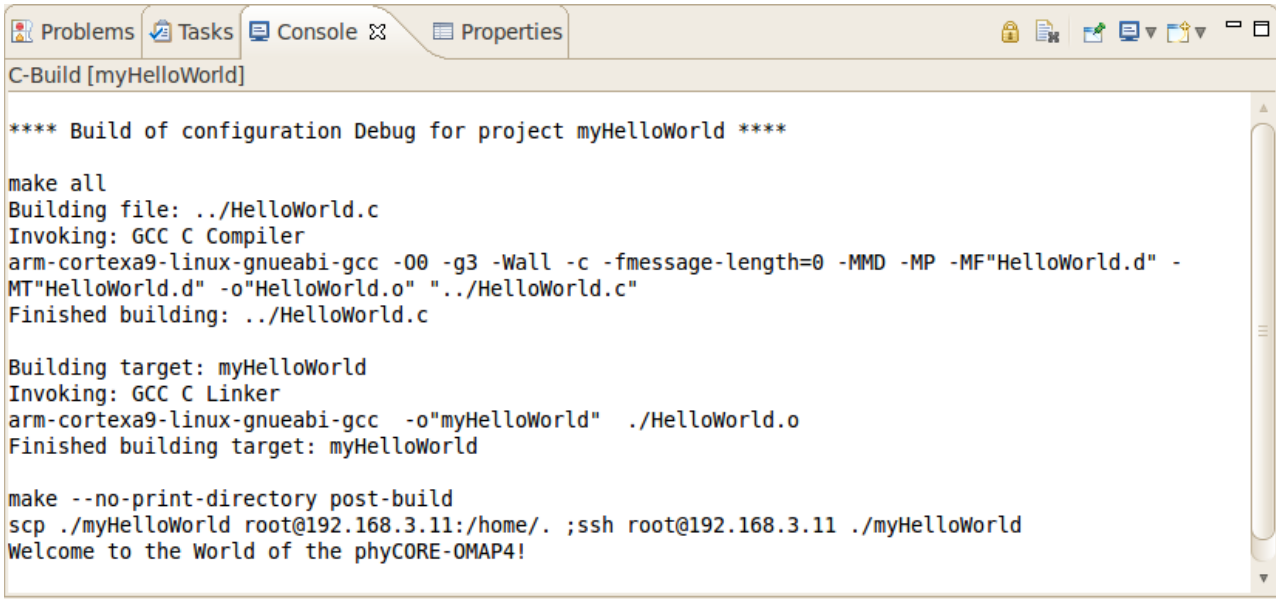


Be sure to enter the semicolon before the ssh command.
Be sure that the file *myHelloWorld* on the target will have execution rights, because otherwise *ssh* will fail.

- 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:



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.

4.1.3 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 phyCORE-OMAP4, 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 tow lines in the *main()* function to declare the buffer and call the *write_tty()* function.
  char buf [] = { "Welcome to the World of the phyCORE-OMAP4! (serial)\n" };
  write_tty(buf, sizeof (buf) - 1);

In the next screenshot you can see the complete program.

HelloWorld.c

```
#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 phyCORE-OMAP4!\n");
    char buf[] = {"Welcome to the World of the phyCORE-OMAP4! (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.

Problems Tasks Console Properties

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 phyCORE-OMAP4!
```

- 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 phyCORE-OMAP4! (serial)
  Welcome to the World of the phyCORE-OMAP4!
- 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 an 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.

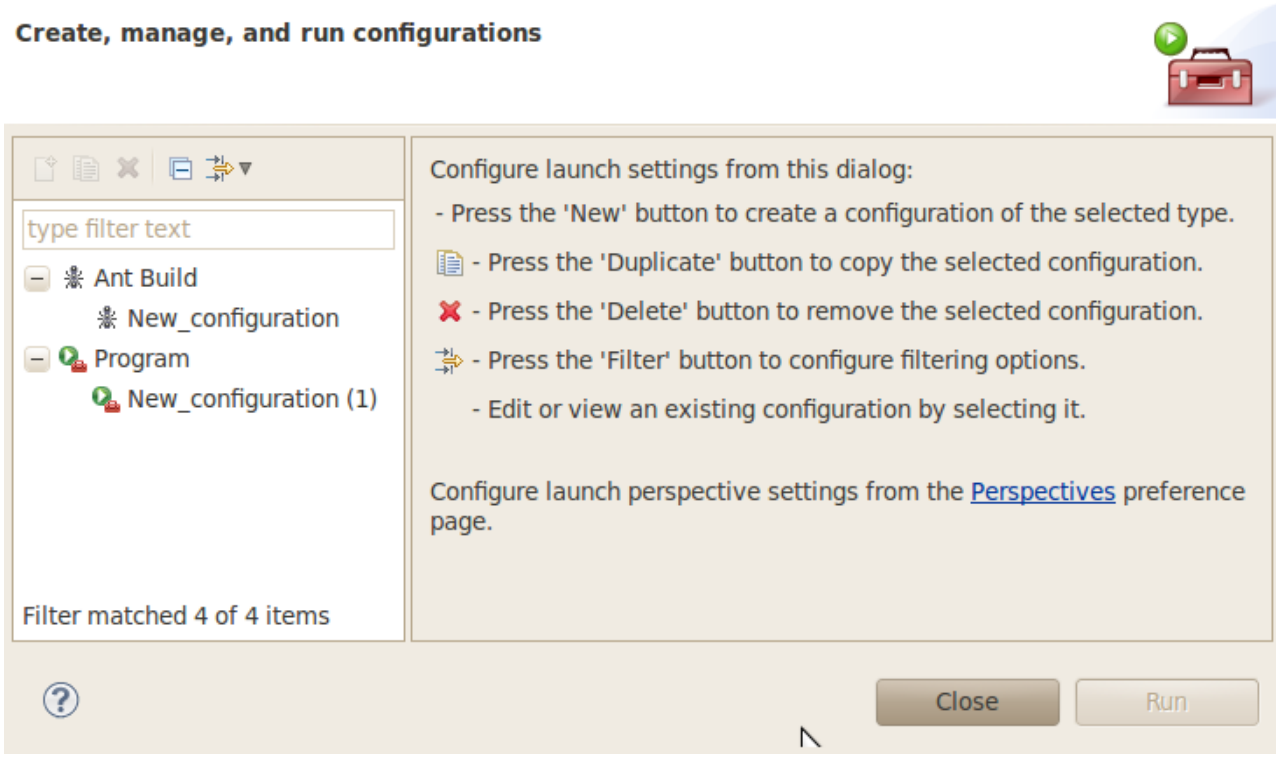
In this section you have changed an existing application. You also learned how to access the serial interface. First you called the function *open()* on the device */dev/console*. The return value of this function was a file descriptor. With the file descriptor you called the function *write()* to send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

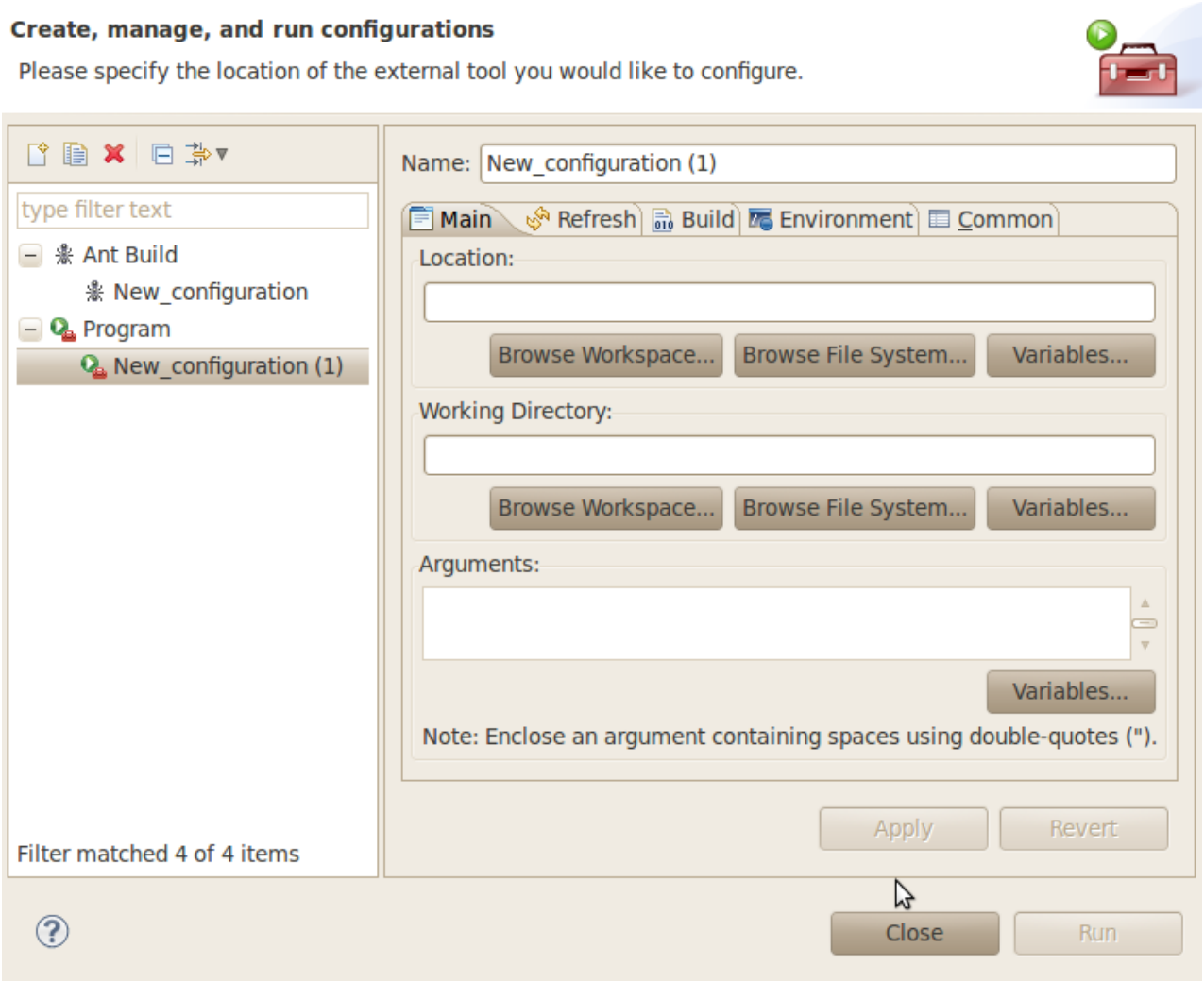
4.1.4 Starting a Program out of Eclipse on the Target

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.

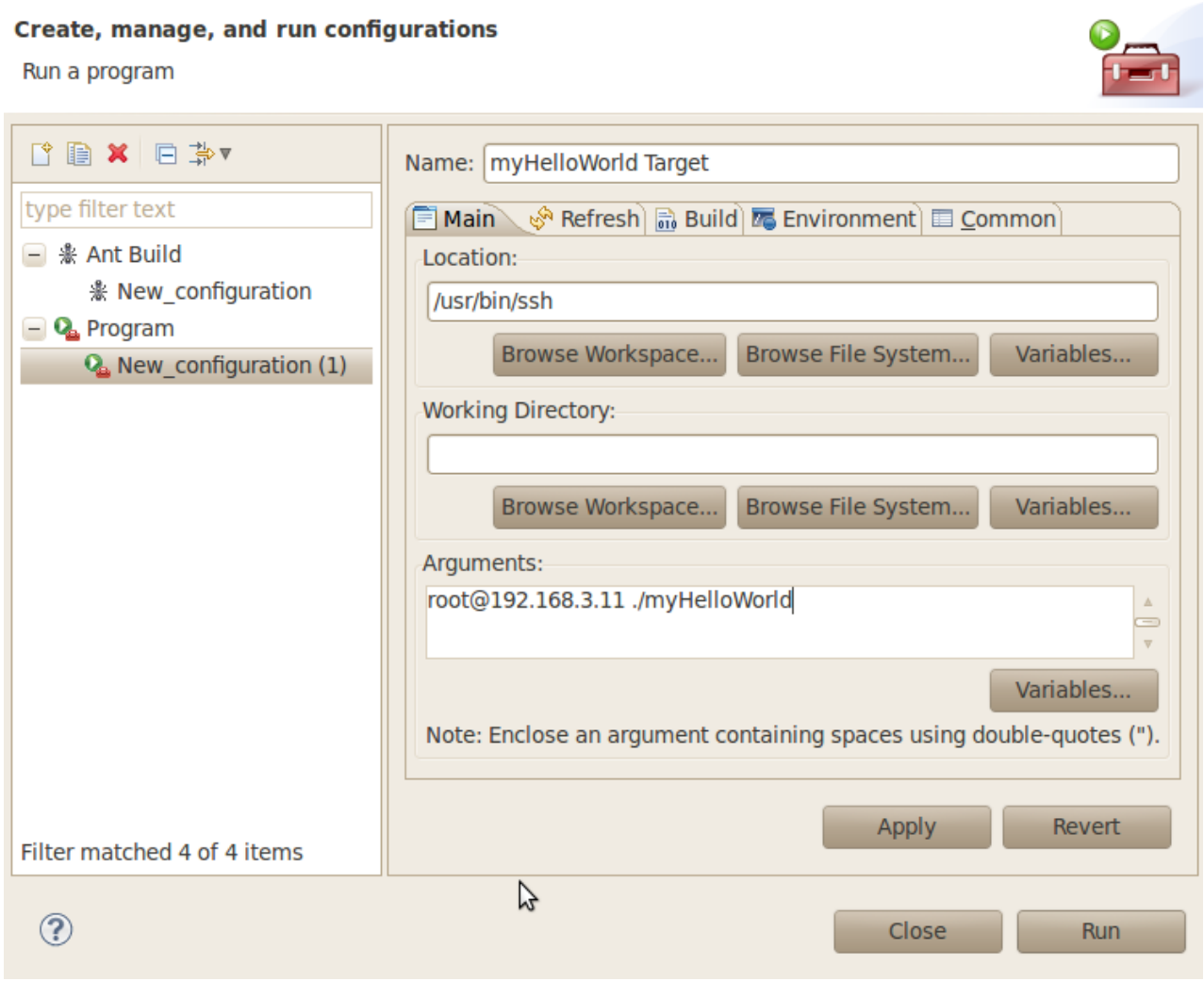
- Select *Run* ▶ *External Tools* ▶ *External Tools Configurations* from the menu bar.



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



- 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*.



- Select *Run*.

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

button from the menu bar.

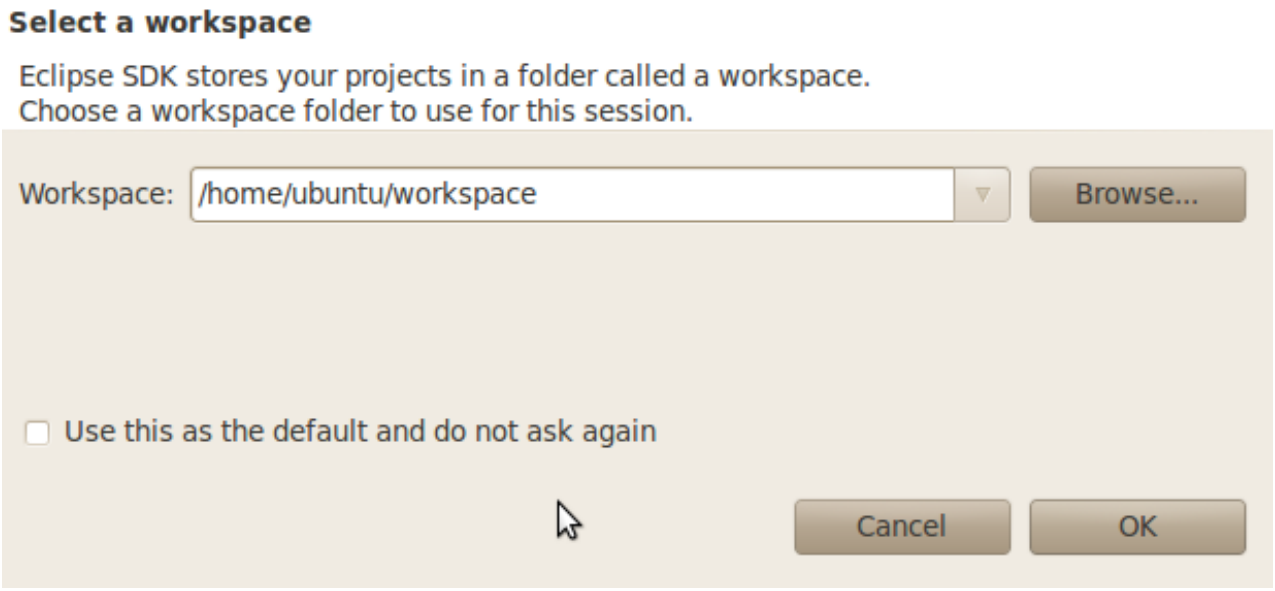
You have successfully create your own Eclipse project and you learned how to execute a program at the target.

4.2 Programming in the Qt C++ perspective

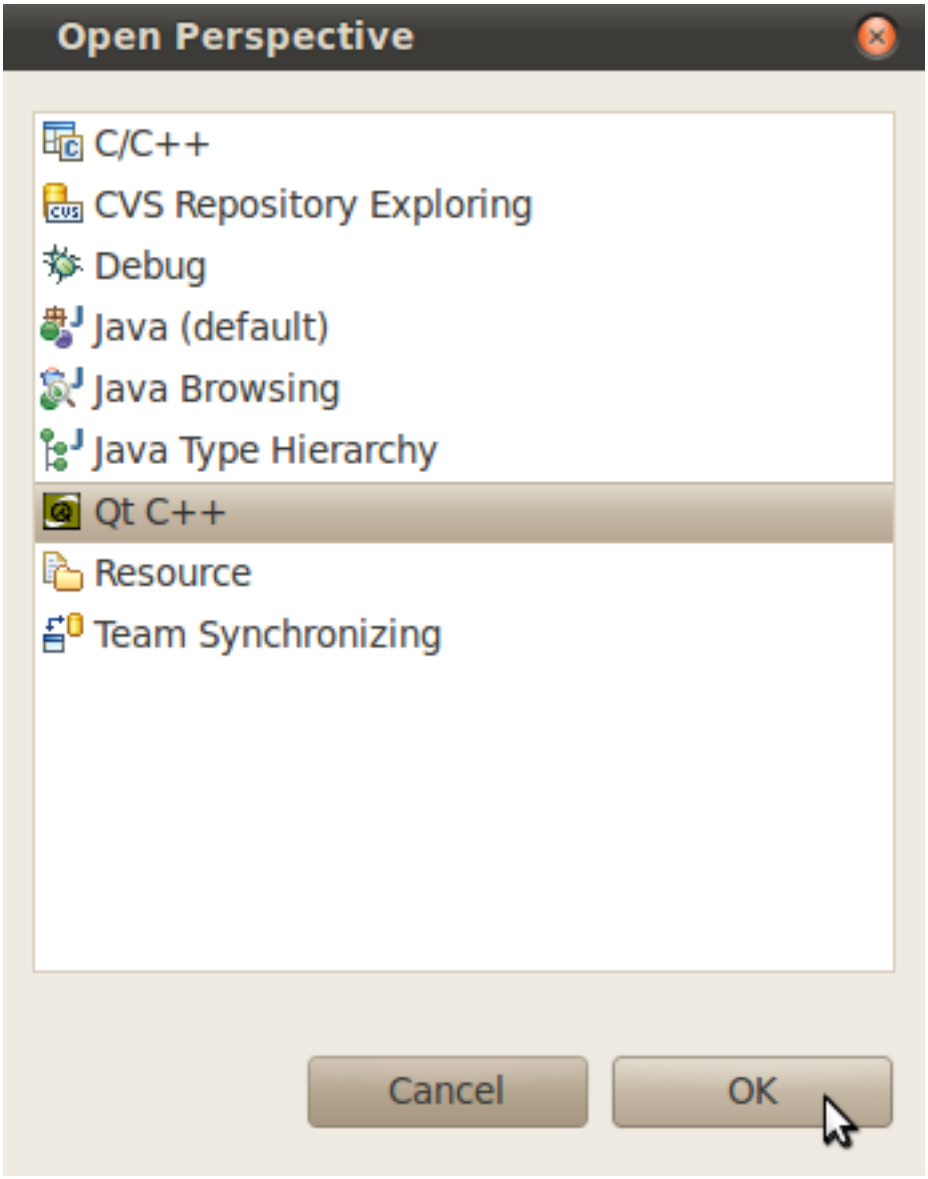
In this section our attention goes to the Qt framework, which gives us tools to develop graphical user interfaces. With the help of an example project we will give you a short introduction of how to work with Qt.

4.2.1 Importing the demo application

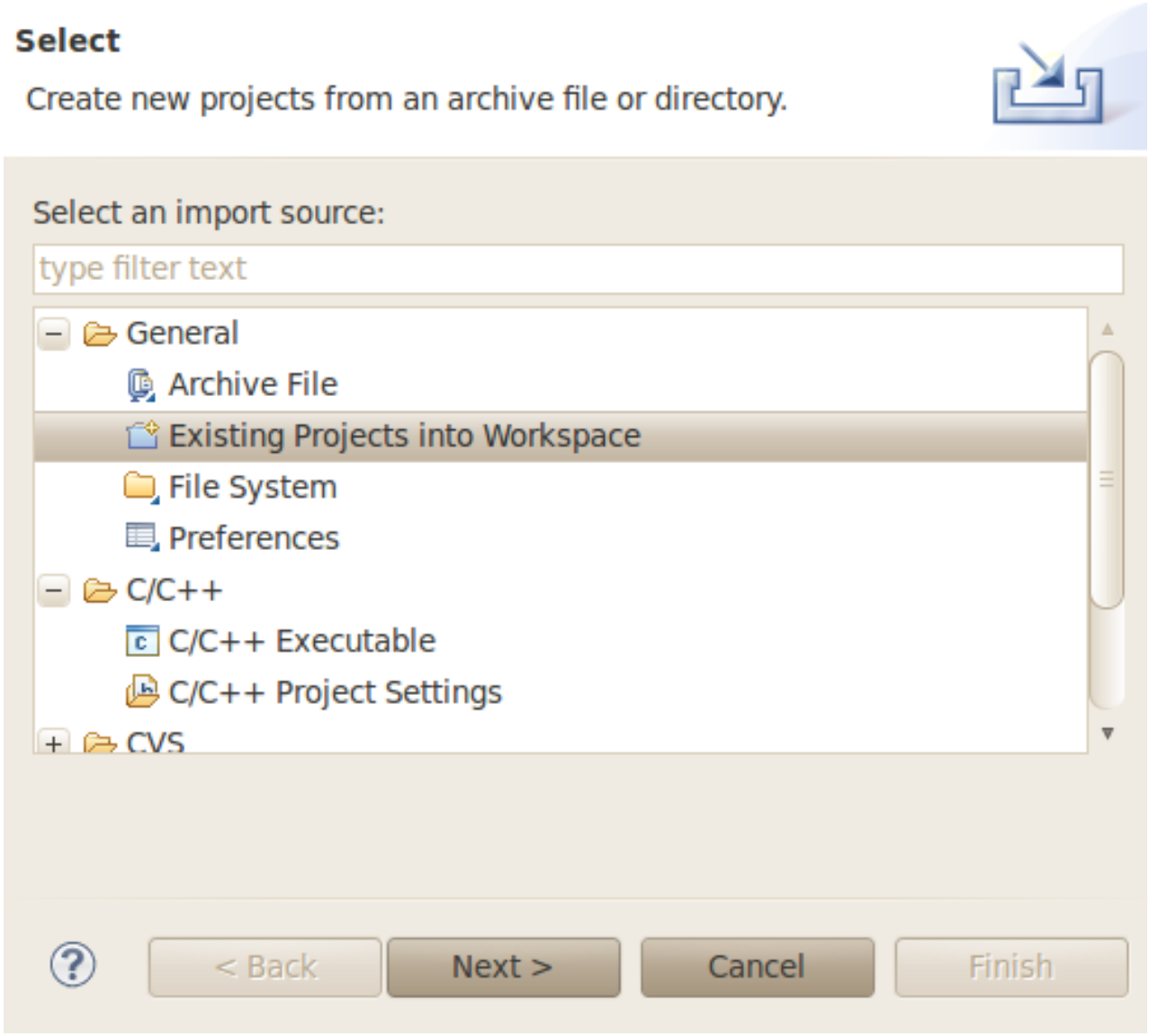
- Click the Eclipse icon to start the application, if it isn't already open. You can find this icon on your desktop.



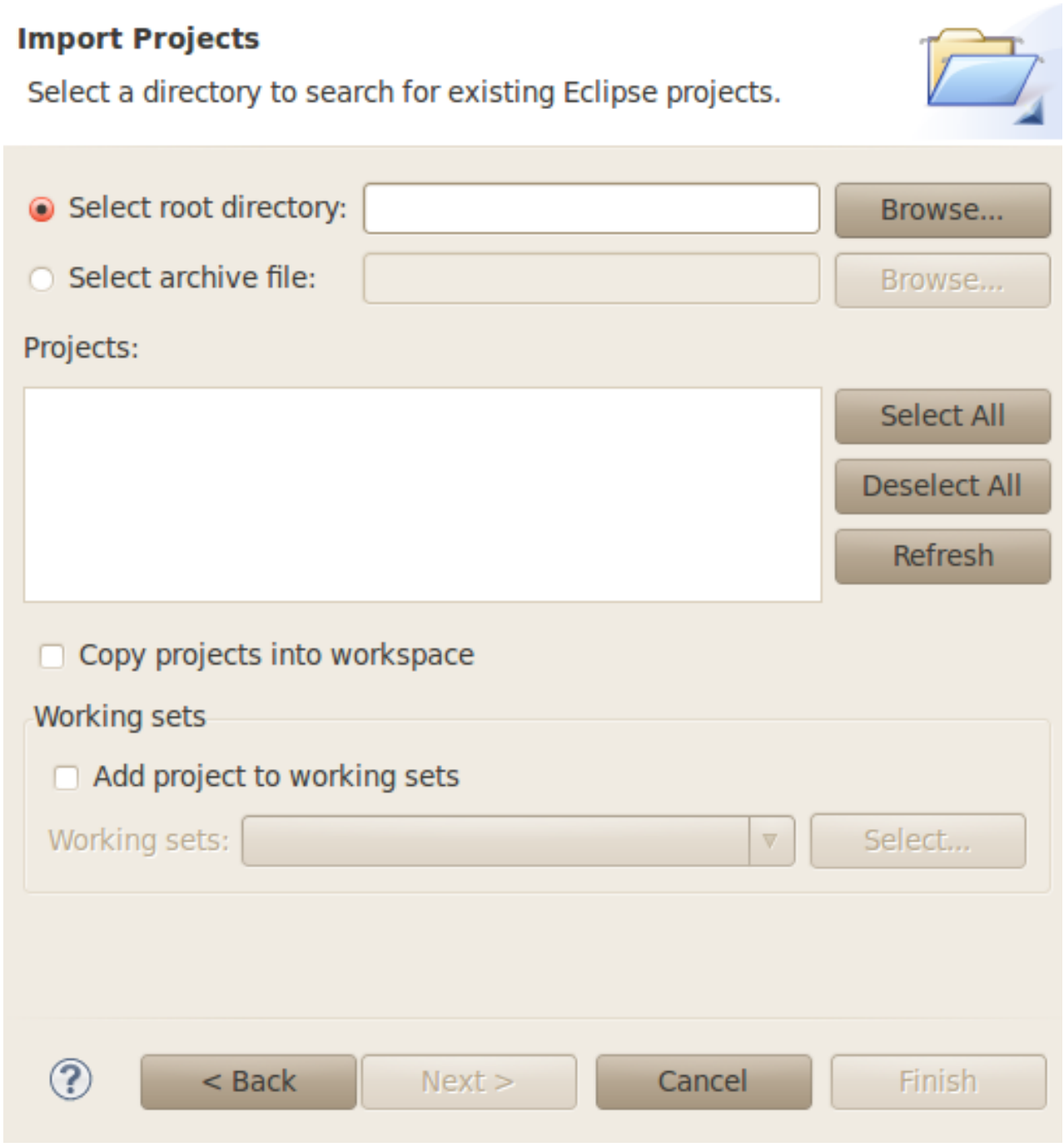
- Now we switch to the Qt C++ perspective. Click *Window* ▶ *Open Perspective* ▶ *Other*.



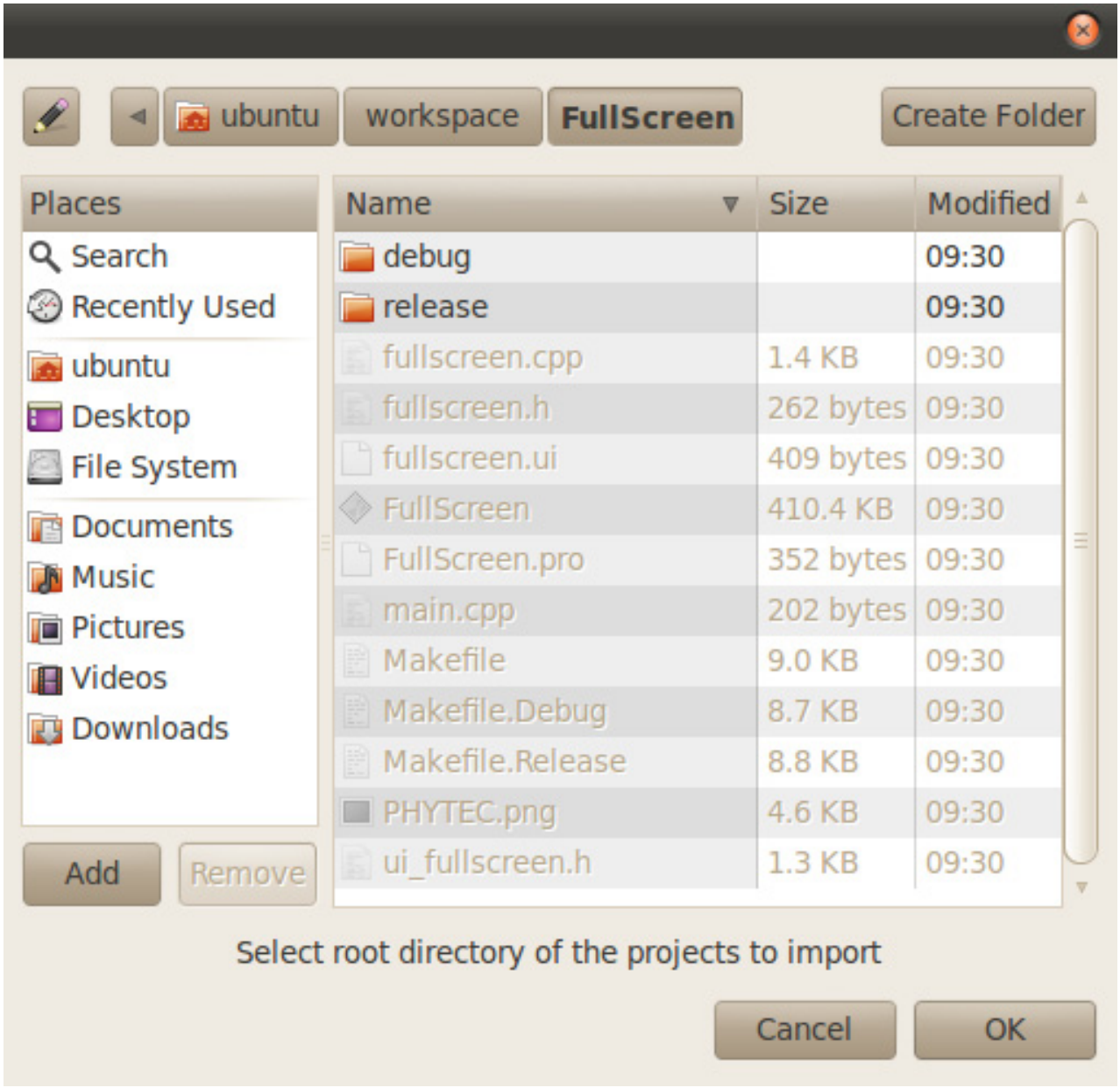
- A dialog opens. Choose *Qt C++* and click **OK**.
- Now we can import the example project. Select *File* ► *Import* from the menu bar.



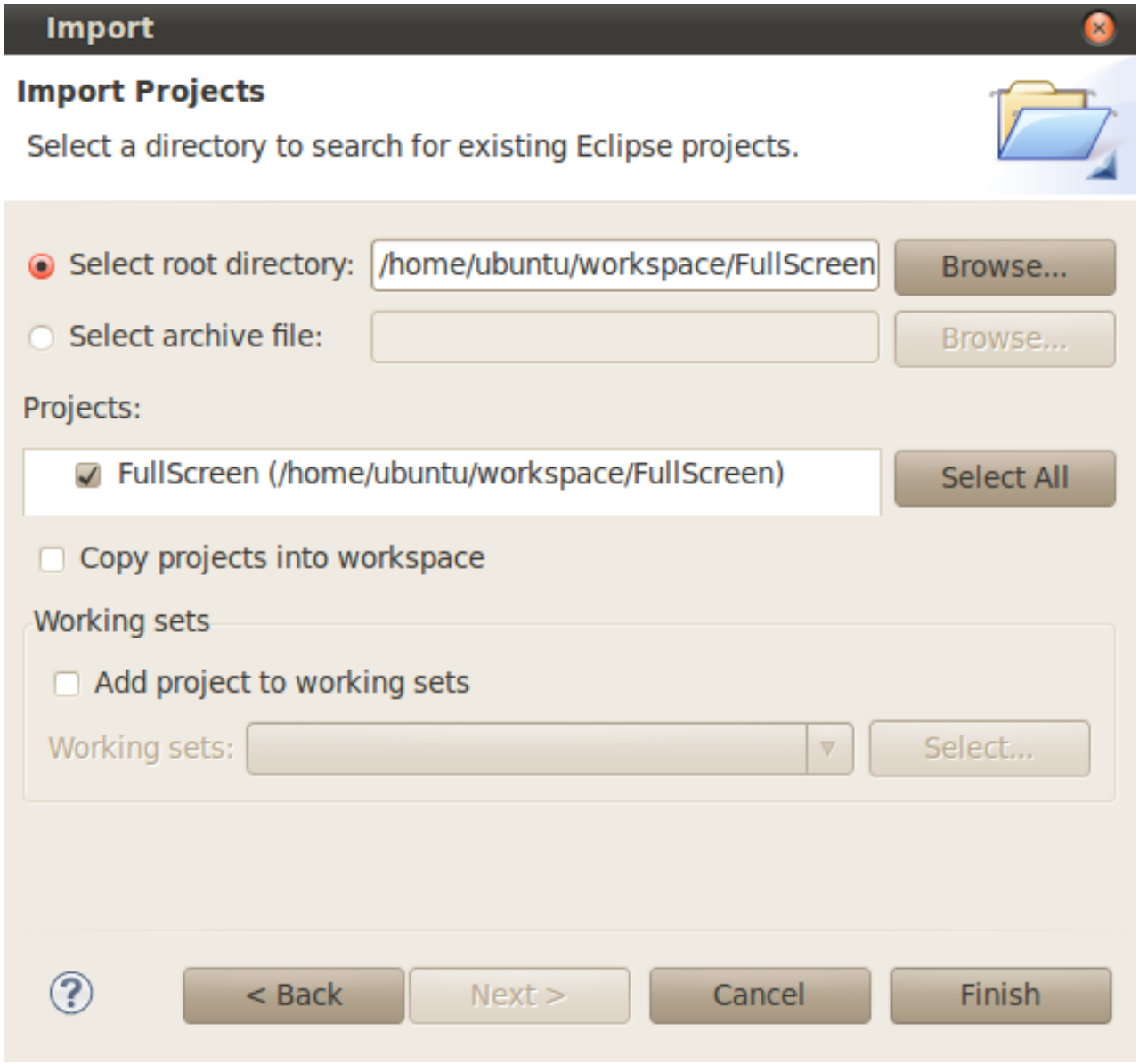
- Select *Existing Projects into Workspace*.
- Click *Next*.



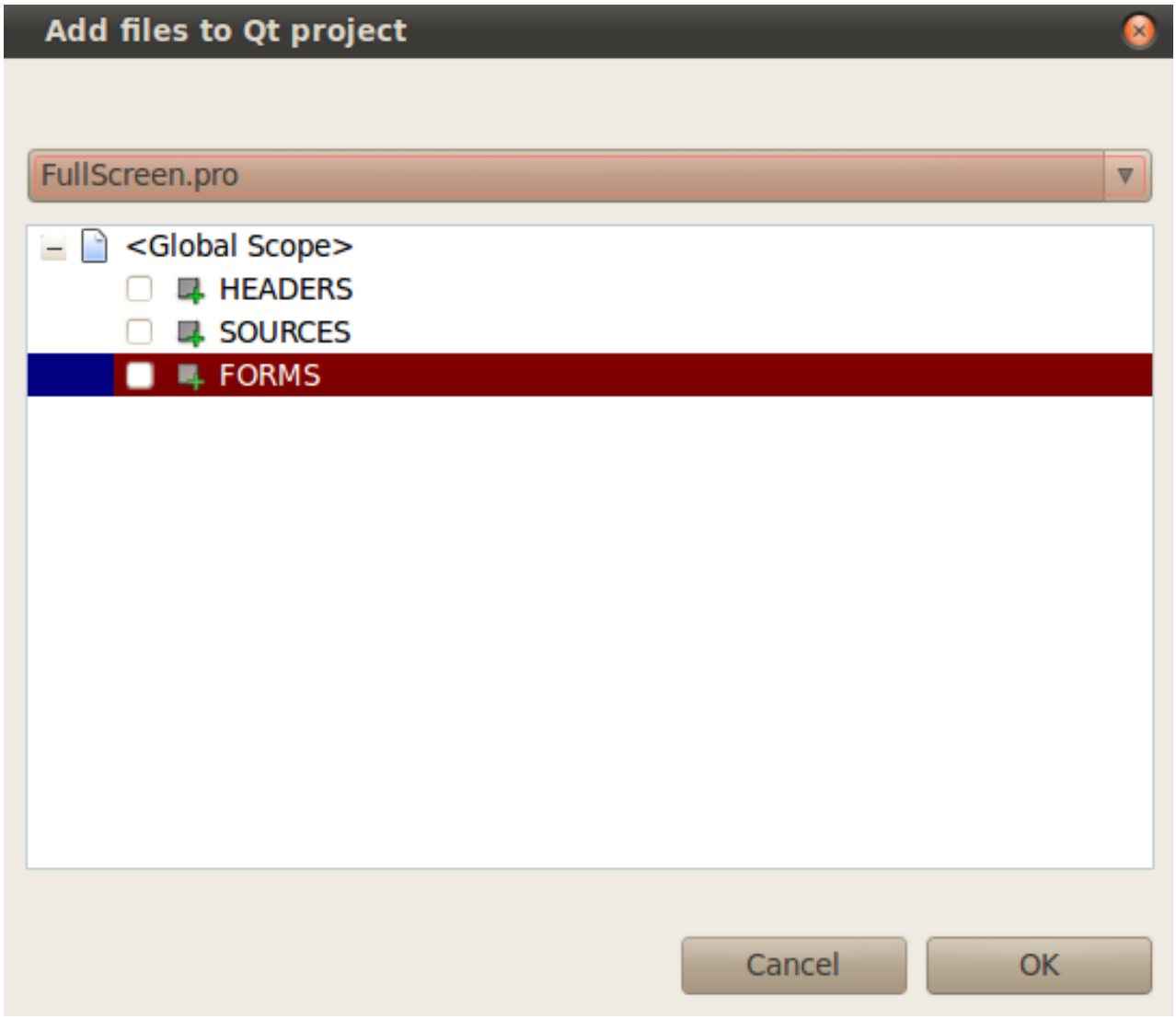
- Select *Browse*.
- Double-click the *FullScreen* directory under */home/ubuntu/workspace/* .



- Click *OK*.



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



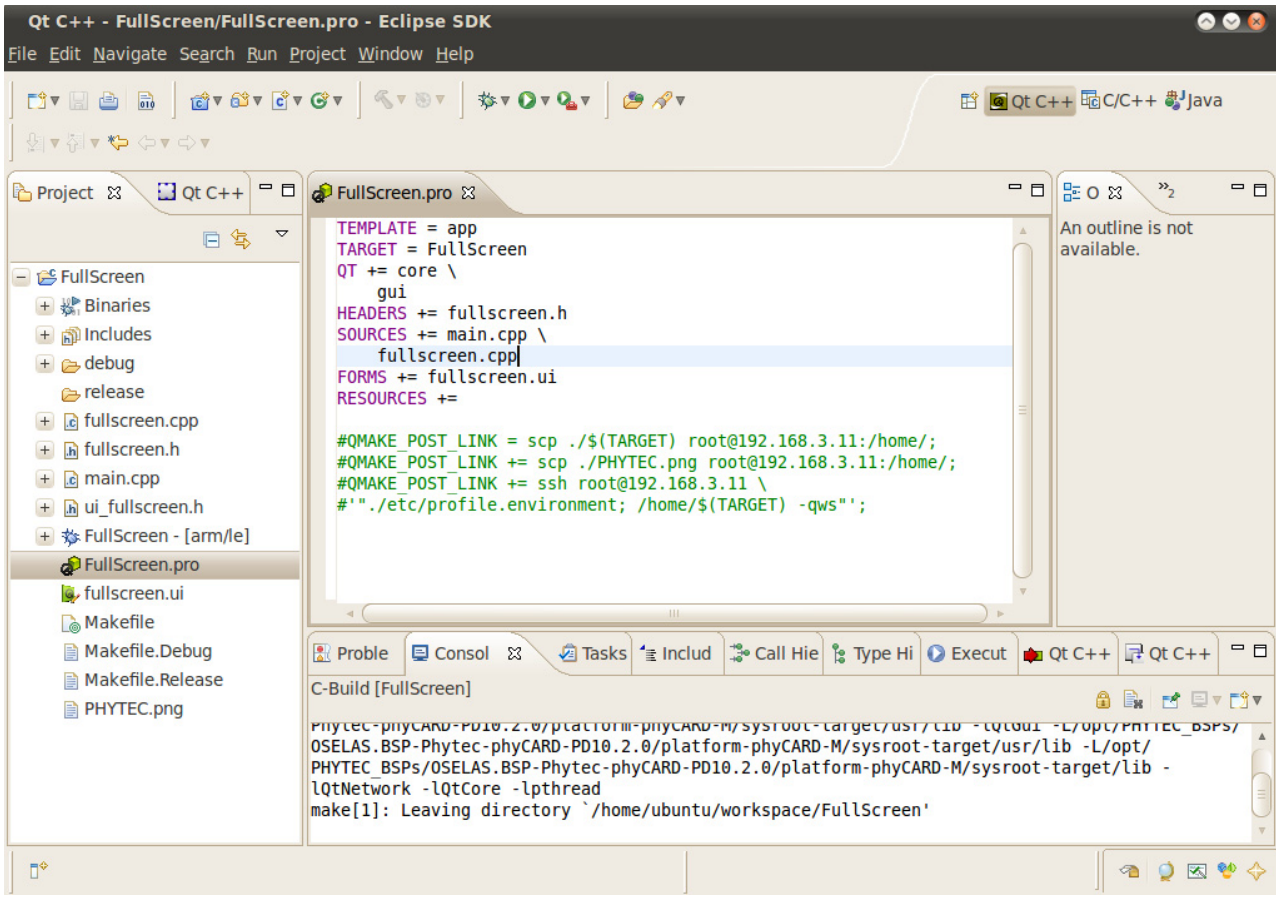
The *FullScreen* program will be compiled and you can find the outputs on the console. After the compilation is finished successfully the *FullScreen* executable for the target is built and can be found under */home/ubuntu/workspace/FullScreen*.

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

4.2.2 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.

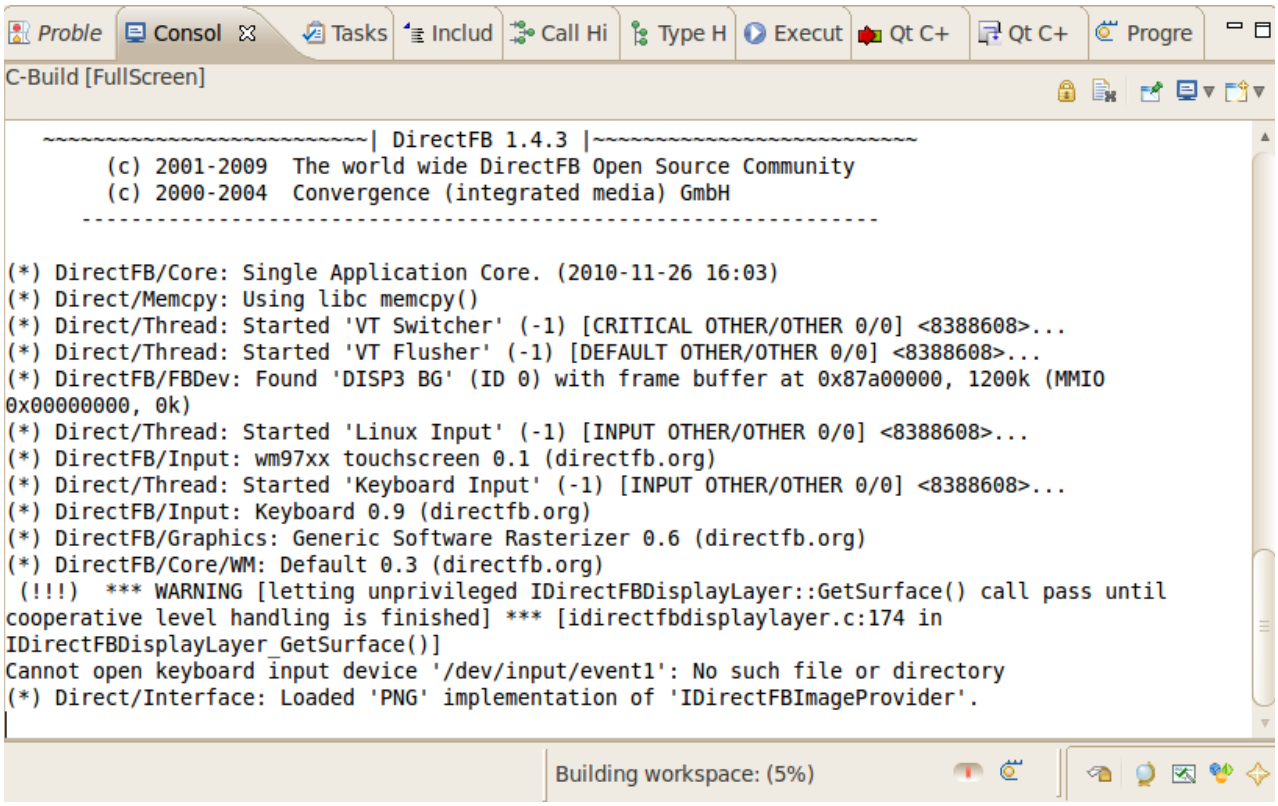


In this file you will find four uncommented rows.

- Remove the commentar 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:



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.

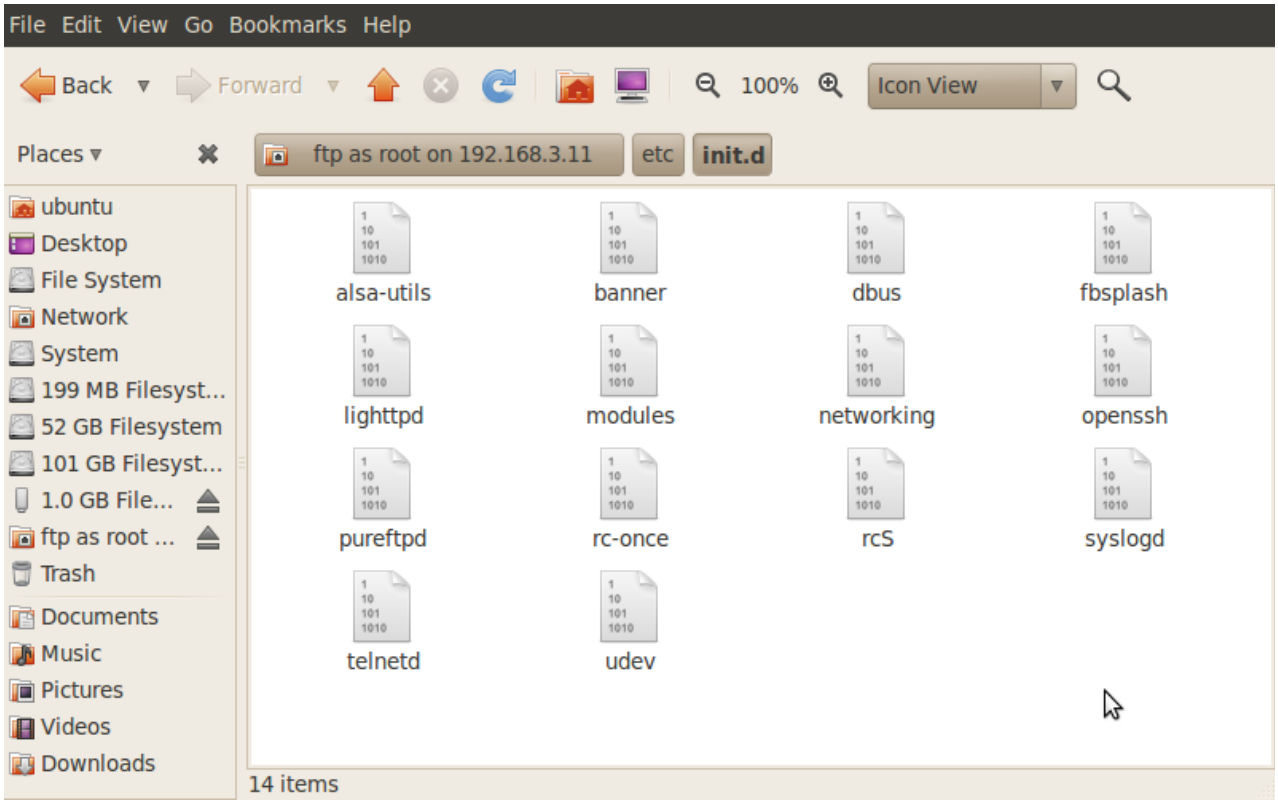
4.3 Automatically Starting the Program when booting the Target

In this passage you will integrate the *myHelloWorld* program into the startup process of the target. When you have finished this part, the *myHelloWorld* application will be started automatically each time you are starting the target.

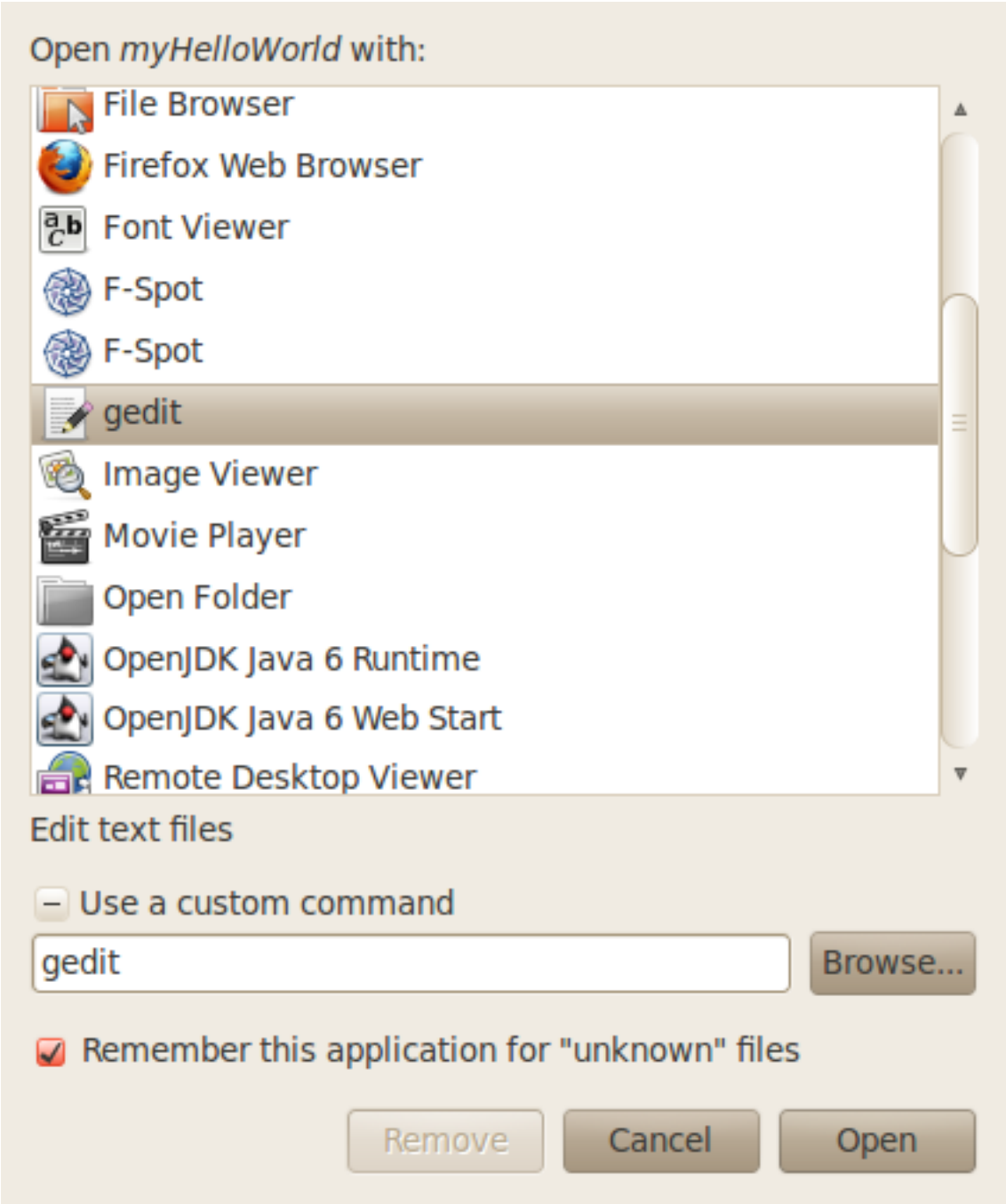
The scripts for controlling the system startup will be found in */etc/init.d*. These are executed directly or indirectly by */sbin/init*, the father of all processes. The configuration of */sbin/init* is placed in */etc/inittab*.
After system startup, */sbin/init* will switch to the default run level, as configured in */etc/inittab*. It calls the run level master script */etc/init.d/rcS* to start or stop services provided by the other scripts in */etc/init.d*. This is done by the help of symbolic links in the directory */etc/rc.d*. These links point to the actual startup scripts in */etc/init.d*.

First you will have to create a startup script in */etc/init.d*.

- Click the *FTP for Target* icon on your desktop.
- Browse to the target's */etc/init.d*. If an authorization dialog should appear, just click *OK* (no password is required).
  In the directory */etc/init.d* you can see the existing scripts.



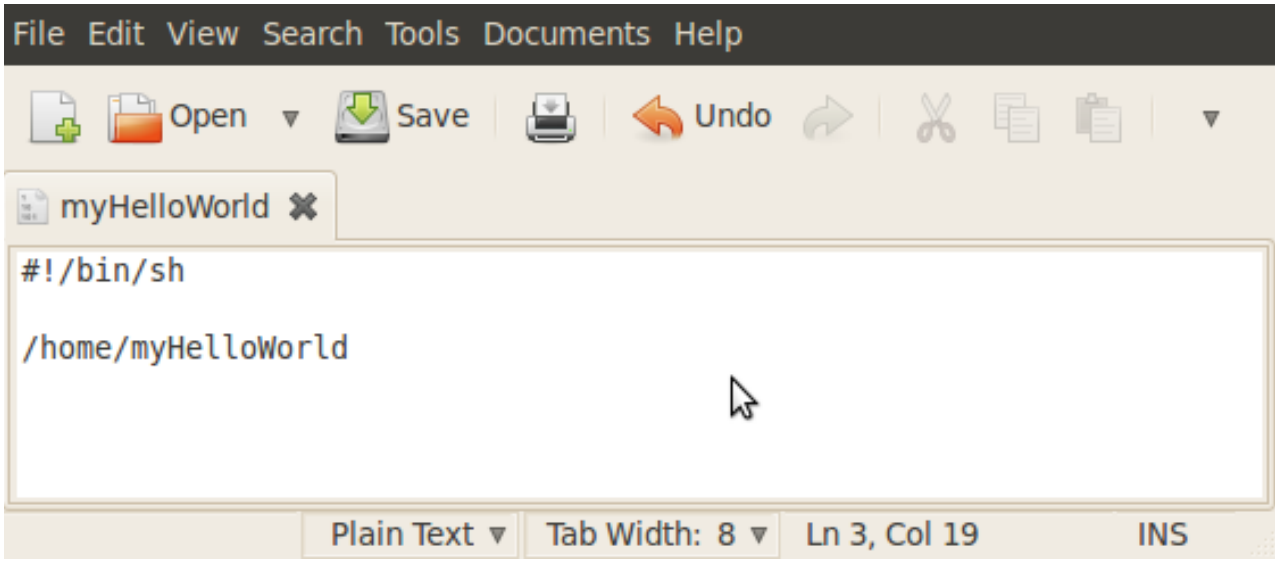
- Right-click in the opened window and select *Create Document* ▶ *Empty File*.
- Enter *myHelloWorld* as file name.
- Right-click on *myHelloWorld* and select *Open with Other Application...*.
- Select **gedit** in the list of applications or enter **gedit** in the custom command and click *Open*.



- Select **gedit** in the list of applications or enter **gedit** in the custom command and click *Open*.

The text editor *gedit* starts with an empty document.

- Enter the following two lines:
  #!/bin/sh
  /home/myHelloWorld

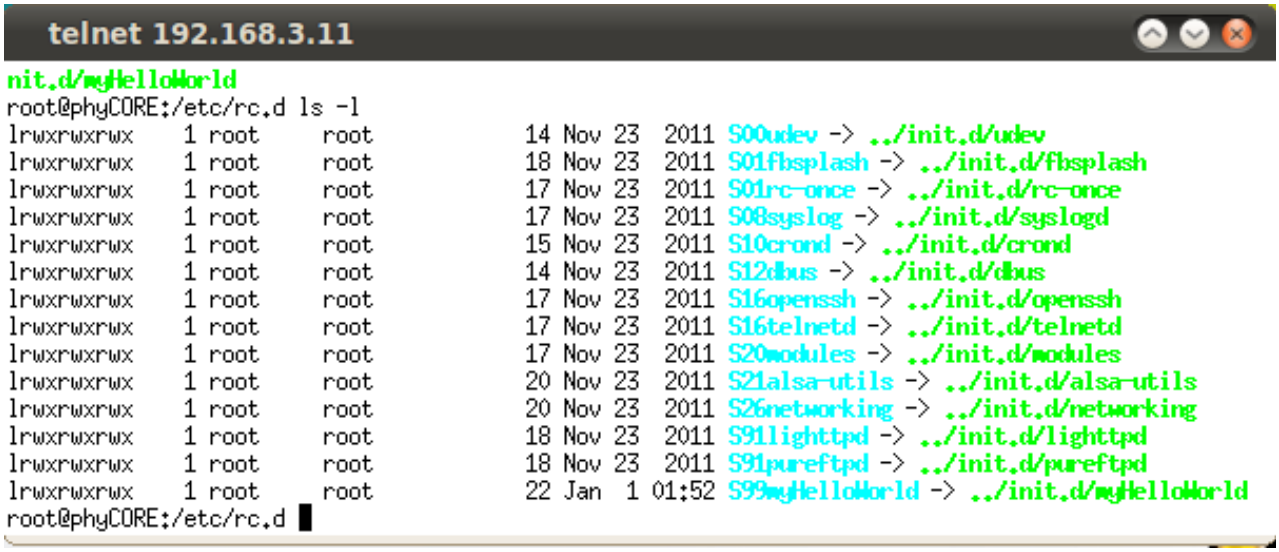


- Select ▶ Save.
- Close the *gedit* write.
- Close the *FTP* window.

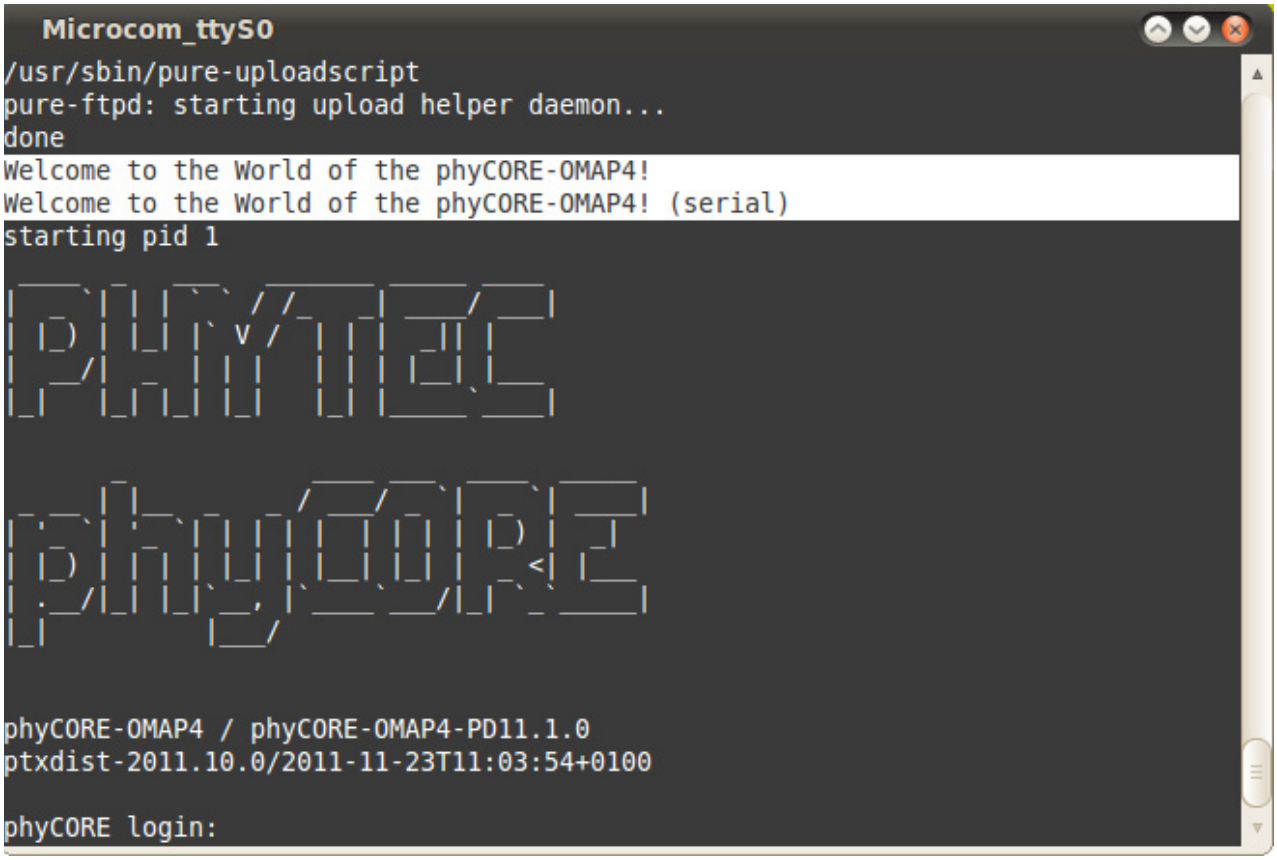
- Click the *Telnet for Target* icon on your desktop.
- Enter **root** and press *Enter* to login.
- The startup script we have created with gedit must be made executable in order to run it later:
  chmod a+x /etc/init.d/myHelloWorld
- Change to the directory */etc/rc.d*. Type the following command:
  cd /etc/rc.d
- Enter **ls -l** to list the directory content.
  You can see the different links to the scripts in the directory */etc/init.d*. The scripts are started in alphabetic order: The script *udev* is the first script started, because the

link starts with *S00*....
To start your *myHelloWorld* application automatically when the system boots, you have to create a new link to the start script */etc/init.d/myHelloWorld*.

- In /etc/rc.d, create a symbolic link which points to /etc/init.d/myHelloWorld. Enter the following command:
  ln -s ../init.d./myHelloWorld S99myHelloWorld
- Type **ls -l** again to check the newly created link.



- Close the window.
- Open *Microcom*.
- Push the RESET button on the target to restart your system.



The program *myHelloWorld* now starts automatically on startup. Because its link in */etc/rc.d* starts with *S99...*, you should see *myHelloWorld*'s output after the output of the two other scripts that start with *S91...*.

- Close *Microcom*.

Now you can add your own programs to the root file system and start these programs automatically when your phyCORE-OMAP4 boots.

You have successfully passed the "Working with Eclipse" part of this Quickstart.

5 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.

5.1 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.

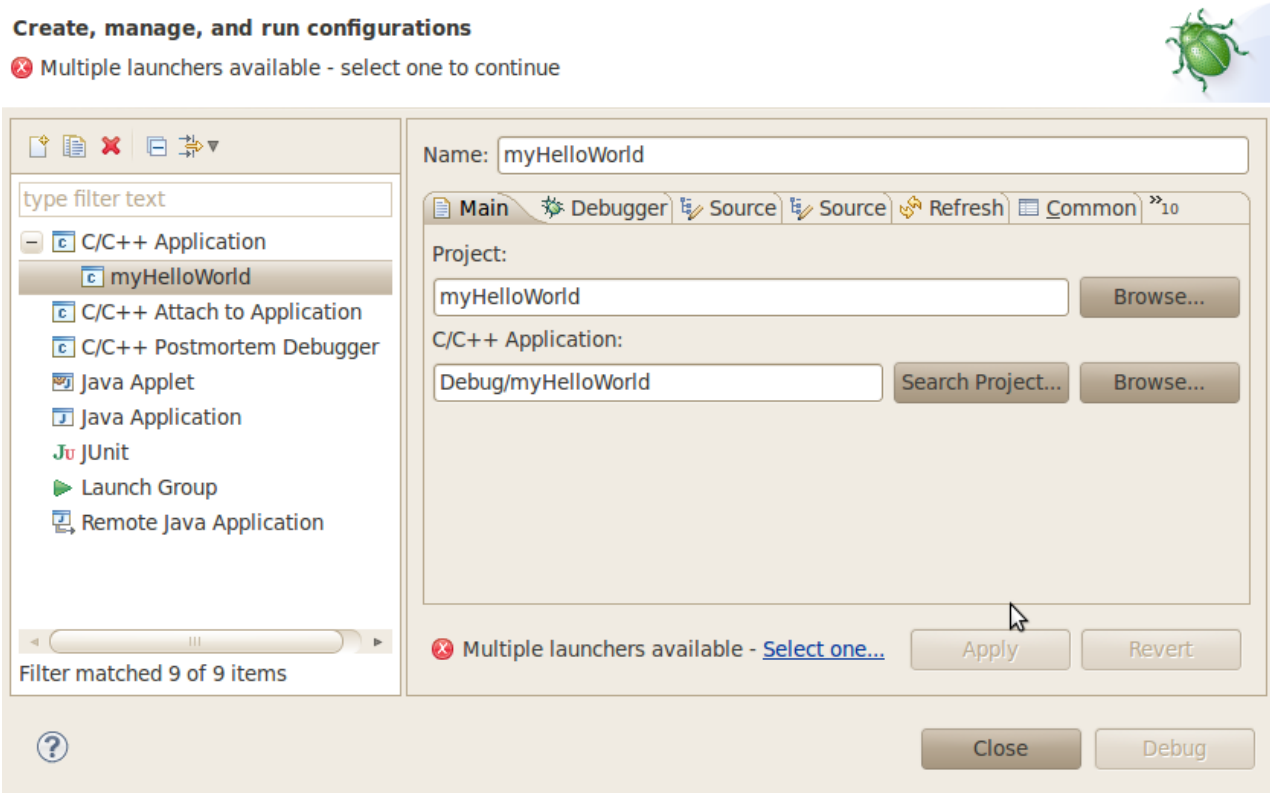
5.2 Configuring and starting the debugger in Eclipse

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.

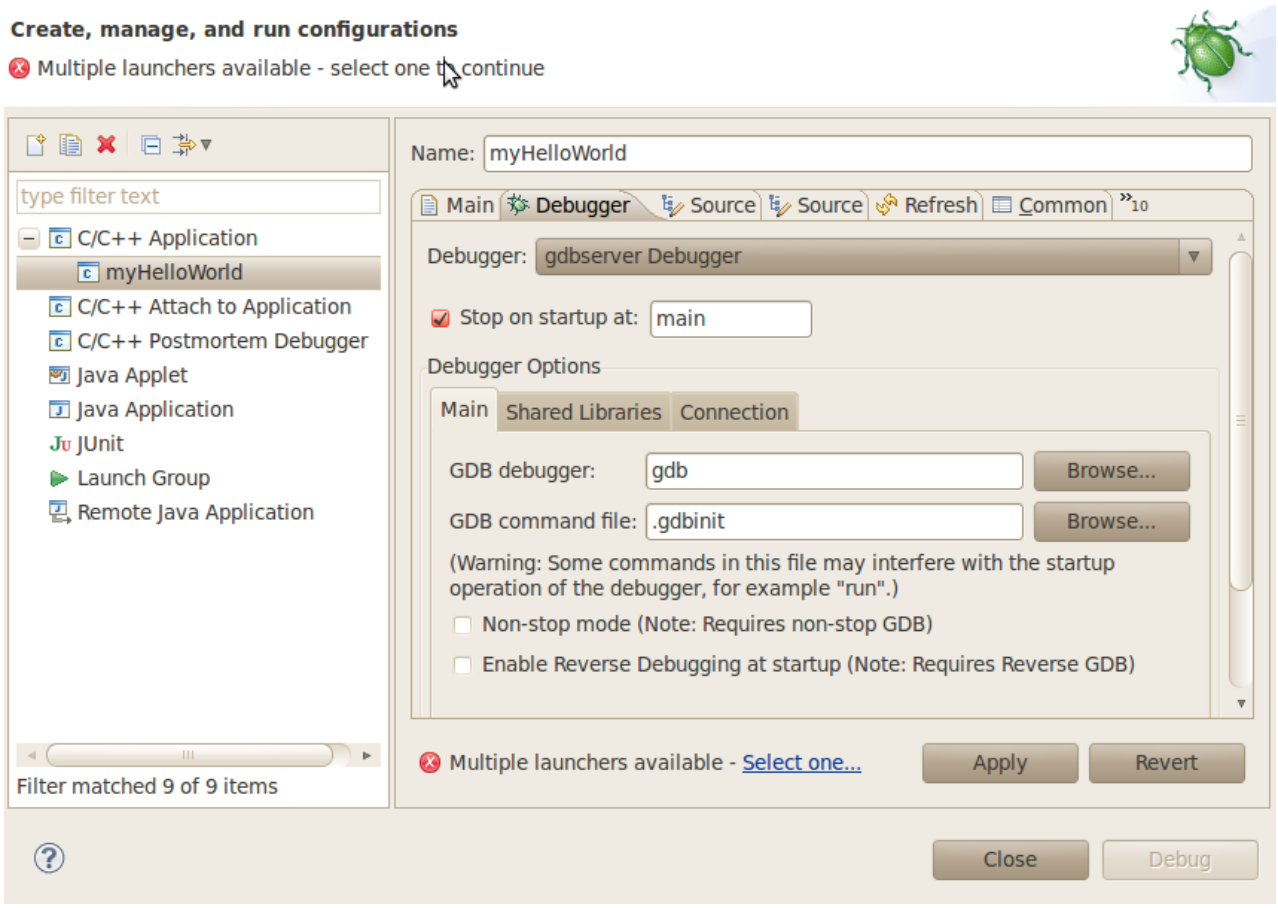
- Start Eclipse if the application is not started yet.
- Right-click on the *myHelloWorld* project in the *Navigator* window.
- Select *Debug As* ► *Debug Configurations*.

A dialog to create, manage and run applications will appear.

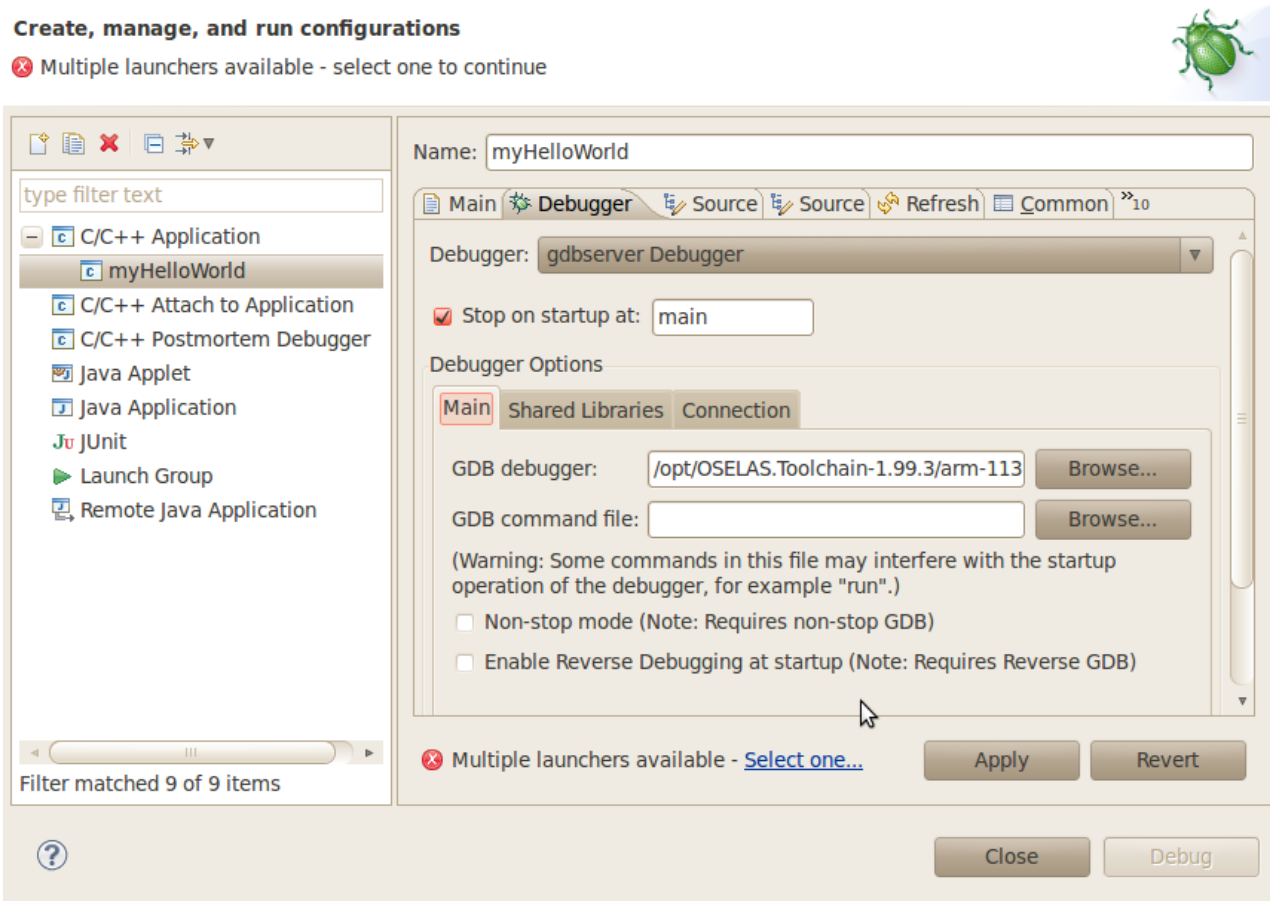
- Select *myHelloWorld* under *C/C++ Application* .



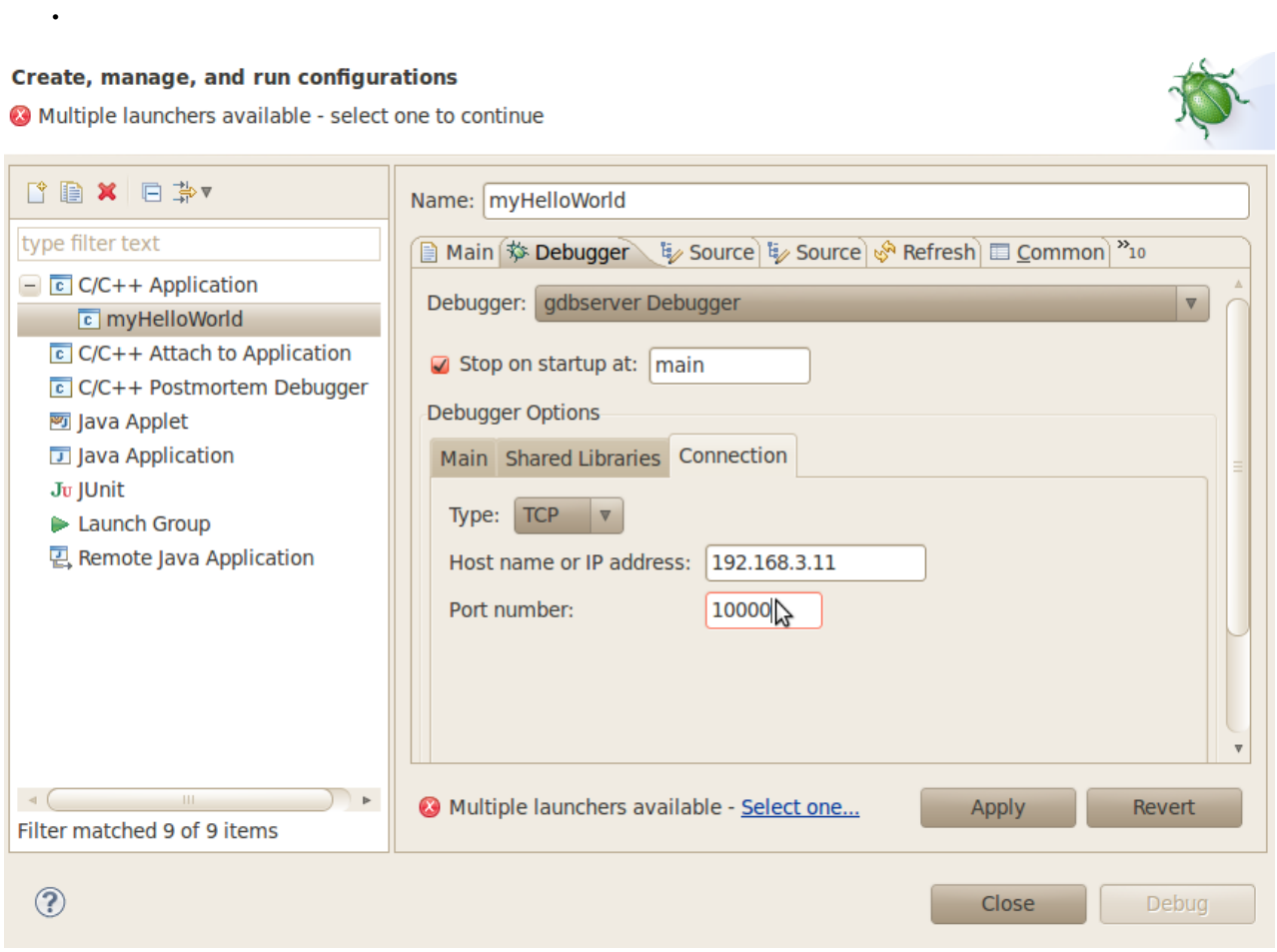
- Select the *Debugger* tab.



- 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/OSELAS.Toolchain*/arm-cortexa9-linux-gnueabi/gcc-linaro-4.5-2011.02-0-glibc-2.13-binutils-2.21-kernel-2.6.36-sanitized/bin*.
- Select the file *arm-cortexa9-linux-gnueabi-gdb*.
- Click *OK*.

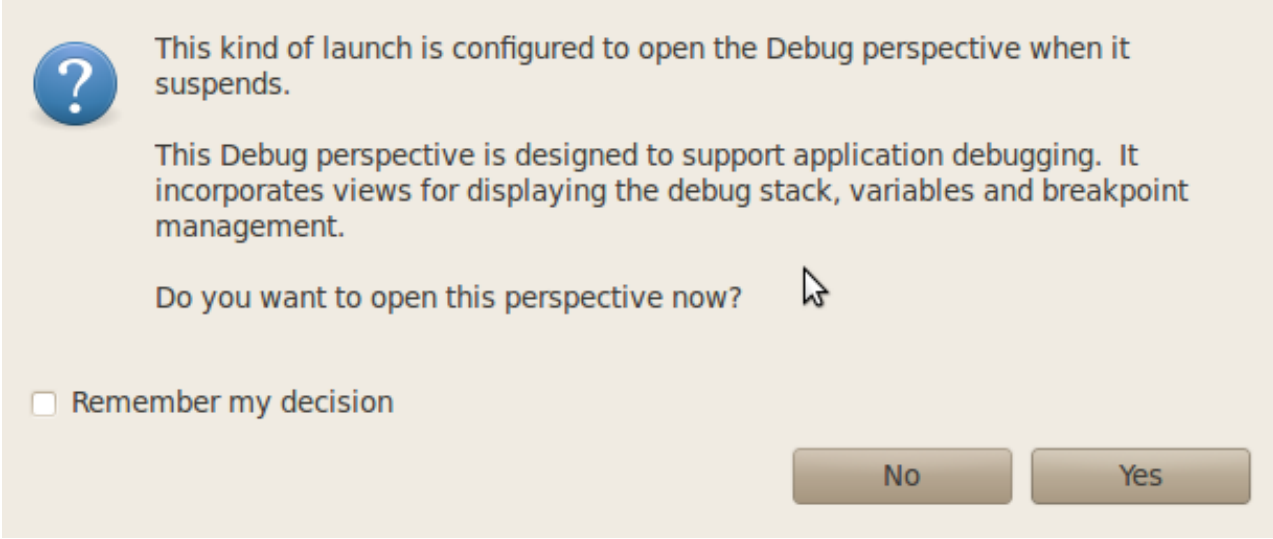


- Keep the *GDB command file* field empty.
- Select the *Connection* tab and select *TCP* in the drop-down box.
- Enter **192.168.3.11** (the target's IP address) in the *Host name* input field. The host's GDB will connect to this IP address to communicate with the target's GDB server.



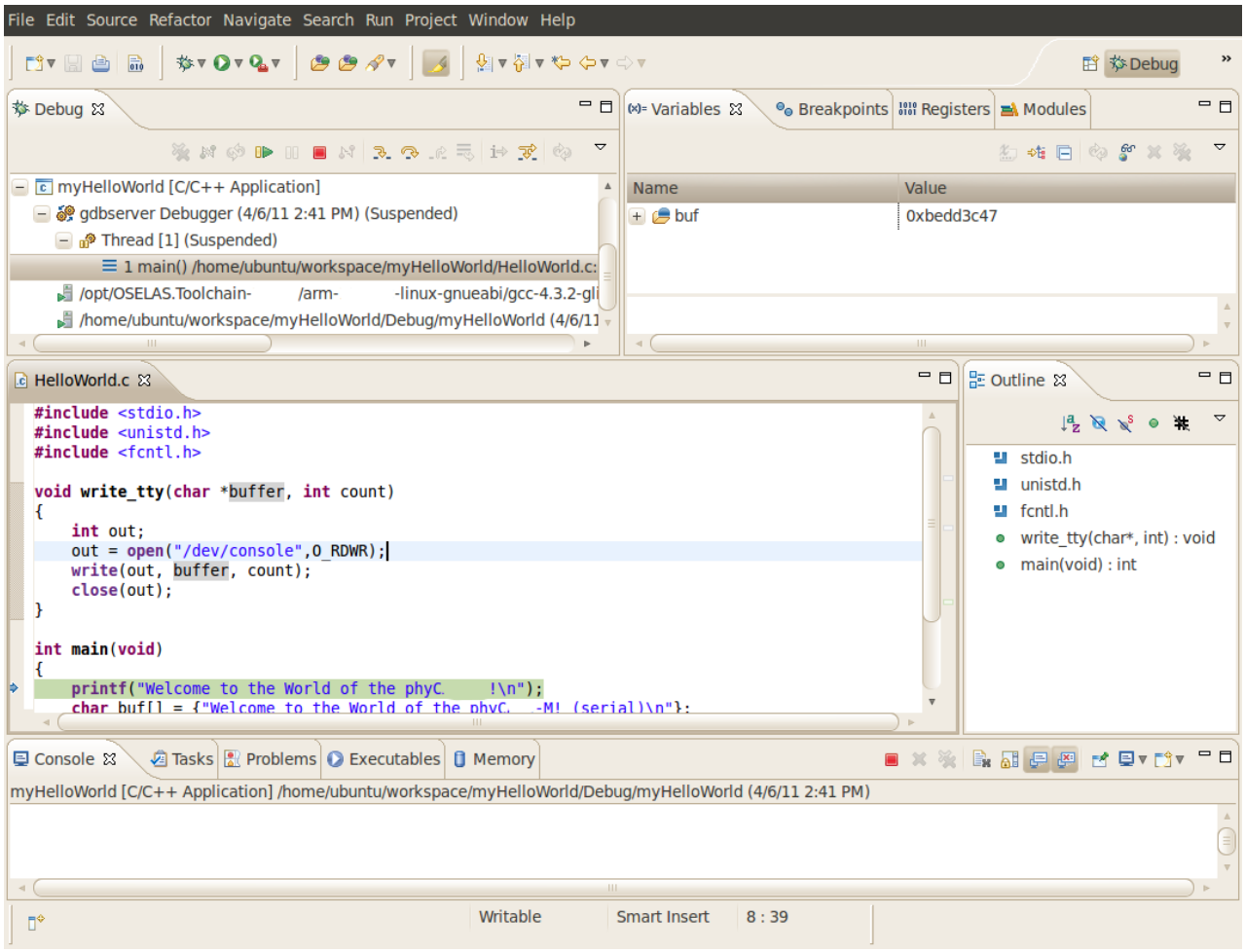
- Click *Apply*.
- Click *Debug*.

A new dialog appears.



- Select Yes to switch to the Debug perspective.

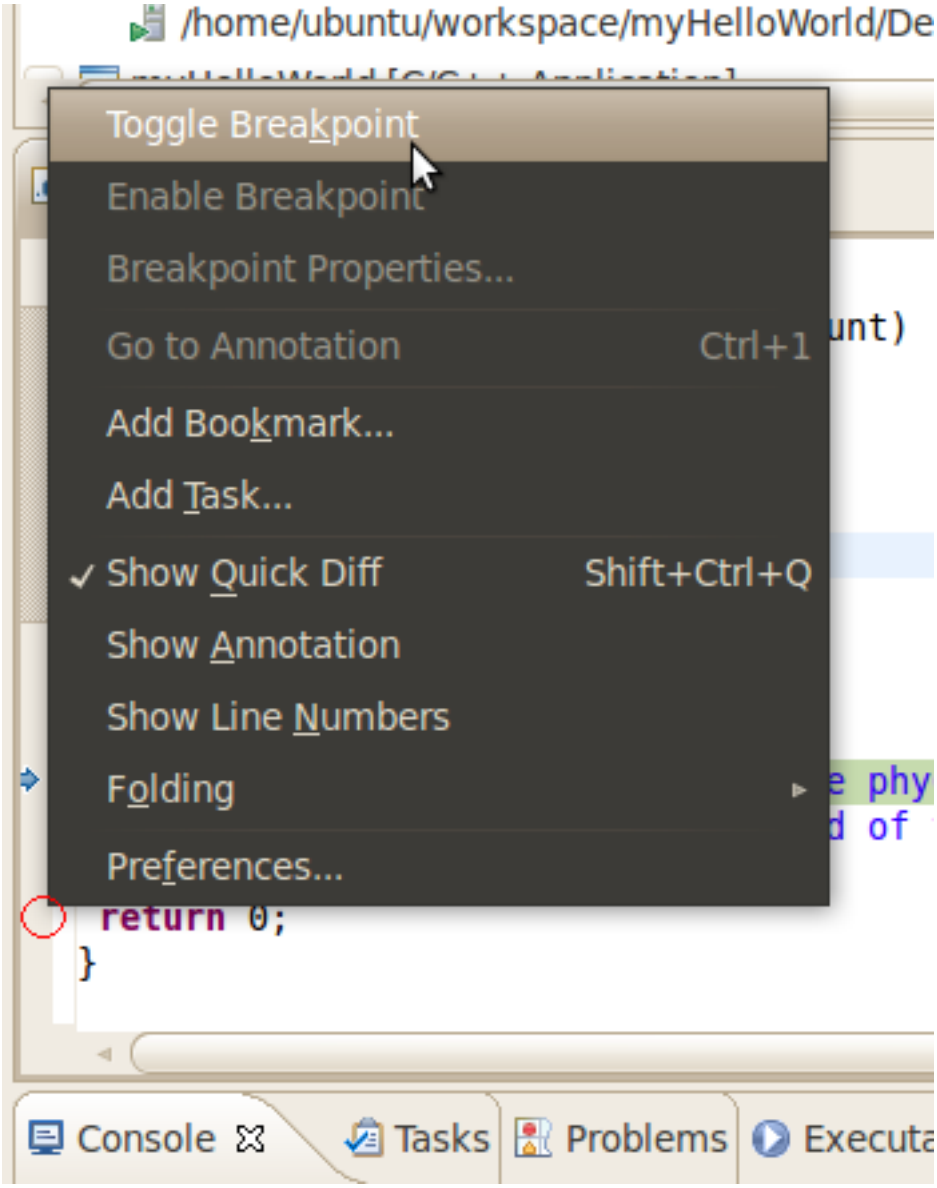
The debug perspective opens and the debugger stops automatically at the first line. The host's GDB is now connected to the GDB server on the target.



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.

5.3 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.

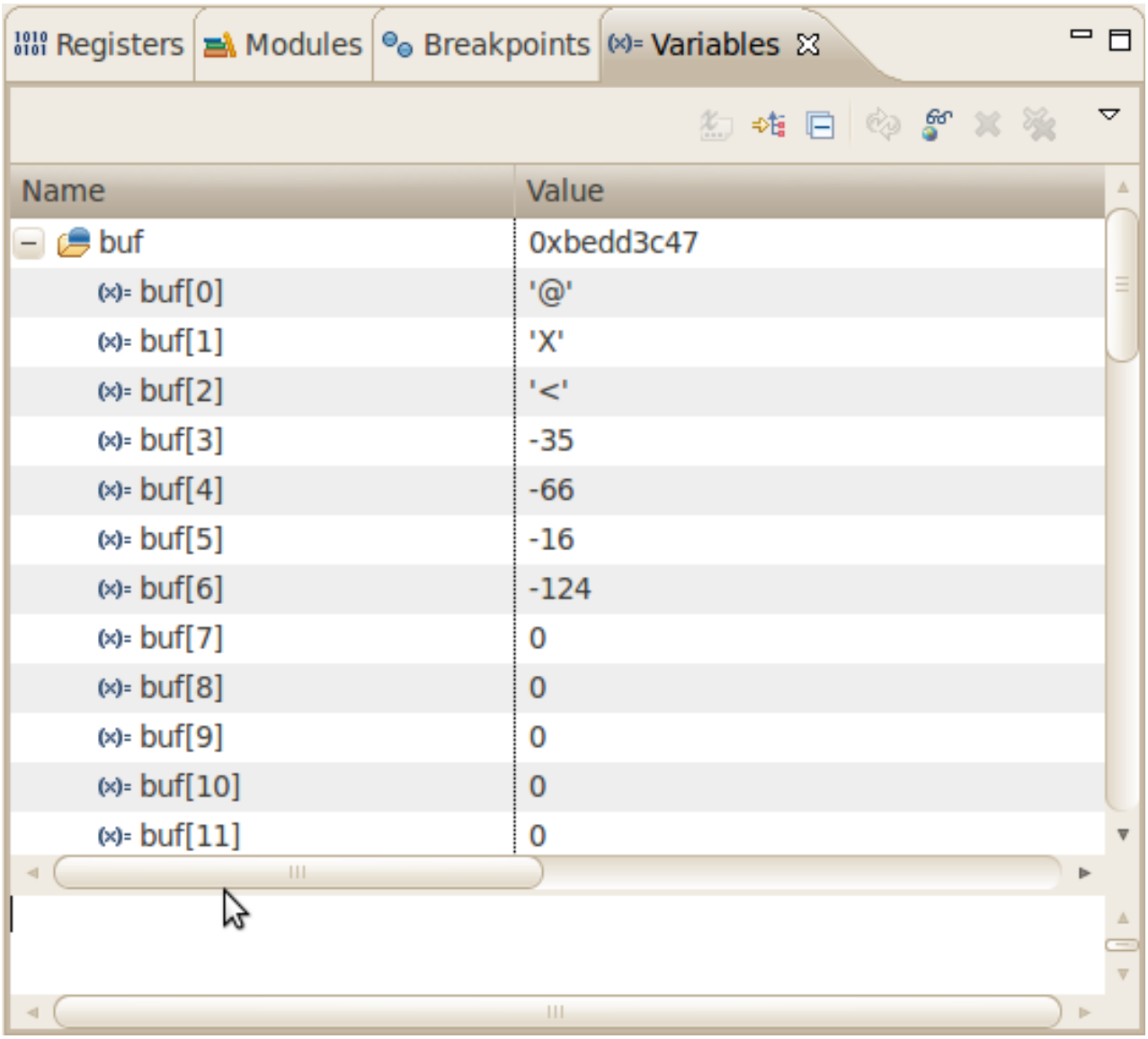


- Select the last line in *main()*.
- Right-click into the small grey border on the left-hand side and select *Toggle Breakpoint* to set a new breakpoint.

5.4 Stepping and Watching Variable Contents

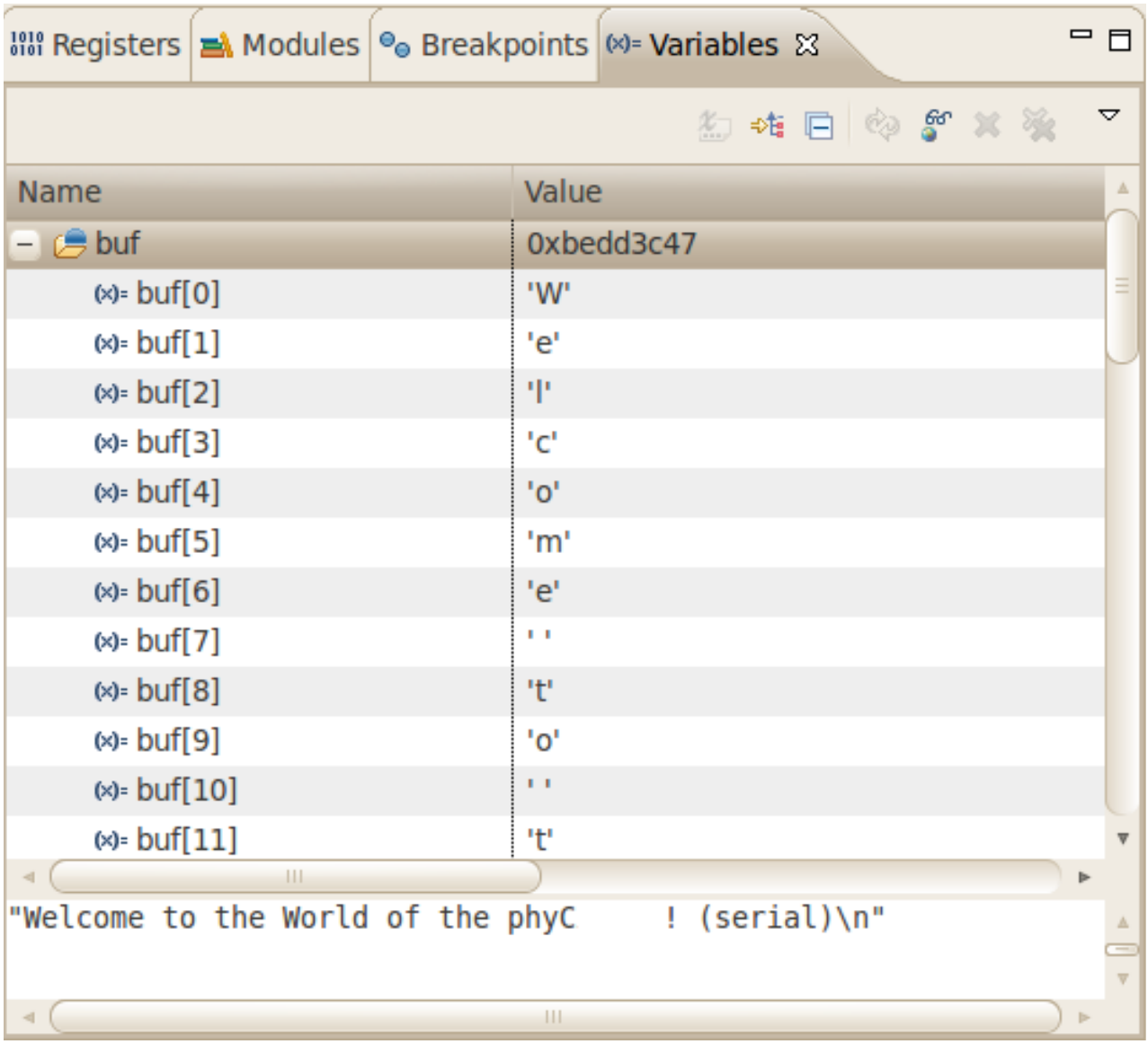
In this part you will step through the examples project with the debugger. You will also learn how to watch the content of a variable.

- Expand *buf* in the *Variables* window.



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

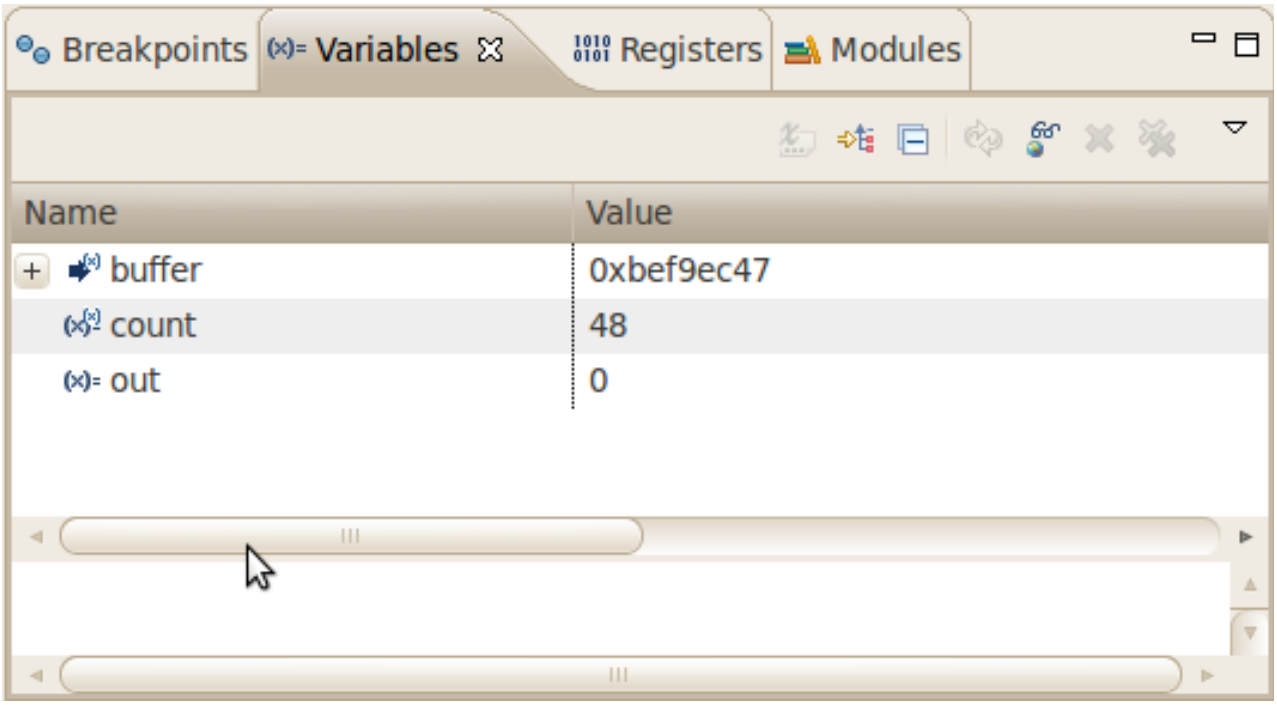
You will see the content of the *buf* variable in the *Variables* window.



- 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:



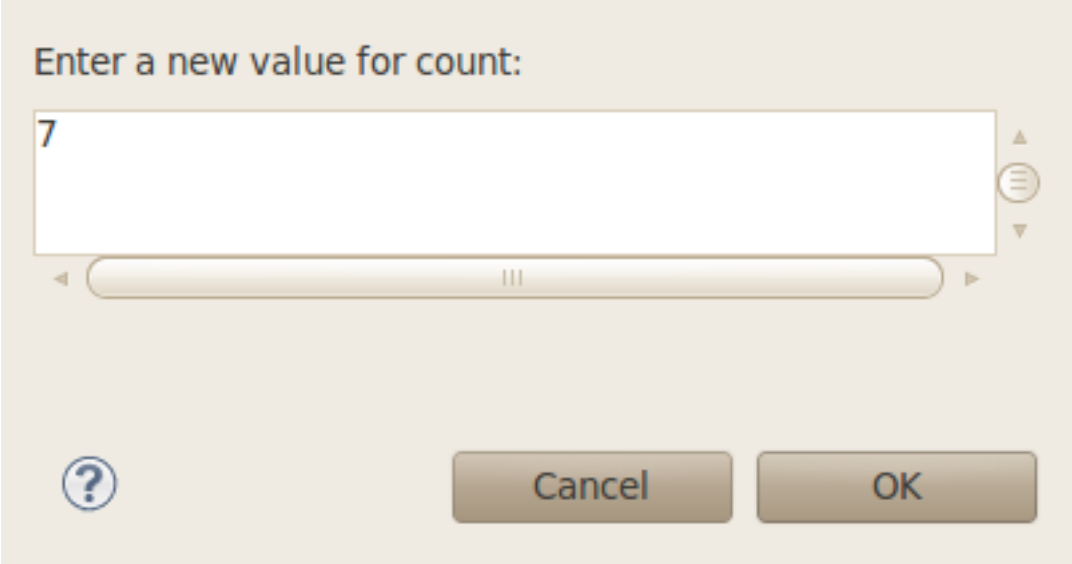
- Click on the variable *buffer*.

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

5.5 Stepping and Watching Variable Contents

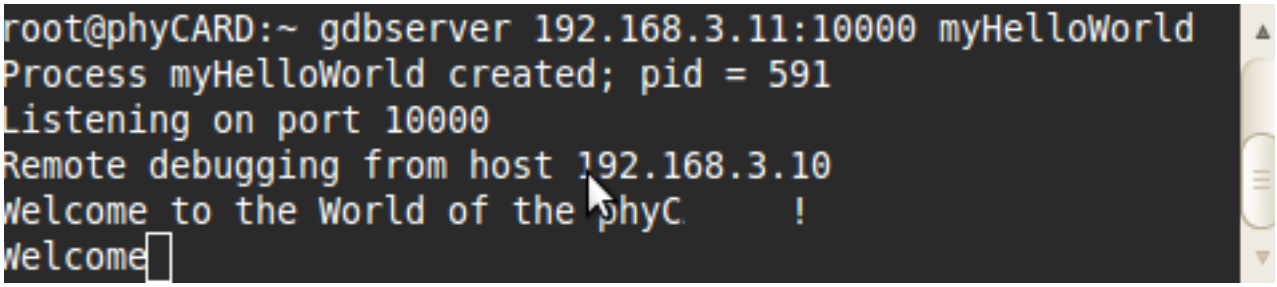
In this section you will change the value of a variable. At the end of this part you will see the effect of this change.

- 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*.

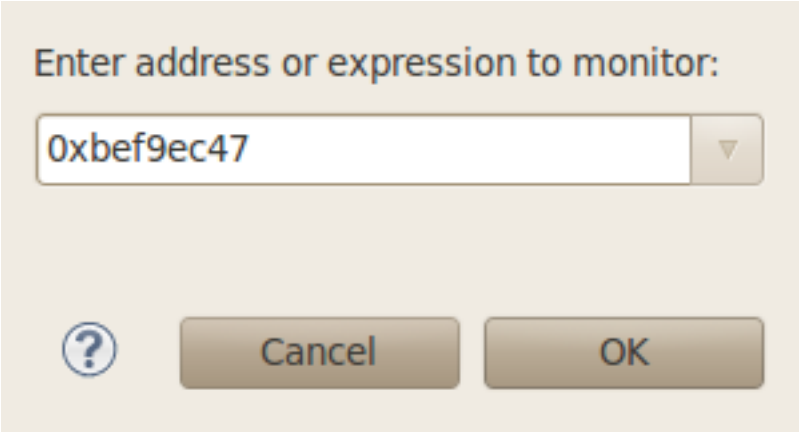


You will see the output *Welcome* in the Microcom window. This means that due to changing the *counter* variable's value, instead of printing the full sentence "Welcome to the World of the phyCORE-OMAP4" string, only the first seven characters of the buffer were written to the screen.

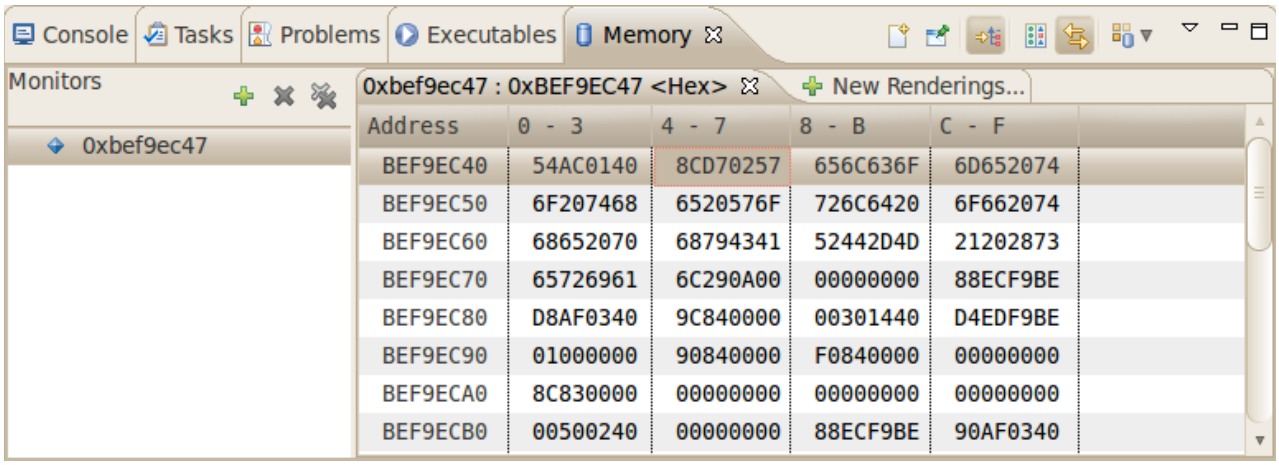
5.6 Using the Memory Monitor

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

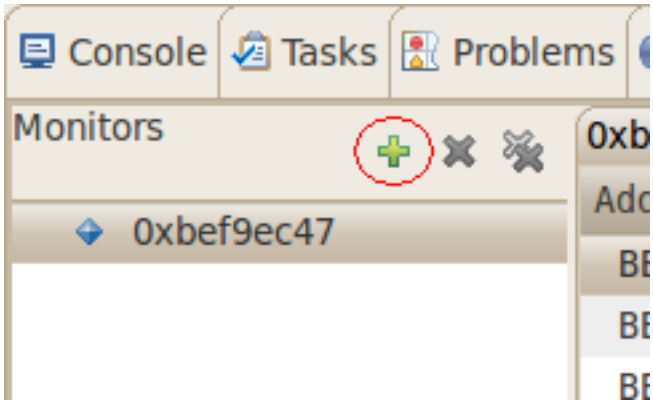
- Select the *Memory* tab.
- Click *Add Memory Monitor*.



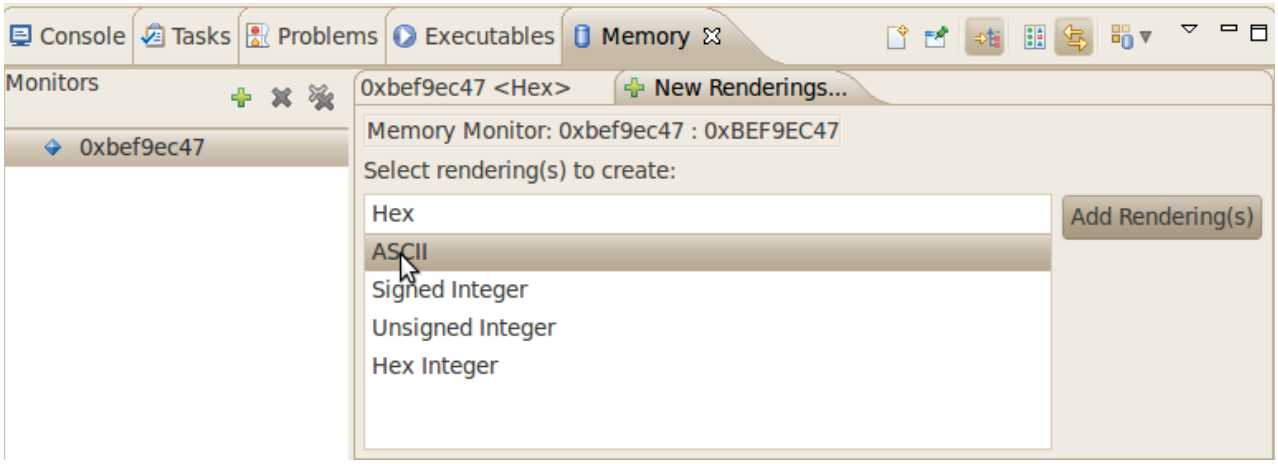
- Enter the address of buffer and click OK. Remember that the variable's address might differ on your system.



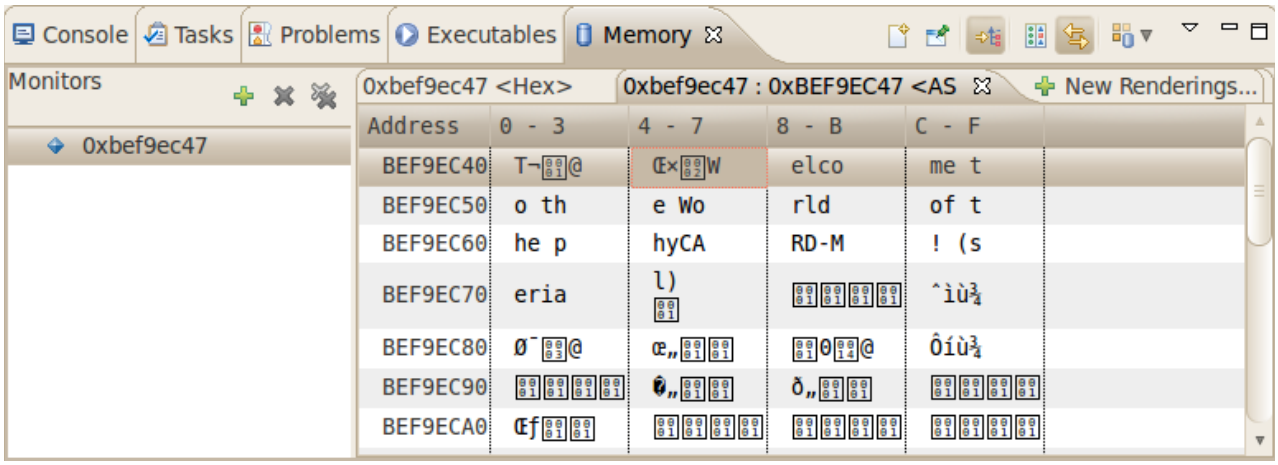
- Change the window size.



- Click *Add Rendering*.

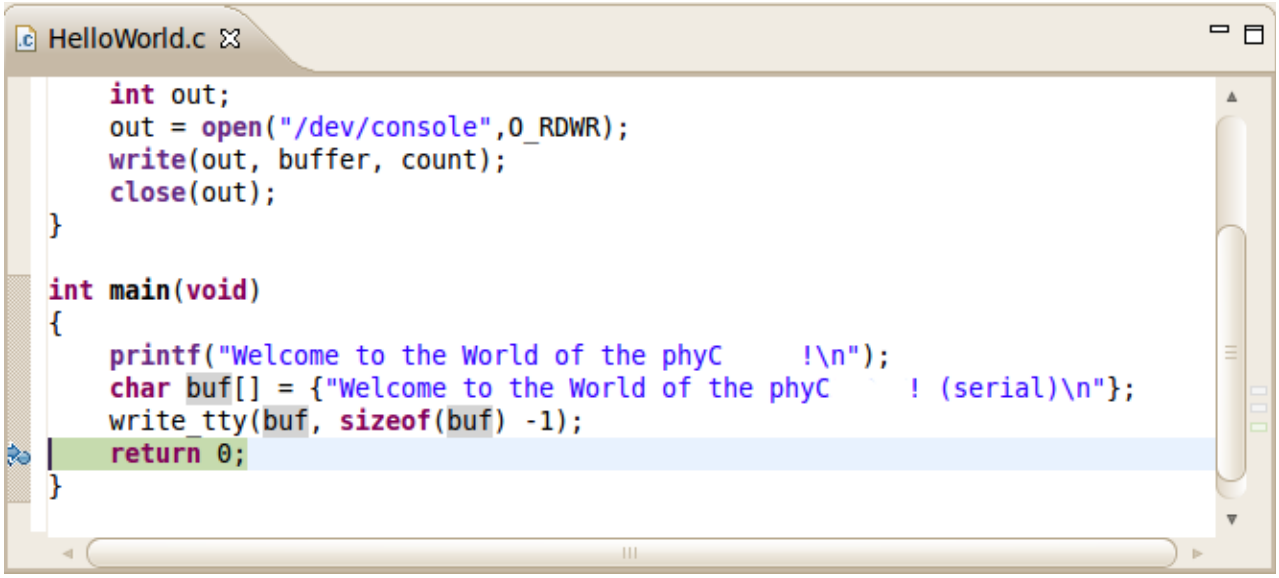


- Select *ASCII* and click *OK*.



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.



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.

6 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 phyCORE-OMAP4 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 phyCORE-OMAP4 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.

7 Installing Linux on the phyCORE-OMAP4

This part provides instructions on how to update bootloader (Barebox) on the phyCORE-OMAP4 and how to write a kernel and/or a root file system image into the target's flash memory.

7.1 Configure Barebox Environments Variables

The following steps will be done on a Linux platform. We assume that you have configured your host platform and executed the setup program provided on the Linux-phyCORE-OMAP4-Disc. Information on how to configure the host platform can be found in the *Getting Started* part of this Quick Start.

- Connect the serial cable with the UART1 (connector P1, TOP) on the target and the first serial interface on your host.
- Connect the cross-over Ethernet cable with the connector X27 on the target and the correct network card of your host.

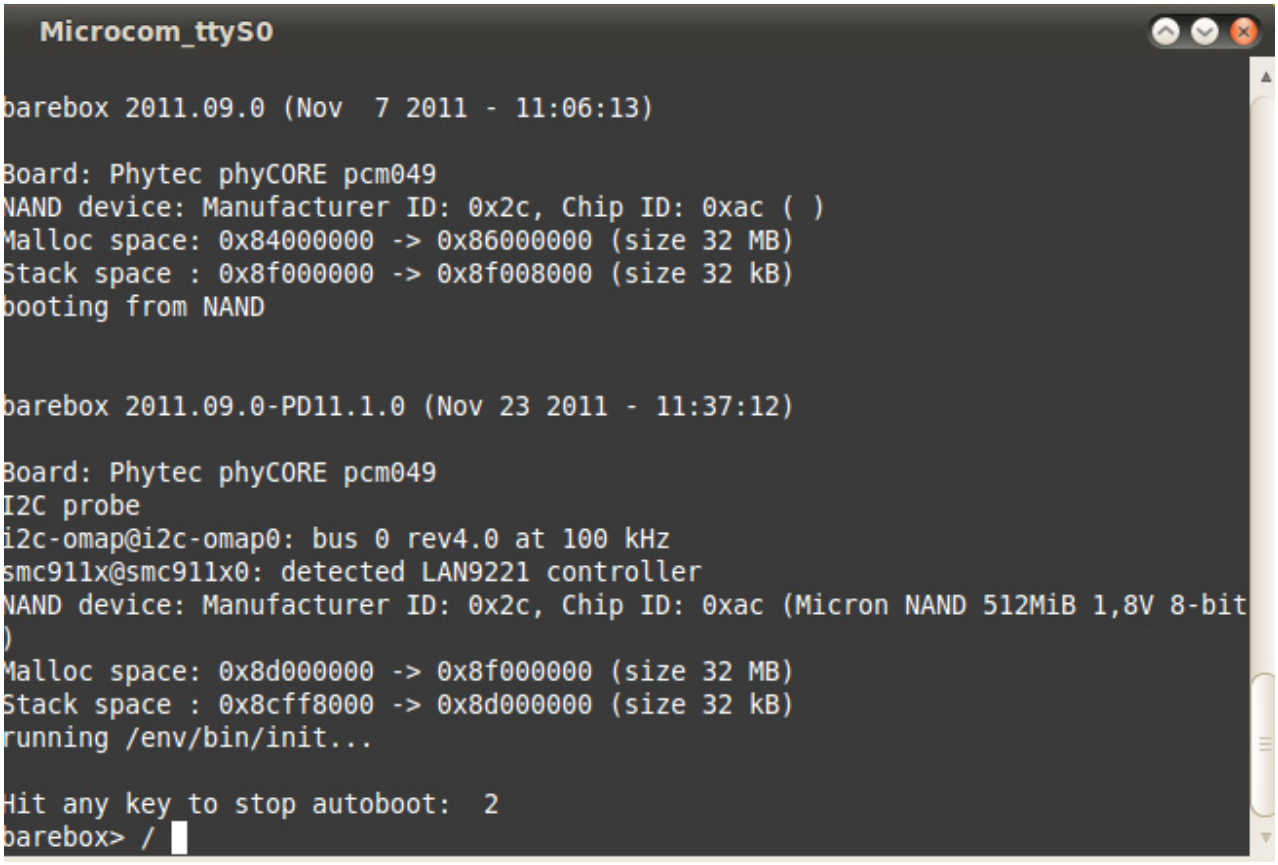
- Click the *Microcom* icon on your desktop.

Microcom is configured with the following configuration:

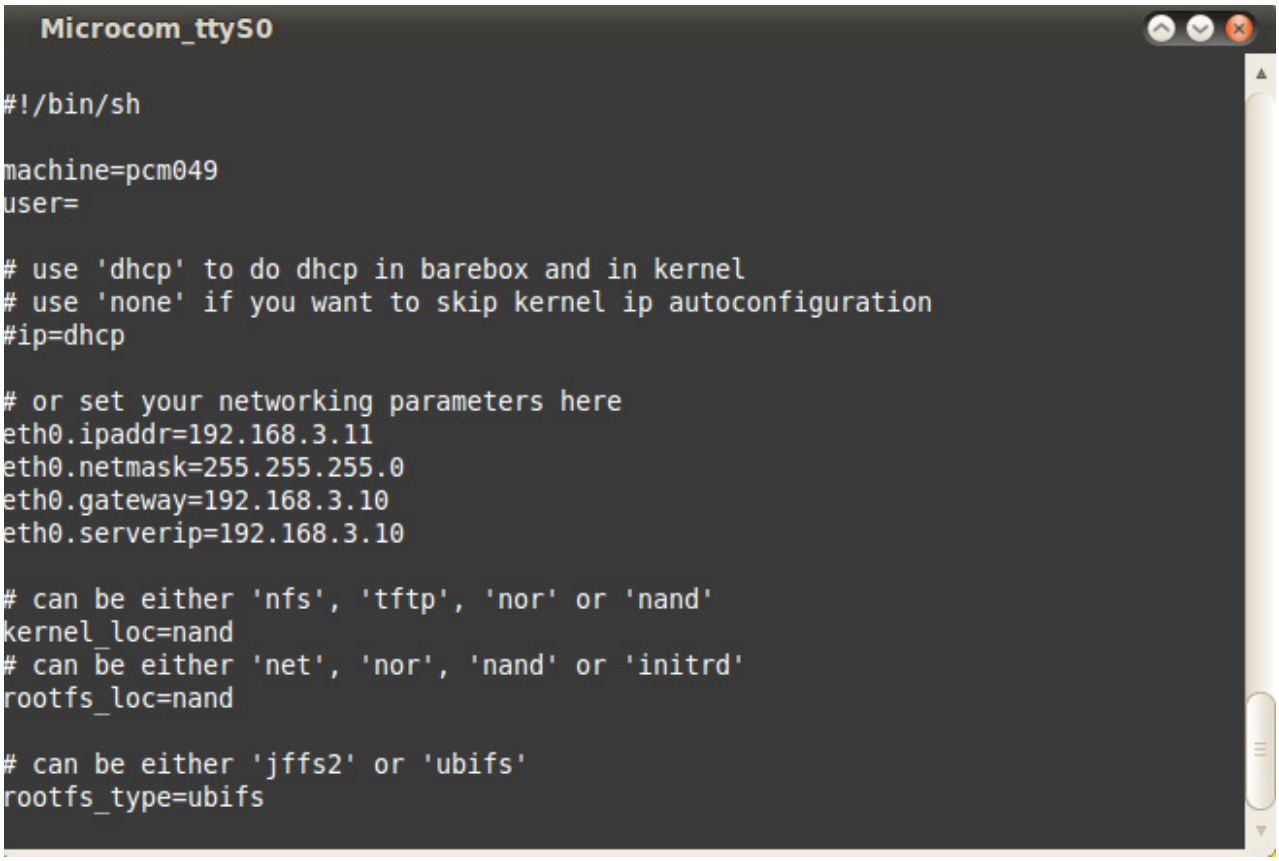
115200 baud, 1 start bit, 8 data bits, 1 stop bit, no parity, no flow control.

If you want to use a program other 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.



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



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.

7.2 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/nand0.bareboxenv**

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

7.3 Update the Bootloader

The bootloader used on the phyCORE-OMAP4 is *Barebox* which is already preinstalled. If nothing is installed or something went wrong while updating the bootloader please contact our support department. Before the new bootloader can be written into the target's flash, the target will have to download the image from a TFTP server. This will be done from the command line of the bootloader. First the image will be downloaded via TFTP from the directory */tftpboot* and copied into the target's RAM. Then the part of the flash where the bootloader should be written to must be erased. Finally the bootloader image can be transferred from RAM into flash.

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.

You need only one simple command to update the bootloader but before executing these commands 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.
- Type the following command to check your Barebox settings:
  edit /env/config
  You will see the configuration file which holds Barebox's environment variables.
- Make sure the following values are set within the configuration file:
  ```
  eth0.ippaddr=target IP address
  eth0.netmask=target netmask
  eth0.serverip=server IP address
  ```
- Type **CTRL-D** to save the settings to the file.
- If you have made any changes to the Barebox environment, type **save** to write these changes to the target's flash. Then press the RESET button on the target. The board reboots with the new settings applied. Then, again, press any key to stop autoboot.
- Type **update -t barebox -d nand -f barebox-image** to download and write the bootloader into target's flash.

- Press the RESET button on the board to restart your target.
  The target should now start with the new bootloader you have written into the flash.

7.4 Writing the Kernel / Root File System into Flash

Writing the kernel or root filesystem into the target's flash is principially the same like updating the bootloader.

In the directory */tftpboot* in your live environment or under */PHYTEC/BSP* on your DVD you can find a file called *uImage-pcm049* - this file is the Linux kernel image. There is another file, *root-pcm049.jffs2*; this file contains the Linux root filesystem.

You can download the kernel or root file system 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 these command, 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.
- Type the following command to check your Barebox settings:
  edit /env/config
  You will see the configuration file which holds Barebox's environment variables.
- Make sure the following values are set within the configuration file:
  ```
  eth0.ippaddr=target IP address
  eth0.netmask=target netmask
  eth0.serverip=server IP address
  ```
- Type **CTRL-D** to save the settings to the file.
- If you have made any changes to the Barebox environment, type **save** to write these changes to the target's flash. Then press the RESET button on the target. The board reboots with the new settings applied. Then, again, press any key to stop autoboot.
- Type **update -t kernel -d nand -f uImage-pcm049** to download and write the kernel into target's flash.
- Type **update -t rootfs -d nand -f root-pcm049.jffs2** to download and write the root file system into the target's flash.
  The copy process can take quite a while, depending on the speed of your system.
- Press the RESET button on the board to restart your target.

The target should now start with the kernel and root file system you have written into the flash.

8 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-phyCORE-OMAP4-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.

8.1 Essential settings

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

8.1.1 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
- Do 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.

8.1.2 Setup of toolchain, PTXdist and BSP

The important packages are installed and 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-phyCORE-OMAP4-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*.

If any error occurs during installation you will find detailed information in our OSELAS.phyCORE-BSP_Documentation at the Linux-phyCORE-OMAP4-Kit-DVD under /PHYTEC/Documentation.

On our Linux-phyCORE-OMAP4-Kit-DVD you will find under */PHYTEC/BSP* an already prebuilt Toolchain for the phyCORE-OMAP4 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 OSELAS.phyCORE-BSP_Documentation on the Linux-phyCORE-OMAP4-Kit-DVD under /PHYTEC/Documentation.

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/OSELAS.BSP*tar.gz
- The BSP is unpacked and we change into the created folder:
  cd OSELAS.BSP-Phytec-*
- The time has come to handle *PTXdist*. First of all we must specify our platform which we will work with:

ptxdist platform configs/phyCORE-OMAP4

- 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

You will also find more information concerning this topic in our OSELAS.phyCORE-BSP_Documentation on the Linux-phyCORE-OMAP4-Kit-DVD under /PHYTEC/Documentation.

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.

8.1.3 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 scroll-bar. 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** he begin 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:** *QT4.6.3* ► **Bin Path:** */opt/PHYTEC_BSPs/OSELAS.BSP-Phytec-phyCORE-PD11.1.0/platform-phyCORE-OMAP4/sysroot-cross/bin* ► **Include Path:***/opt/PHYTEC_BSPs/OSELAS.BSP-Phytec-phyCORE-PD11.1.0/platform-phyCORE-OMAP4/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.

8.1.4 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 phyCORE-OMAP4 can access to the /tftpboot/ folder to load the images.

8.2 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 which will be executed at the first start after installation in order to set the right permissions to the created user.

9 Installation of the modified Ubuntu

In this chapter you will find an instruction on how to install the modified Ubuntu on the Linux-phyCORE-OMAP4-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-phyCORE-OMAP4-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 them 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 finishes, the "Installation complete" window appears. Click **Restart now** to restart your computer. Ubuntu is now installed.

10 Revision History

Date	Version numbers	Changes in this manual
06-12-2011	Quickstart L-762e_0	First draft

