
QuickStart Instructions

Linux-PowerPC

phyCORE[®]-MPC5121e-tiny

Using Eclipse and the GNU Cross Development Toolchain

Note: The PHYTEC Linux-PowerPC-Disc includes the electronic version of the phyCORE-MPC5121e-tiny English Hardware Manual

Edition: December 2009

A product of a PHYTEC Technology Holding company

In this manual copyrighted products are not explicitly indicated. 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 believed 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 which 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 2009 PHYTEC Messtechnik GmbH, D-55129 Mainz.

Rights - including those of translation, reprint, broadcast, photomechanical 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 Technologie Holding AG Robert-Koch-Str. 39 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.com

1st Edition: December 2009

1	Introduction	1	
1.1	Rapid Development Kit Documentation	1	
1.2	Professional Support Packages available	2	
1.3	Overview of this QuickStart Instruction.....	2	
1.4	Conventions used in this QuickStart	3	5 min
1.5	System Requirements	4	
1.6	The PHYTEC phyCORE-MPC5121e-tiny	5	
1.7	Software Development Toolchains	8	
1.7.1	Eclipse.....	8	
1.7.2	The Gnu Cross Development Toolchain	9	
2	Getting Started.....	10	
2.1	Requirements of the Host Platform	10	
2.2	Configuring the Host Platform	11	
2.2.1	Installing Software packages:.....	11	35 min
2.2.2	Set up Network Card Configuration.....	17	
2.2.3	Disabling the Firewall	19	
2.2.4	Set up TFTP-Server	20	
2.3	Linux-PowerPC-Kit Setup.....	22	
2.4	Advanced Configuration Information	31	
2.5	Connecting the host with the target	33	
2.6	Copying an Example to the Target.....	38	
2.6.1	Copying a Program to the Target	38	
2.6.2	Using Telnet to execute a Program on the Target.....	41	
2.6.3	Using SSH to execute a Program on the Target.....	42	
2.7	Advanced Information	45	
2.7.1	Copying a program to the Target with the Command Line.....	45	
2.7.2	Executing a Program on the Target.....	45	
2.7.3	Executing a Program directly on the Target using SSH.....	45	
3	Getting More Involved.....	46	
3.1	Configuring and Compiling the Kernel.....	46	
3.2	Writing the Kernel into the Target's Flash.....	51	
3.3	Opening an Existing Project.....	56	
3.4	Creating a New Project.....	62	
3.5	Changing the Demo Application	73	
3.6	Starting a Program out of Eclipse on the Target	76	70 min
3.7	Automatically starting the Program when booting the Target	78	

4	Debugging an Example Project	83
4.1	Starting the GDB Server on the target.....	83
4.2	Configuring and Starting the Debugger in Eclipse	85
4.3	Setting a Breakpoint	90
4.4	Stepping and Watching Variables Contents.....	91
4.5	Changing Variables Values	93
4.6	Using the Memory Monitor.....	95
5	Further Information	98
6	Summary	99



20 min

1 Introduction



5 min

In this QuickStart you can find general information information on the PHYTEC phyCORE[®]-MPC5121e-tiny and an overview of the Eclipse software development tool and GNU GCC C/C++ Cross-Development Toolchain. You can also find instructions on how to run example programs on the phyCORE[®]-MPC5121e-tiny, mounted on the PHYTEC phyCORE[®] Development Board MPC5121e-tiny, in conjunction with the Eclipse development tool.

Please refer to the [phyCORE-MPC5121e-tiny Hardware Manual](#) for specific information on such board-level features as [jumper configuration](#), [memory mapping](#) and [pin layout](#).

1.1 Rapid Development Kit Documentation

This "Rapid Development Kit" includes the following electronic documentation on the enclosed "PHYTEC Linux-PowerPC-Disc":

- PHYTEC [phyCORE-MPC5121e-tiny Hardware Manual](#) and [Development Board Hardware Manual](#)
- MPC5121e-tiny controller [User's Manuals and Data Sheets](#)
- this QuickStart Instruction with general "Rapid Development Kit" description, software installation advice and an example program enabling quick out-of-the box start-up of the phyCORE[®]- MPC5121e-tiny in conjunction with the Eclipse / GNU GCC C/C++ software development toolchain

1.2 Professional Support Packages available

This Kit comes with free installation support. If you have any questions concerning installation and setup, you're 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.

1.3 Overview of this QuickStart Instruction

This QuickStart Instruction gives 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[®]-MPC5121e-tiny in conjunction with the Eclipse IDE and GNU GCC/C++ software tools. It is structured as follows:

- 1) The "*Getting Started*" section describes the configuration of the host platform and the setup to install the tools used in this QuickStart.
- 2) The "*Getting More Involved*" section provides step-by-step instructions on how to configure and build a new kernel, modify the example, create and build new projects and copy output files to the phyCORE[®]-MPC5121e-tiny using Eclipse.
- 3) The "*Debugging*" section provides information on how to debug an application with the Eclipse debugging interface.

In addition to the dedicated data for this Rapid Development Kit, the PHYTEC Linux-PowerPC-Disc contains supplemental information on embedded microcontroller design and development.

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 corresponds 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 part you might leave the path of this QuickStart.



This is a warning. It helps you to avoid annoying problems.



Provides useful supplementary information about the topic.



At the beginning of each chapter you can find information of the time to pass the following chapter.



You have successfully passed an important part of this QuickStart.



Provides information to solve problems.

1.5 System Requirements

Use of this "Rapid Development Kit" requires:

- the PHYTEC phyCORE-MPC5121e-tiny,
- the PHYTEC Development Board with the included DB-9 serial cable, AC-to-DC adapter supplying 12 V DC/min. 2 A,
- PHYTEC Distribution based on OSELAS from Pengutronix
- an IBM-compatible host-PC (586 or higher)
- openSUSE Linux 11.0 OSS (x86) and KDE 3.5 desktop
- recommended free disk space: 2 GB

For more information and example updates, please refer to the following sources:

PHYTEC

<http://www.phytec.de>
support@phytec.de



<http://www.freescale.com/>

1.6 The PHYTEC phyCORE-MPC5121e-tiny

The phyCORE-MPC5121e-tiny represents an affordable yet highly functional Single Board Computer (SBC) solution in the size 60 x 76 mm² for the tiny-Board. The standard board is populated with a [Freescale PowerPC Microcontroller MPC5121e](#) featuring a 32-bit processor architecture with Double precision FPU, 400 MHz processor speed, Peripheral component interconnect (PCI) controller, ATA controller, Fast Ethernet controller (FEC), DMA subsystem, 12 programmable serial controllers (PSC) configurable for the following functions: UART , AC97 and Codec (Soft Modem, ESAI, SPI, I2S).

All applicable LocalPlus data/address lines and applicable signals extend to two high-density 160-pin Molex SMT pin header connectors (pin width is 0.635 mm./25mil) lining the circuit board edges. This enables the phyCORE-MPC5121e-tiny to be plugged like a “big chip” into target hardware.

The standard memory configurations of the phyCORE-MPC5121e-tiny features 128MByte DDR-SDRAM and 16MByte external Flash. The external Flash supports direct on-board programming without additional programming voltages.

phyCORE-MPC5121e-tiny Technical Highlights

- phyCORE dimensions 60 x 76 mm with two high-density 160-pin Molex SMT pin header connectors
- Processor: Freescale Embedded PowerPC MPC5121e, 400 MHz clock

Internal Features of the MPC5121e:

- e300 Power Architecture processor core
 - 760 MIPS at 400 MHz (-40 to +85 °C)
 - 32 k instruction cache, 32 k data cache
- Power modes include doze, nap, sleep, deep sleep, and hibernate
- AXE – Auxiliary Execution Engine
- MBX Lite – 2D/3D graphics engine (not available in MPC5123)

- DIU – Display interface unit
- DDR1, DDR2, and LPDDR/mobile-DDR SDRAM memory controller
 - Multi port controller, listening on five incoming ports
- Flexible multi-function external bus interface
- MEM – 128 Kbyte on-chip SRAM
- USB 2.0 OTG controller with integrated physical layer (PHY)
- DMA subsystem
- EMB – Flexible multi-function external memory bus interface
- NFC – NAND flash controller
- LPC – LocalPlus interface
- FEC – Fast Ethernet controller
 - Supports 100Mbps IEEE 802.3 MII, 10 Mbps IEEE 802.3 MII
- PCI interface, version 2.3
- PATA – Parallel ATA integrated development environment (IDE) controller
- SATA – Serial ATA controller with integrated physical layer (PHY)
- SDHC – MMC/SD/SDIO card host controller
- PSC – 12 programmable serial controllers
 - configurable for the following protocols: UART, Codec/PCM, serial audio data, I2S, multi-channel data, SPI, and AC97
- I2C – 3 inter-integrated circuit communication interfaces
- S/PDIF – Serial audio interface
- CAN – Controller area network
 - Implementation of CAN protocol, version 2.0 A/B
- BDLC – J1850 interface
- VIU – Video Input, ITU-656 compliant
- RTC – On-Chip real-time clock
- On-chip temperature sensor
- IIM – IC Identification module
- IEEE 1149.1 compliant JTAG boundary scan

The PHYTEC Development Board, in card dimensions (240 x 240 mm²/9.44 x 9.44 in².), is fully equipped with all mechanical and electrical components necessary for the speedy and secure insertion of PHYTEC phyCORE-MPC5121e-tiny Single Board Computer.

Development Board Technical Highlights

- reset and power push button
- CPLD with power logic
- two software programmable LED for processor/peripheral
- LEDs for supply voltage monitoring
- power supply for regulated input voltage of +12V. It supplies regulated +3.3 V for the phyCORE-MPC5121e-tiny and further supply voltages.
- two DB-9 sockets for the RS-232 interface
- two DB-9 plugs for two separate CAN interfaces
- Ethernet socket
- USB AB Interface for the phyCOREs Hig Speed OTG Interface
- Compact Flash Card Socket
- 2x PCI Socket
- ATA-Interface
- SATA Interface
- Graphic Interface (TTL/LVDS)
- Audio Interface with WM9712G
- ATX Power Supply sockets
- Expansion Connector

1.7 Software Development Toolchains

1.7.1 Eclipse

The Eclipse Platform provides support for C/C++ development. Because the Eclipse Platform is only a framework for developer tools, it doesn't support C/C++ directly; it uses external plug-ins for support. This QuickStart shows you how to make use of the CDT -- a set of plug-ins for C/C++ development in conjunction with the GNU GCC C/C++ Toolchain.

The CDT is an open source project (licensed under the Common Public License) implemented purely in Java as a set of plug-ins for the Eclipse SDK Platform. These plug-ins add a C/C++ Perspective to the Eclipse Workbench that can now support C/C++ development with a number of views and wizards, along with advanced editing and debugging support.

Due to its complexity, the CDT is broken down into several components, which take the form of separate plug-ins. Each component operates as an autonomous project, with its own set of committers, bug categories, and mailing lists. However, all plug-ins are required for the CDT to work properly. Here is a list of the plug-ins/components:

- **Primary CDT plug-in** is the "framework" CDT plug-in.
- **CDT Feature Eclipse** is the CDT Feature Component.
- **CDT Core** provides Core Model, CDOM, and Core Components.
- **CDT UI** is the Core UI, views, editors, and wizards.
- **CDT Launch** provides the launch mechanism for external tools such as the compiler and debugger.
- **CDT Debug Core** provides debugging functions.
- **CDT Debug UI** provides the user interface for the CDT debugging editors, views, and wizards.
- **CDT Debug MI** is the application connector for MI-compatible debuggers.

1.7.2 The Gnu Cross Development Toolchain

Cross-development in general refers to the overall software development process that produces a single application or a complete system running on a platform that is different from the development platform. This is an important concept when the target system doesn't have a native set of compilation tools, or when the host system is faster and has greater resources.

The platform where the actual development takes place is called the host platform. The platform where the final application is tested and run is called target platform. In this QuickStart we are using a x86-architecture based Linux as host platform. As target platform we are using the power-pc architecture with a MPC5221e CPU.

Building a program for a CPU architecture different from the one used on the machine where the compilation is done, is accomplished using a cross-compiler toolchain and cross-compiled libraries. In this QuickStart we are using the GNU C/C++ Cross Development Toolchain.

2 Getting Started

**35 min**

TIME

In this chapter you will establish the basis to pass the steps in this QuickStart. First you will learn how to configure the host platform. You will install additional software packages and setup the network configuration for connecting your host to the target. After connecting the host to the target you will copy an application to the target. At the end of this chapter you will be able to start a first demo application on the target.

2.1 Requirements of the Host Platform

To pass the following steps in this Quickstart, you will need a host pc with an installation of openSUSE 11.0 OSS (x86) and KDE 3.5 desktop.

When you are installing openSUSE 11.0, you can select *KDE 3.5 as Desktop selection*. The default packages to use openSUSE 11.0 with your host pc will be selected automatically. This default selection will suffice to pass the steps in this QuickStart Instruction. The installation of additional packages and configurations will be described on the following pages.

In the following configuration steps we assume, 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 this QuickStart Instruction you will have to shutdown the firewall and configure the network card of your host pc. If your host pc is connected to another network, changing the ip address can cause conflicts with existing hosts.

2.2 Configuring the Host Platform

In this passage you will learn how to configure the host platform. You will execute the following steps in this passage:

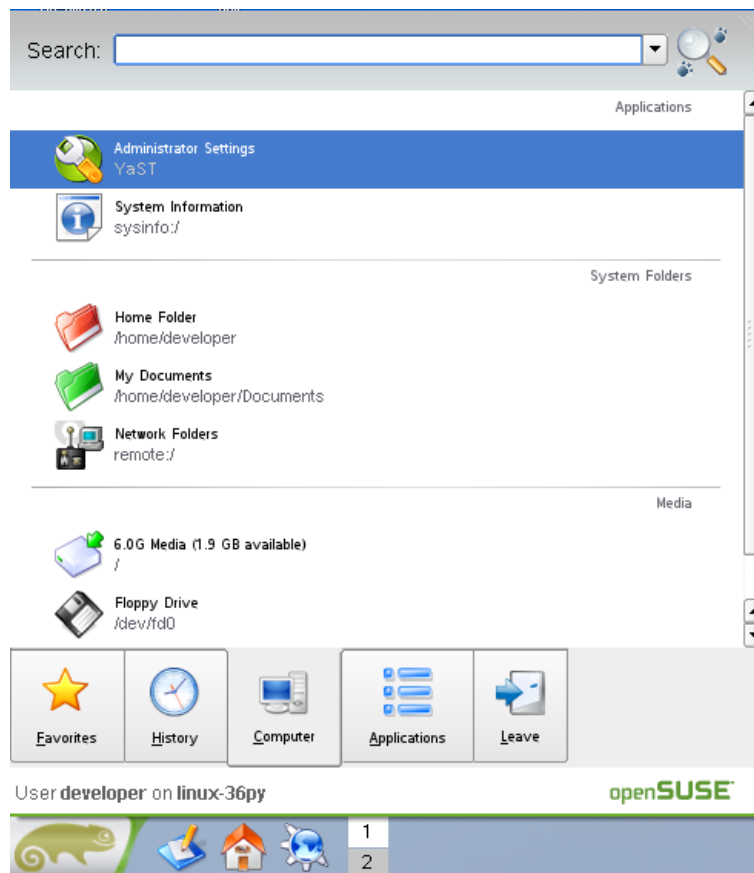
- Installing additional software packages. These packages are necessary to accomplish the steps in the QuickStart Instruction.
- Setup the network configuration to use the host pc with your target.
- Disabling the firewall. If the firewall is enabled, you will have problems with connections to the target.
- Setup a TFTP-Server. You can use a TFTP Server to download files (e.g. a Kernel images or root file systems) into the RAM of the target from the bootloader.

2.2.1 Installing Software packages:

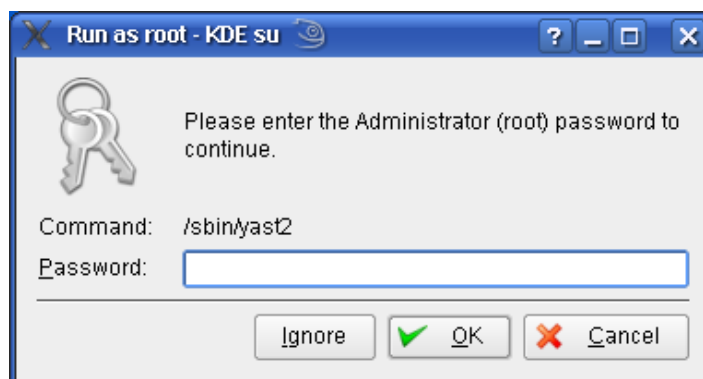
To accomplish the steps in the QuickStart Instruction you will have to install additional packages.



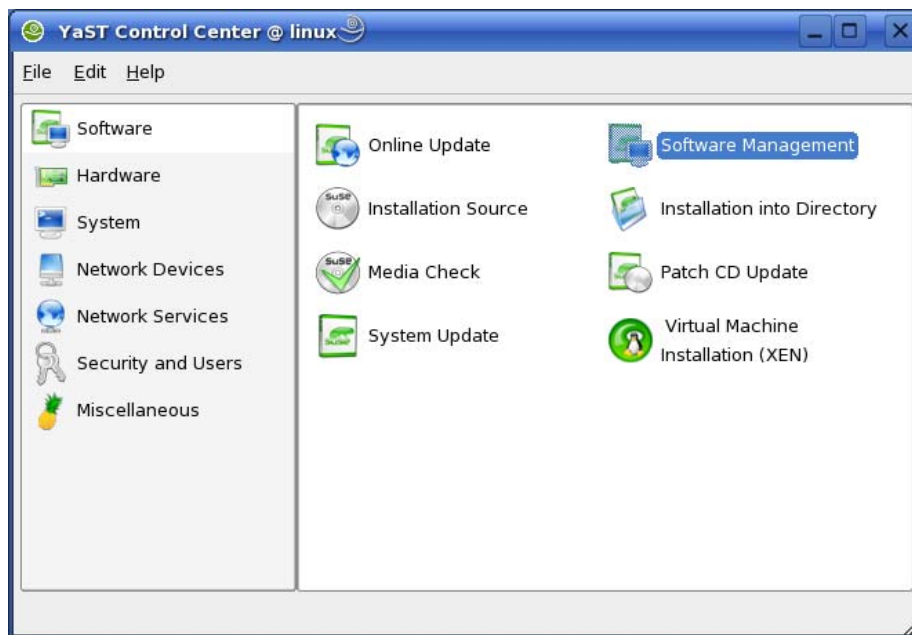
If you don't install all of these packages, the setup may fail or some configuration steps won't work correctly.



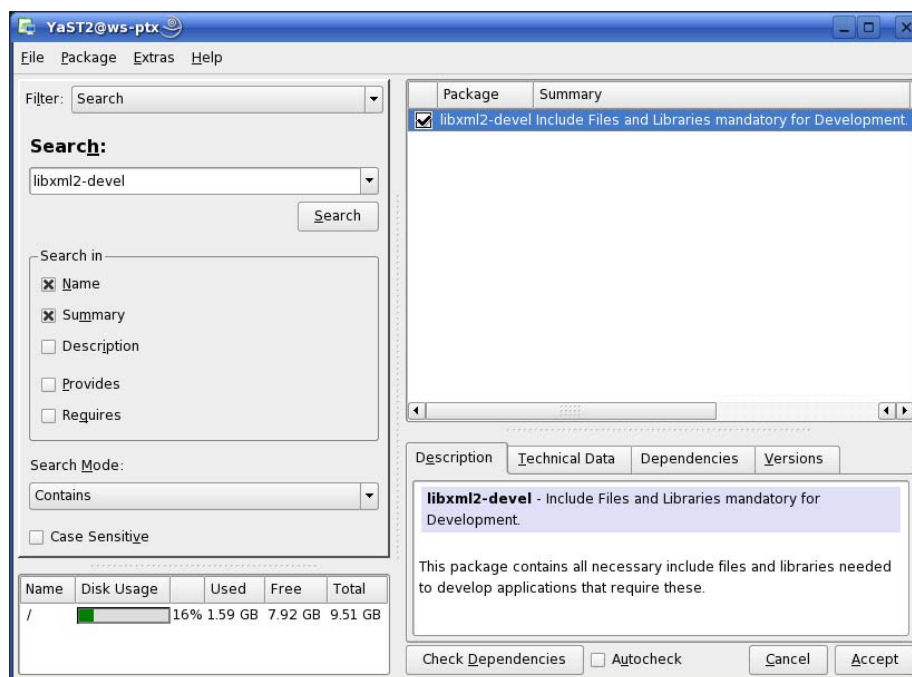
- Open the *K menu* from the lower-left corner of the desktop and click on the tab *Computer*.
- Open the *Administrator Settings / YaST*.



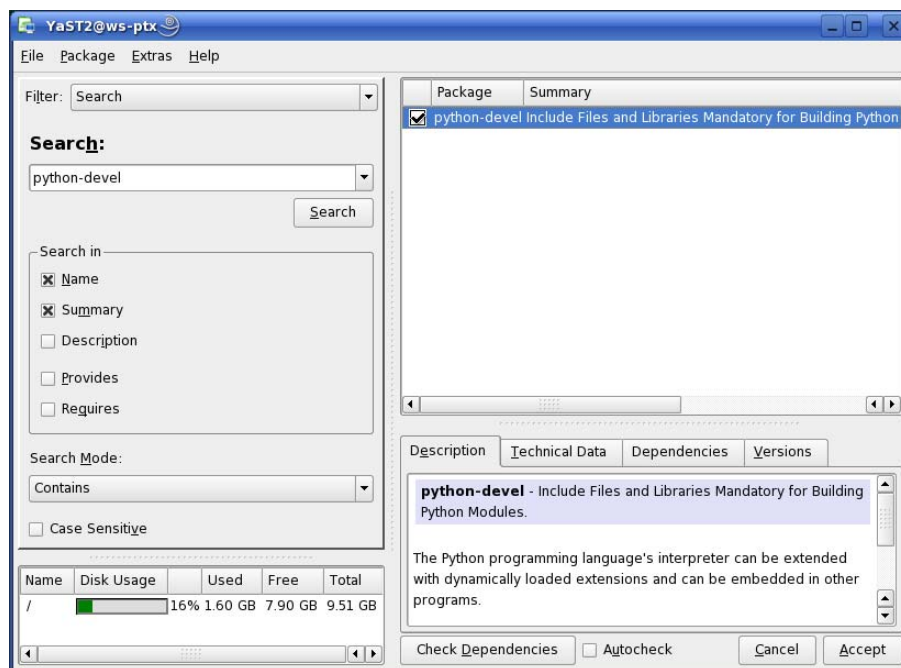
- Enter your root password and click *OK*.



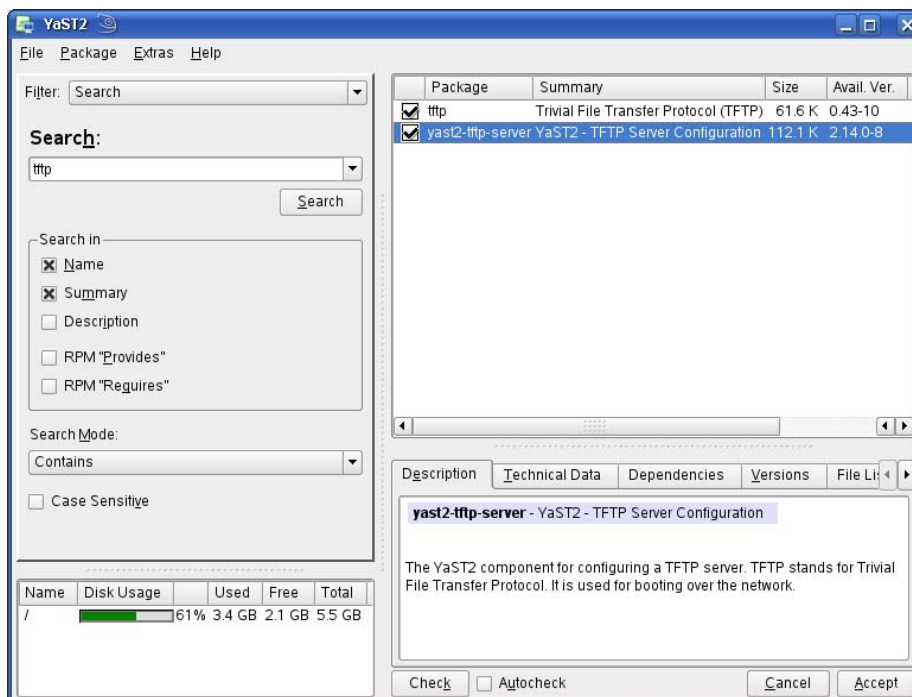
- Open *Software Management* in *Software*.



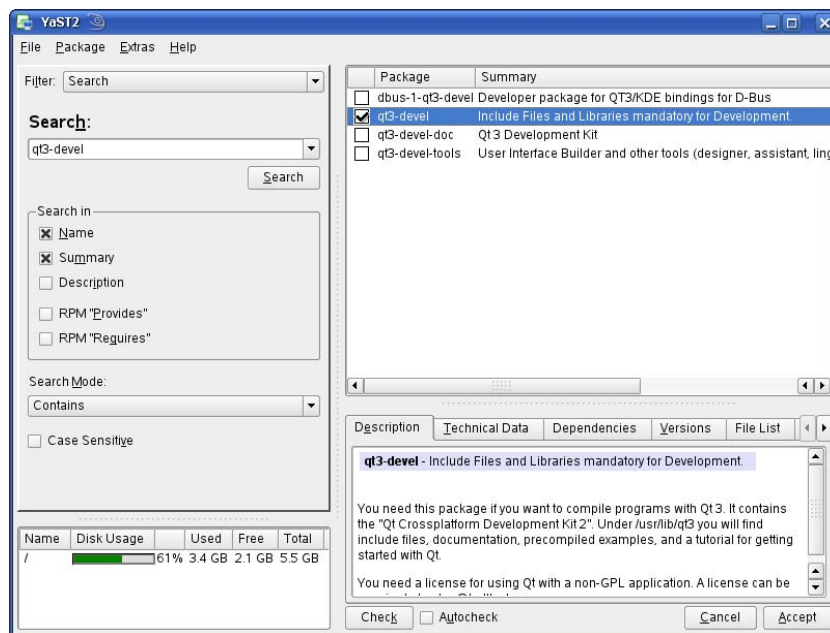
- Select the filter *Search*.
- Type **libxml2-devel** and click *Search* button.
- Check *libxml2-devel*.



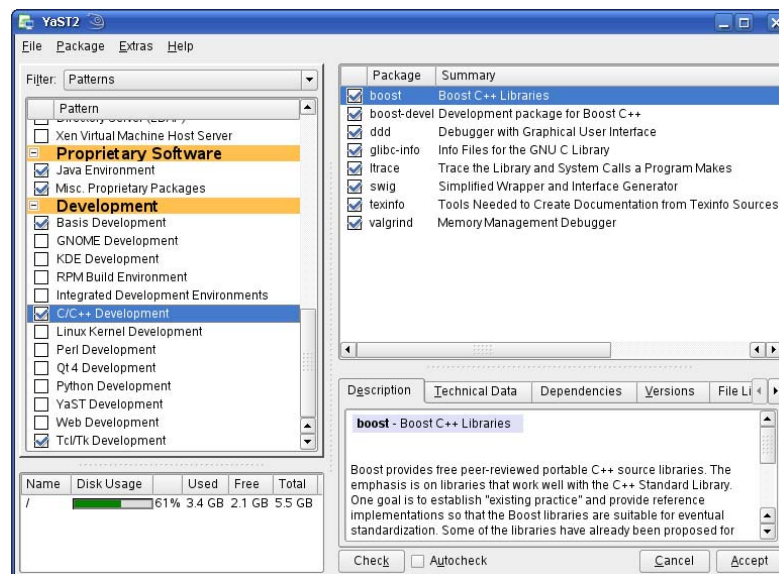
- Type **python-devel** and click *Search* button.
- Check *python-devel*.



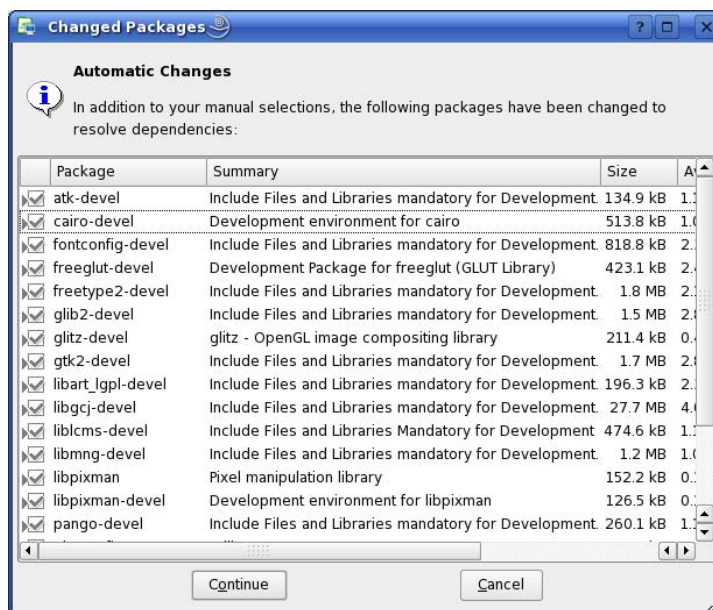
- Type **tftp** and click *Search* button.
- Check the packages *tftp* and *yast2-tftp-server*.



- Type **qt3-devel** and click *Search* button.
- Check **qt3-devel**.



- Select the filter *Patterns*.
- Select *Basis Development*, *C/C++ Development*, and *Tcl/Tk Development*.
- Click *Accept*.



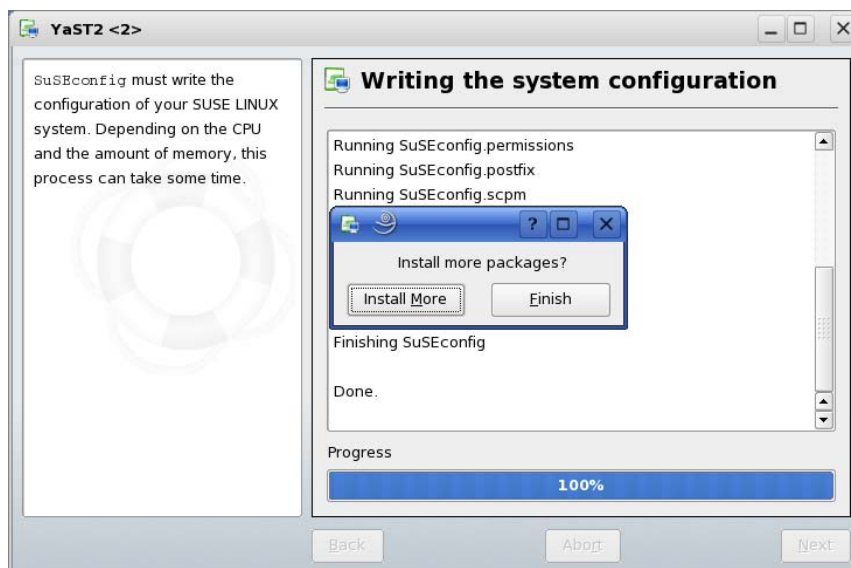
Some additional packages will be selected automatically to resolve dependencies.



If problems occur while resolving dependencies, we recommend going back to a default configuration.

CAUTION!

- Click on *Continue* to install the packages.



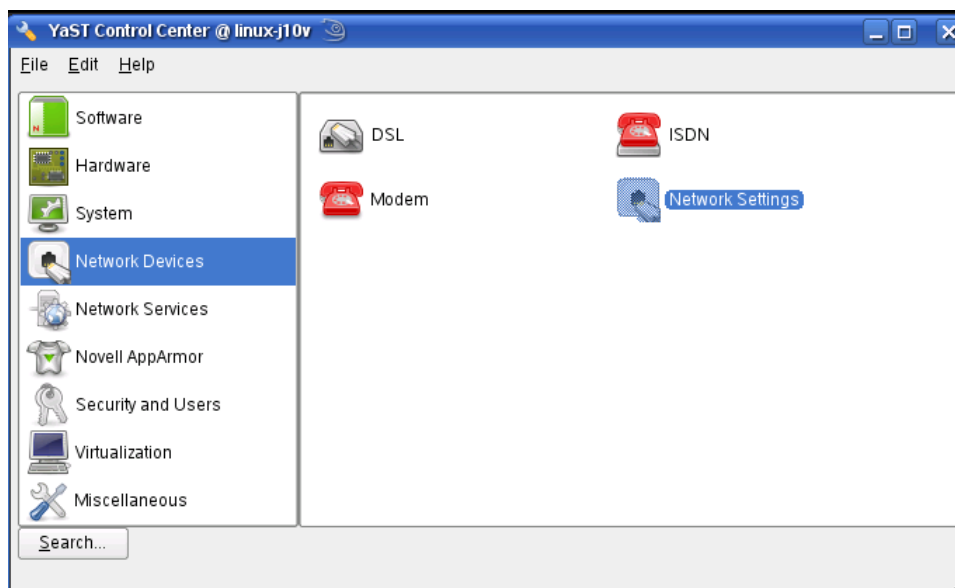
- Click *Finish*.

2.2.2 Set up Network Card Configuration

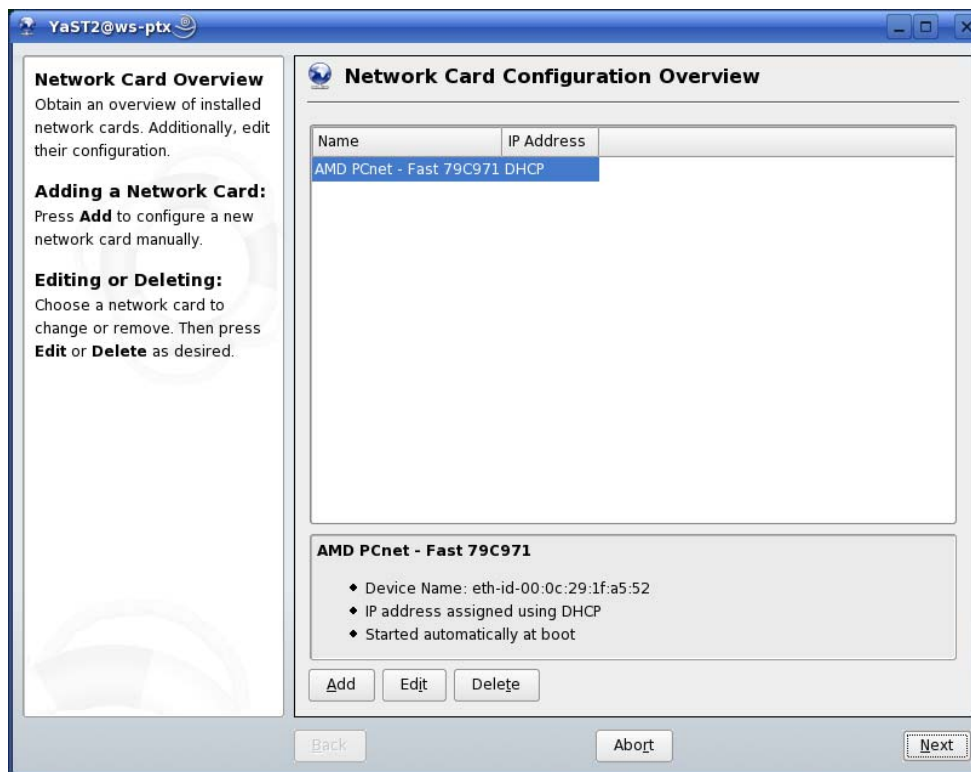


In the following step you will have to configure the ip address of your host. We recommend disconnecting your host from any other network. If you change the host's ip, chances are that problems may occur with other hosts in the network.

- Open the YasST Control Center if it is not already open.



- Choose *Network Settings* in *Network Devices*.

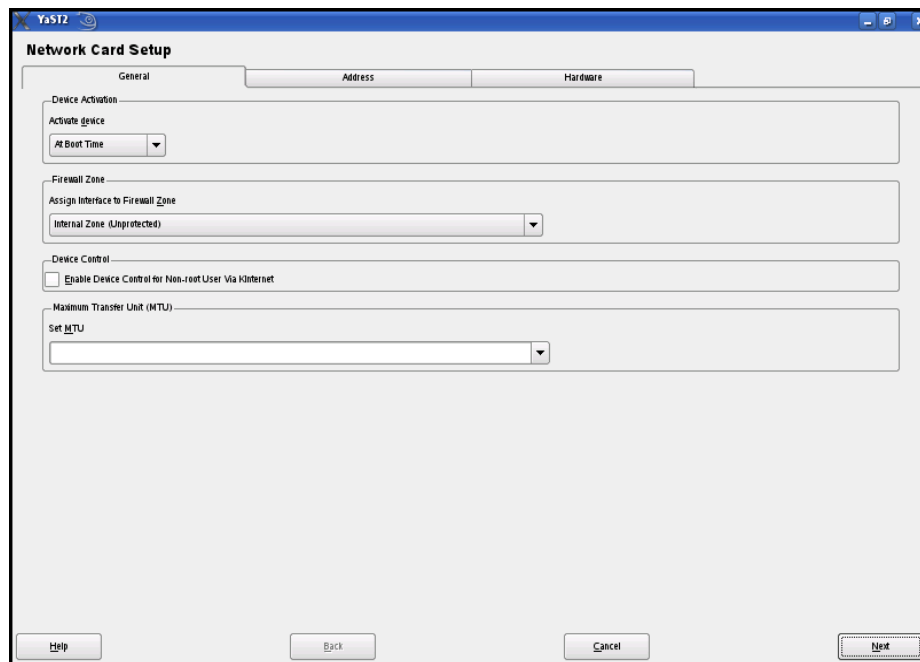


- Select the right network card, if more than one network card is installed on your host.
- Click *Edit* to enter the *Network Card Setup*.
- Choose *Static address setup*.
- Enter IP Address **192.168.3.10** and Subnet Mask **255.255.255.0**

2.2.3 Disabling the Firewall

To ensure that there are no problems with connections to the target, the host's firewall should be disabled.

- Select the General tab in the upper-left corner.



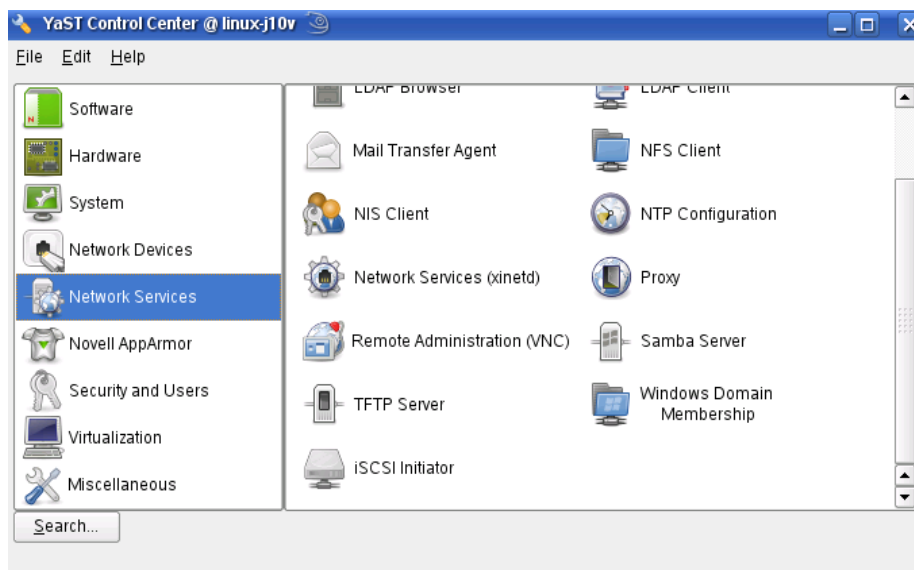
- Use the drop-down box in the *Firewall Zone* settings to set the current interface to *Internal Zone (Unprotected)*.
- Then press Next, and in the following window click *Finish* to complete the settings.

The firewall is now disabled for this network card.

2.2.4 Set up TFTP-Server

Later in this QuickStart you will learn how to write a new kernel image into the flash memory of the target. To download the kernel image from the target you need to have a tftp server up and running. In this passage we show you how to configure a tftp server.

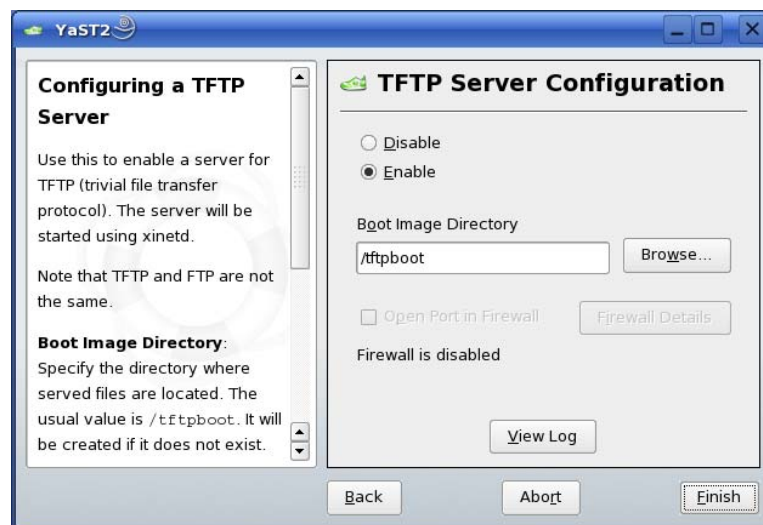
- Open the YaST Control Center if it is not already open.



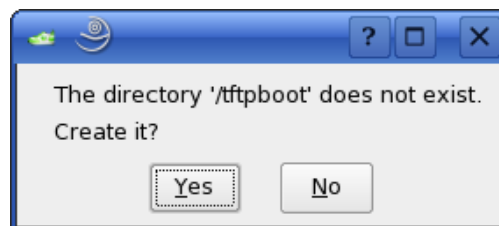
- Choose *TFTP Server* in *Network Services*.



If the *TFTP Server* icon does not exist, restart the YaST Control Center.



- Switch the selection to *Enable*.
- The path of the boot image directory should be */tftpboot*. If there is a different path, change it to */tftpboot*.
- Click Finish.



- Click *Yes* to create the */tftpboot* directory.
- The TFTP Server will be started.
- Close the YaST Control Center.



You have successfully finished the configuration of the host platform.

2.3 Linux-PowerPC-Kit Setup

In this section you will find a description of the Linux-PowerPC-Kit setup. The whole setup will be done by a graphical interface. At the end of the setup, you will find all programs to develop applications for the target on your host PC.

The setup contains the following programs:

- *GNU C/C++ cross development tool chain* – you can use this tool chain to develop programs for the target on your host PC.
- *Eclipse SDK with CDT* – the Eclipse SDK is a platform and application framework for building software which can use the GNU C/C++ cross development tool chain.
- *Microcom* – a program for serial communication with the target.
- *Linux Kernel archive* – this kernel archive contains the Linux kernel source code as well as all patches needed to compile the kernel for the phyCORE[®]MPC5121e-tiny.
- *HelloWorld* – this example program can be used to test how to download and execute a program on the target.

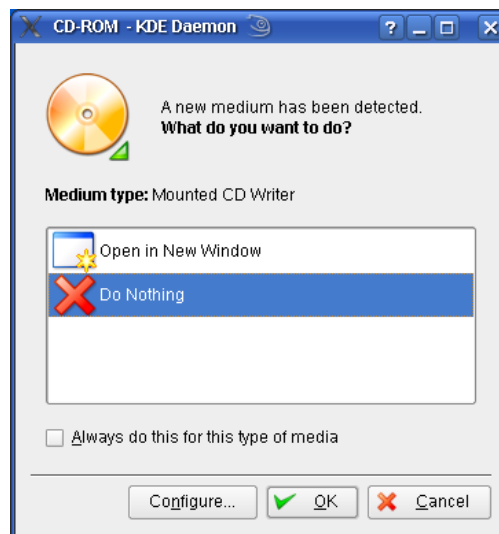
There will be some additional configuration on your host PC:

- The setup program will create desktop links to the installed programs.
- The setup will also create desktop links to access the target via FTP, SSH, and Telnet.
- The path of the cross development tool chain will be added to the \$PATH environment variable.
- Read and write access to the serial interface will be added to your user account so you use the serial communication program Microcom.
- The setup will configure Microcom.

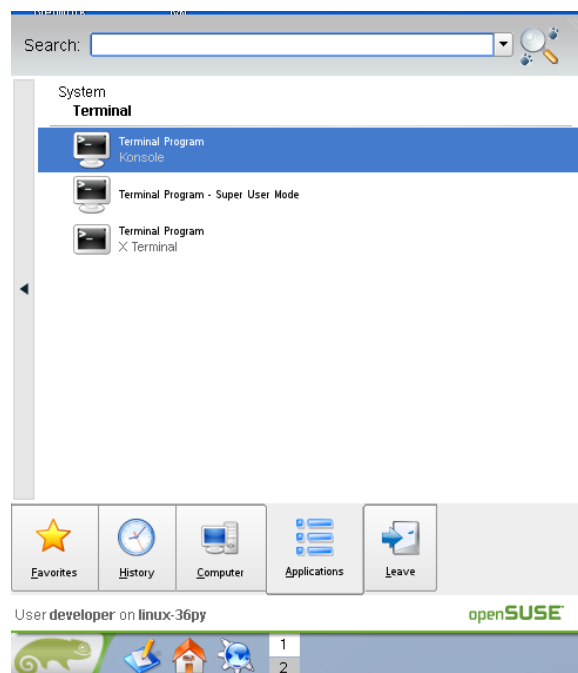
Starting the Setup:

- To start with the Linux-PowerPC-Kit Setup enter your PHYTEC Linux-PowerPC-Disc into your CD-ROM drive.

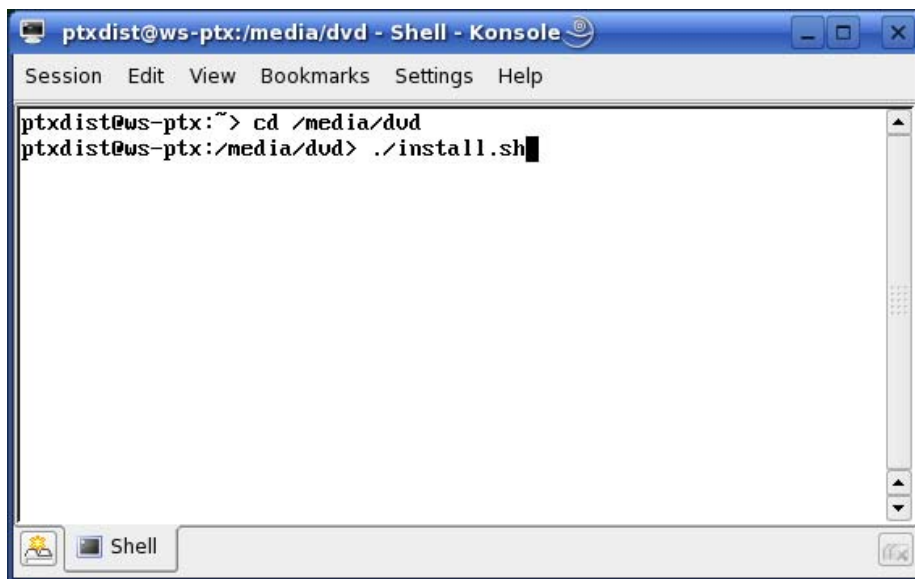
The following dialog probably appears:



- Click *Cancel*.



- From the *K menu*, select the *Applications* tab.
- Select *System* ► *Terminal* ► *Terminal Program / Konsole*.



```
ptxdist@ws-ptx:/media/dvd - Shell - Konsole
Session Edit View Bookmarks Settings Help
ptxdist@ws-ptx:~> cd /media/dvd
ptxdist@ws-ptx:/media/dvd> ./install.sh
```

In the next step you will have to change to the directory accordant to your drive. If you use a dvd drive change to the directory */media/dvd*. If you have installed a cdrom drive in your host, you will have to change to the directory */media/cdrom*.

- Type

cd /media/cdrom

or

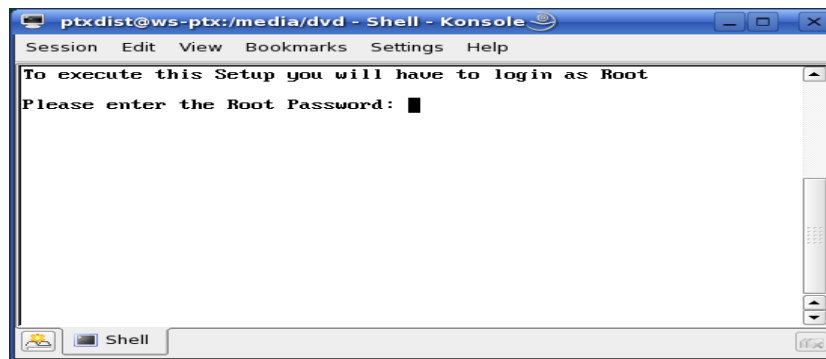
cd /media/dvd

accordant to your drive.



The media may be mounted on a different mount point in the directory */media*. The mount points can be shown with the command **ls /media**. Change to the accordant directory if no directory *cdrom* or *dvd* should exist.

- Enter **./install.sh** to start the setup program.



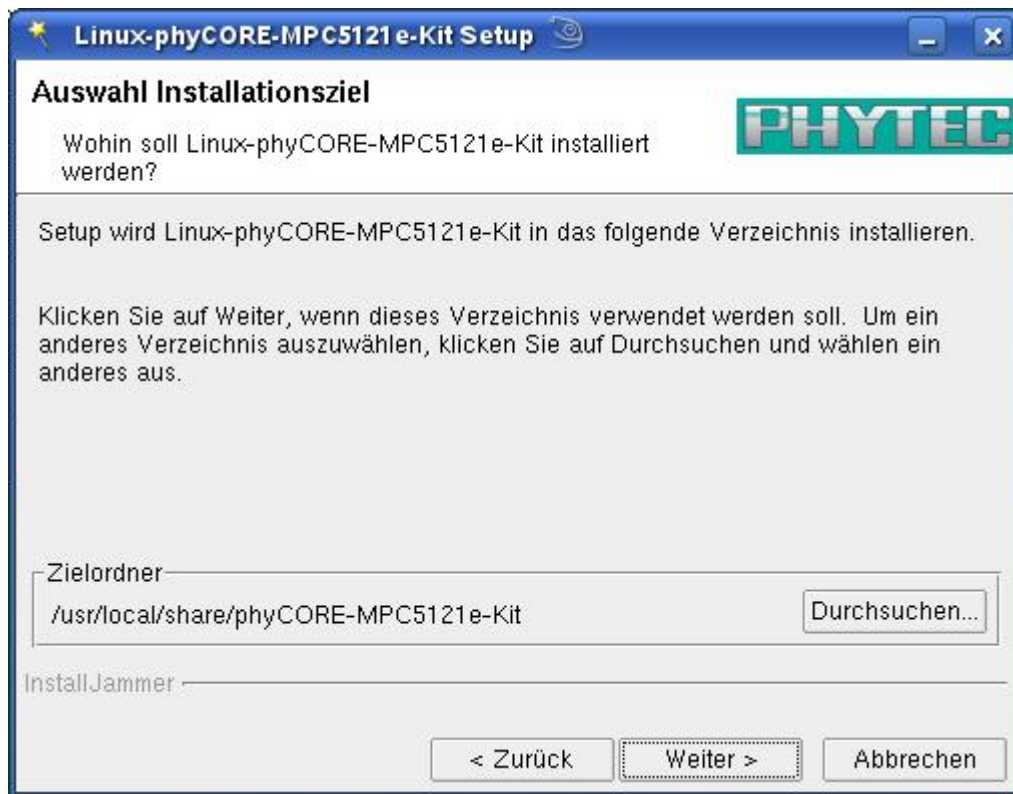
- Enter the root password.



- Click on *Yes* to proceed.
The welcome screen appears.



- Click on *Next* to continue.

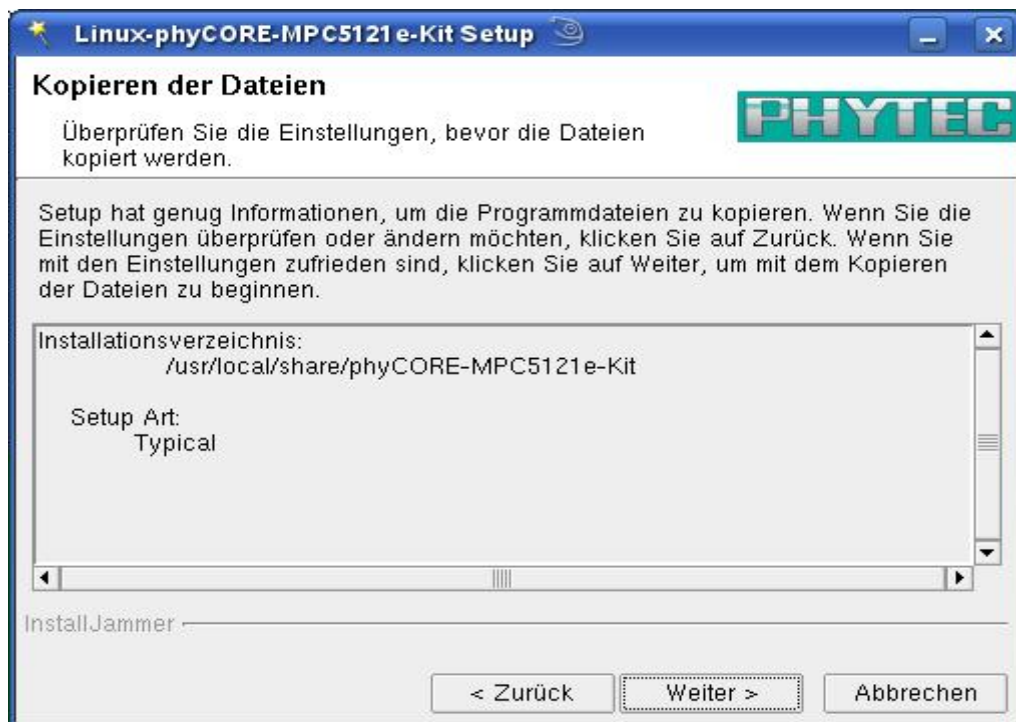


- Click on *Next* to continue.

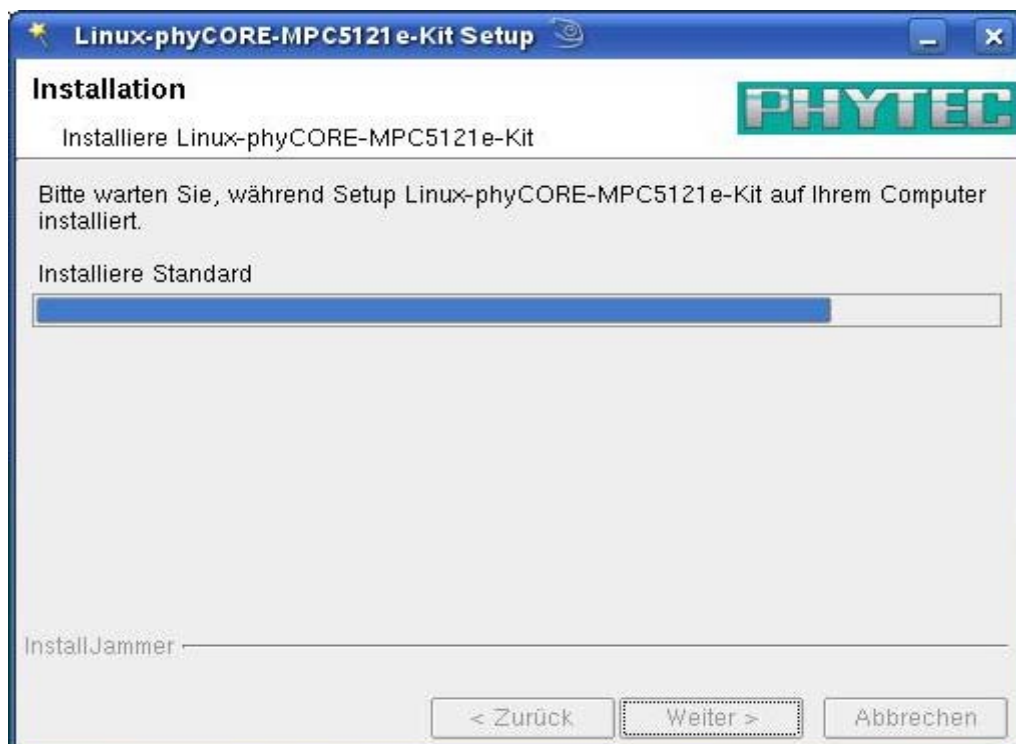


The default destination location is `/usr/local/share/phyCORE-MPC5121e-Kit`. All path and file statements within this QuickStart Instruction are based on the assumption that you accept the default install paths and drives. If you decide to individually choose different paths you must consider this for all further file and path statements when working with this QuickStart.

We strongly recommend accepting the default destination location.

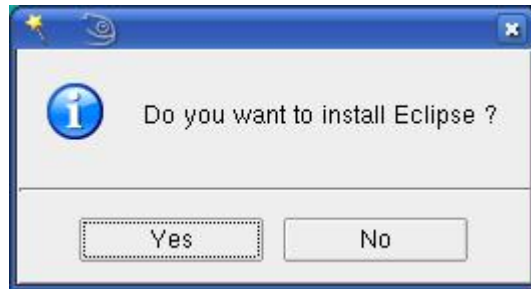


- Click on *Next* to install the setup files to */usr/local/share/phyCORE-MPC5121e-Kit*



The GNU GCC C/C++ Toolchain will be installed to the standard default directory `/opt/OSELAS.Toolchain-1.99.3/powerpc-603e-linux-gnu`. The program `mkimage` will be installed in `/usr/local/bin`. All other programs and examples will be installed in the selected destination directory.

After the files are copied, a dialog box for the Eclipse installation will appear.



- Click on *Yes* to install Eclipse. If you want to skip the installation of Eclipse choose *No*.



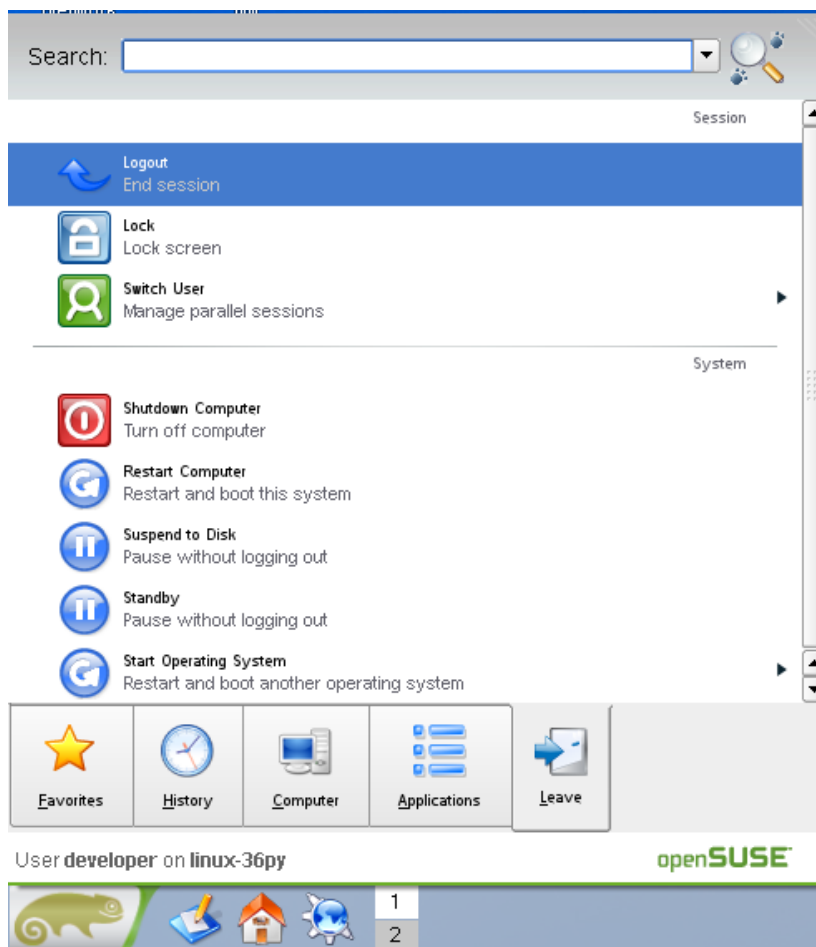
We recommend installing Eclipse even if you have already installed Eclipse on your system. This version of Eclipse includes additional plugins.

- Click *Next*.



- Click *Finish* to exit the setup.
- Close the terminal window.

Now you will have to restart the KDE desktop.



- Open the *K Menu* from the lower-left corner of the desktop.
- Select the *Leave* tab and choose *Logout*.
- When the display manager appears, enter your login name and password to restart the KDE desktop.



You have successfully installed the software for the Linux-PowerPC-Kit. You can now use the programs you need to develop your own applications for the target on your host system. The setup program did all necessary configurations. In the following passage you can find some advanced configuration information.

2.4 Advanced Configuration Information

In this part you can find some information on how to change the configuration steps of the setup program by your own. The setup program performed all the following configuration steps. The information in this part is for users who want to use the Linux-PowerPC-Kit on another Linux distribution than openSUSE. This is also interesting for users who want to see what configurations were done by the setup program.

During the setup program the GNU GCC C/C++ cross-compiler was installed in the directory `/opt/OSELAS.Toolchain-1.99.3/powerpc-603e-linux-gnu/gcc-4.3.2-glibc-2.8-binutils-2.18-kernel-2.6.27-sanitized/bin`. To start the cross-compiler directly from every part of the system, the directory of the cross compiler was added to the `$PATH` environment variable.

You can manually add the directory of the cross compiler to the `$PATH` by adding the following line in the file `/etc/profile`:

```
export PATH=/opt/OSELAS.Toolchain-1.99.3/powerpc-603e-linux-  
gnu/gcc-4.3.2-glibc-2.8-binutils-2.18-kernel-2.6.27-  
sanitized/bin:$PATH
```

You can open a terminal program and use the cross-compiler directly from the command line. For example you can compile a C program with the following command:

```
powerpc-603e-linux-gnu-gcc -o HelloWorld HelloWorld.c
```

In the standard configuration only the user root has write access to the serial interface. To use a serial communication tool like Microcom with normal user rights, you have to be a member of the group `uucp`.

A user can be added to this group with the following command:

```
groupmod -A <username> uucp
```

The serial communication program was configured during the setup 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 this program with these settings.

2.5 Connecting the host with the target

In this section you will learn how to connect your host PC with 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 flash on the target, and you will be able to login with the serial communication program Microcom as well as via a Telnet session using a peer-to-peer network connection.

- Connect the serial cable with the lower connector P2 on the target and the serial interface COM1 on your host.



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

- Connect the cross-over Ethernet cable with the connector Ethernet on the target and the appropriate network card of your host.



- Click the *Microcom* icon on your desktop.
- Connect the AC adapter with the power supply connector X19 (12V) on your board.



The power connector should have 12 VDC inside and outside should be ground.

After connecting the board with the power supply the target starts booting. When the target finished loading the file system you will see a screen similar to the following screenshot:

```

lighttpd: starting
lighttpd: done
/usr/sbin/pure-ftpd
pure-ftpd: starting pure-ftpd: /usr/sbin/pure-ftpd
/usr/sbin/pure-uploadsript
pure-ftpd: starting upload helper daemon...
done
loading modules

OSELAS(R)-phyCORE-12-2 (PTXdist-1.99.12/2009-11-24T07:42:44+0100)

PHYCORE

*** PHYTEC BSP PD09.1.0R1 based on OSELAS(R) ***

starting pid 690, tty '/dev/console': '/sbin
phyCORE login: █

```

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



When the target is connected with the Power supply, first the bootloader U-Boot is loaded from the flash memory. Then the bootloader is uncompressing and booting the Linux kernel from the flash. The kernel will then mount 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 mount time by JFFS2 through a scan of the MTD's log content.

In addition to its 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.



Troubleshooting:

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

At the end of the setup, you had to restart the KDE desktop. If you haven't done yet, restart the KDE desktop and try again.

It is also possible that your user account is missing read and write access to the serial interface:

- Open the *YaST Control Center*.
- Choose *Security and Users*.
- Choose *User Management*

In the line of your user name should be the group *uucp*.

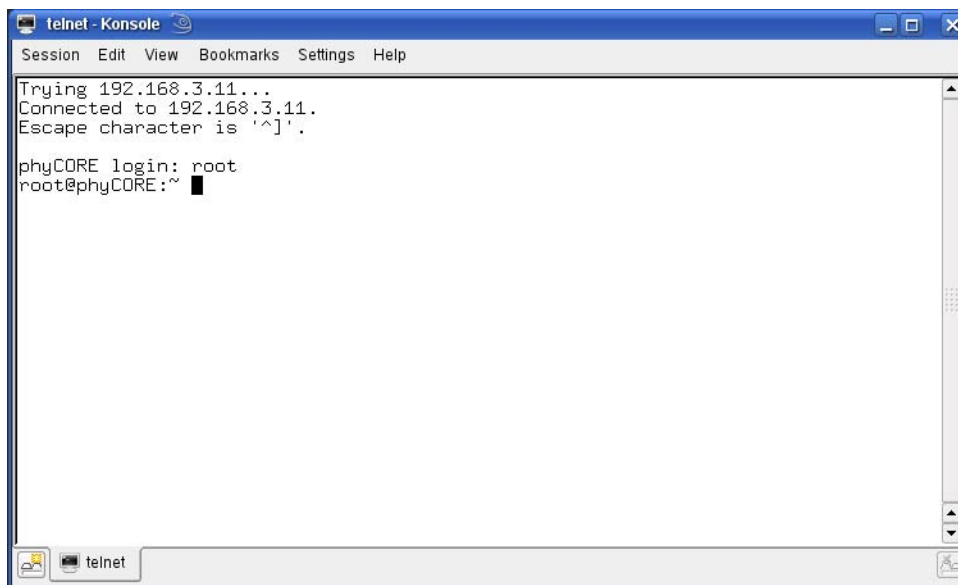
- If the group is missing select your user name and click the *Edit* button.
- Select the tab *Details*.
- Check in *Groups* the group *uucp*.
- Click *Accept*.
- Click *Finish* and close YaST.
- You need to log out and log in again for the new group membership to take effect.

Now you can test the network connection to the target.



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

A new window with a connection to the target opens.



If you see the user login in the opened window, the network configurations were configured correctly.

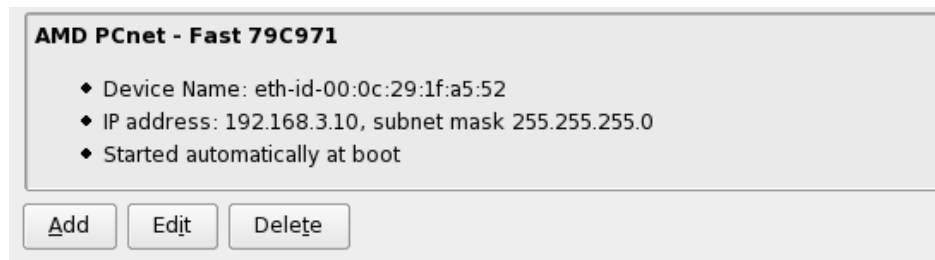
- Close the window.



Troubleshooting:

If you don't see the user login, check the 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 with the network card you have configured with the IP address 192.168.3.10.

If you don't see the login, you may not have set up the right IP address of your host. You can check the settings of your network card by opening YaST. In the YaSt Control Center you can select *Network Settings* in *Network Devices*. There should be the following configuration:



Information how to configure your network device can be found in the section *Configuring the Host Plattform*.

You have successfully setup all configurations to access your target from your host.

2.6 Copying an Example to the Target

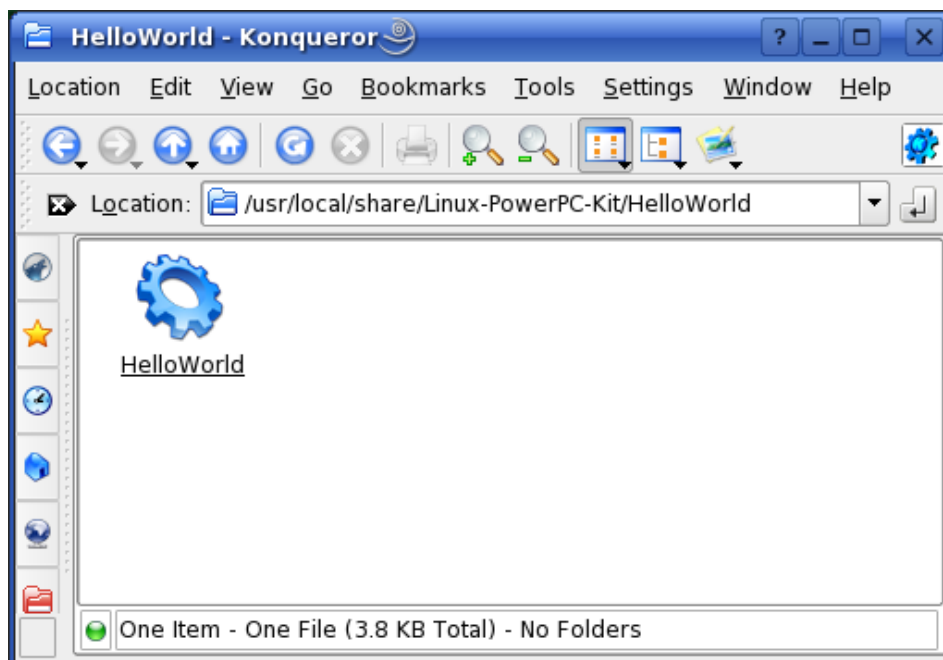
In this section you will learn how to copy an example program to the target using the FTP protocol with the *Konqueror* browser. After that you will execute the example on the target. At the end of this passage you can find some information how to copy and execute a file on the target using the command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

2.6.1 Copying a Program to the Target

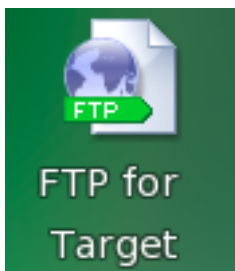


- First click on the icon *phyCORE-MPC5121e-Kit* on your KDE desktop.

A new window with the content of the installation directory opens.

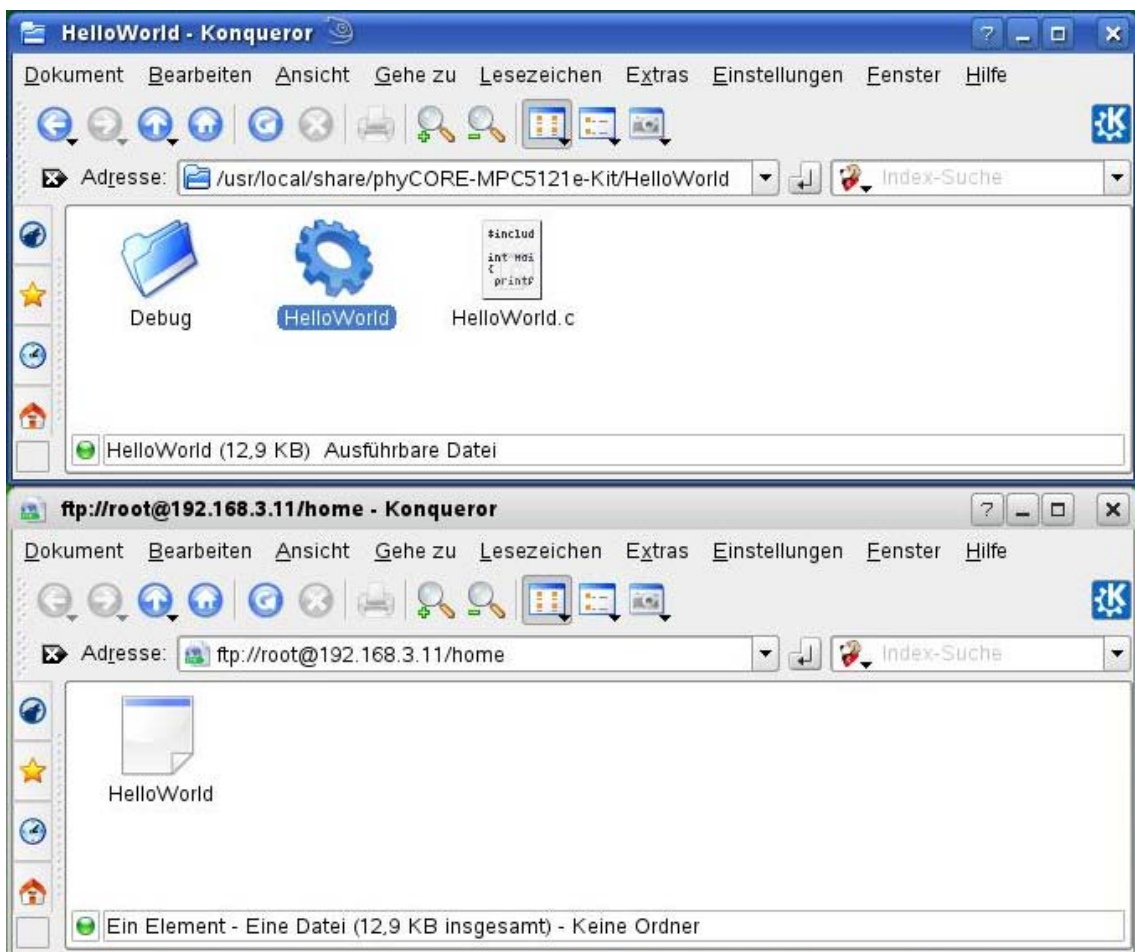


- Enter the directory *HelloWorld*.



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

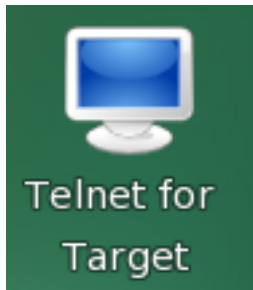
A window with an FTP session to the target opens. User is *root* and password is empty. Now you have two windows opened, one for the target and one for the host. You can use these two windows to copy files per “drag and drop” from the host to the target (and vice versa).



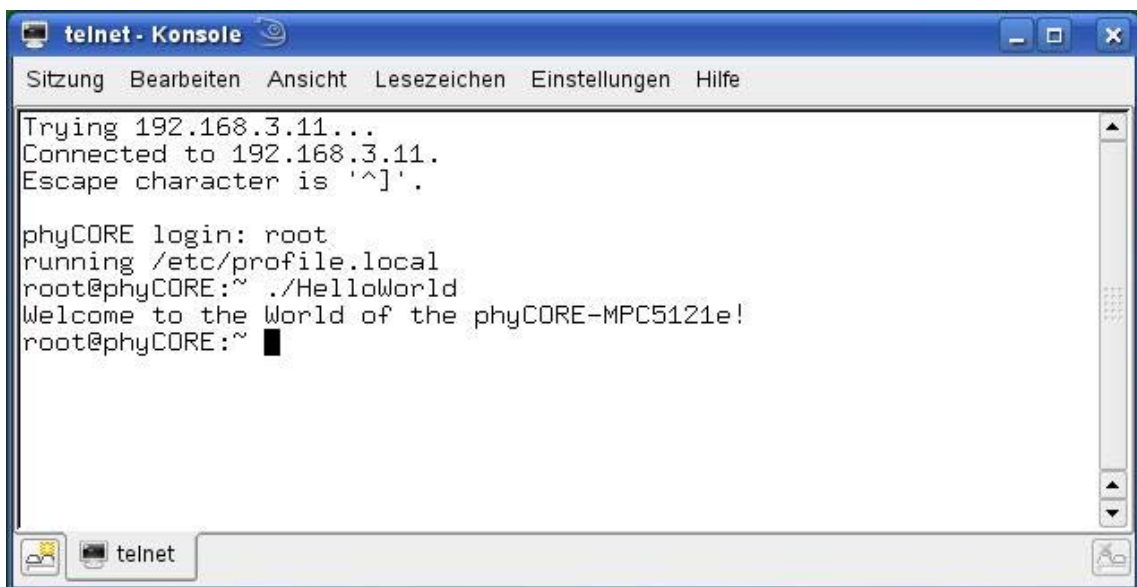
- Select the window that lists *HelloWorld* program on your hard disk.

- Click the *HelloWorld* program and hold the left mouse button pressed.
- Drag the program into the window with the FTP session to the target and release the mouse button.
- Choose *Copy here* in the appearing context menu.
- Close the two windows.

2.6.2 Using Telnet to execute a Program on the Target



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



```
telnet - Konsole
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
Trying 192.168.3.11...
Connected to 192.168.3.11.
Escape character is '^]'.

phyCORE login: root
running /etc/profile.local
root@phyCORE:~ ./HelloWorld
Welcome to the World of the phyCORE-MPC5121e!
root@phyCORE:~
```

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

The program starts and you see the following output:

Welcome to the World of the phyCORE-MPC5121e!

2.6.3 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 into the target via SSH from the command line for one time. This is necessary to add the RSA public key of the target to the list of the known hosts.



TIP

There are several authentication methods when using SSH. The method used on the phyCORE-MPC5121e-tiny is the *hosts.equiv* method combined with RSA-based host authentication.

If the machine the user logs in from is listed in */etc/hosts.equiv* on the remote machine, and the user name is the same on both sides, the user is allowed for log in.

On the target, the file */etc/hosts.equiv* has the following entry:

```
# file: /etc/hosts.equiv
#
# Allow access from everywhere.
#
+ +
```

The “+ +” means that every user can login from every host.

When the host connects to the target, the file *~/.ssh/known_hosts* (on the host) is consulted when using *hosts.equiv* with 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.

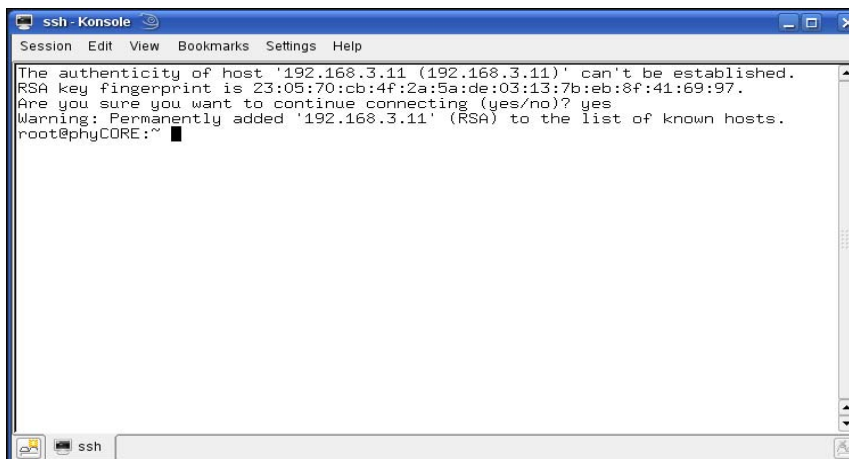


This authentication method closes security holes due to IP spoofing, DNS spoofing and routing spoofing. But note that */etc/hosts.equiv* is, in general, inherently insecure and should be disabled if security is a concern.



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

A new window opens.



In this window you can see that the authenticity of the target 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 login prompt.



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

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

- Type **./HelloWorld** start the program you had copied to the phyCORE-MPC5121e-tiny before.

The program starts and you should see the following output:

Welcome to the World of the phyCORE-MPC5121e!

- Close the SSH window.

If you click on the Icon *SSH for Target* again, the host will connect directly to the target.

- Close the window.



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

2.7 Advanced Information

2.7.1 Copying a program to the Target with the Command Line

- Open a new terminal window.
- Change to `/usr/local/share/phyCORE-MPC5121e-Kit/HelloWorld`:
cd /usr/local/share/phyCORE-MPC5121e-Kit/HelloWorld
- Copy the application to the target by typing:
ftp -u ftp://root:root@192.168.3.11/ HelloWorld
Be sure to enter a slash followed by a space after the IP address.

2.7.2 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**.

2.7.3 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 logout automatically.

3 Getting More Involved

**70 min**

TIME

In this chapter you will pass some continuative topics. First you will configure and compile your own kernel. With the kernel configuration tool you can add additional features or disable them if they are not needed. After compiling the kernel, you will learn how to write the newly created kernel into target's flash memory and how to start the new kernel.

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 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-MPC5121e-tiny boots.

3.1 Configuring and Compiling the Kernel

In this part you will learn how to configure and build a new Linux kernel and a root filesystem. First you will copy the kernel archive to your home directory and extract the kernel source. Then you will configure the kernel and the root filesystem 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.

The kernel used by PHYTEC is based on a standard kernel available from www.kernel.org. Additionally, the kernel archive in your setup installation directory already includes all necessary patches for the phyCORE-MPC5121e-tiny.

In the end, you need the following files:

- ptxdist-1.99.12.tgz (Tool from our partner Pengutronix, that configures and compiles BSPs)
- ptxdist-1.99.12-patches.tgz (Patches for the tool)
- OSELAS.BSP-Phytec-phyCORE-MPC512x-tiny-PD09.1.0R1.tar.gz (BSP for phyCORE-MPC5121e-tiny)
- powerpc-603e-linux-gnu.tar.bz2 (ready-to-use toolchain)

All files are downloadable from our ftp-server. Please create a temporary directory within your home directory, for example *local/*, and copy the files into it.

Now, open a new terminal if not already open and install PTXdist:



- Click the terminal icon on your desktop.
- Type the following commands to extract the PTXdist-archieives:

```
cd local  
tar zxvf ptxdist-1.99.12.tgz  
tar zxvf ptxdist-1.99.12-patches.tgz
```
- Now you should check if all necessary tools are installed on your host. Therefor call the configure script:

```
cd ptxdist-1.99.12  
./configure
```

After a long list of checking ... it should return with:

```
configure: creating ./config.status
config.status: creating Makefile
config.status: creating scripts/ptxdist_version.sh
config.status: creating rules/ptxdist-version.in
ptxdist version 1.99.12 configured.
Using '/usr/local' for installation prefix.
Report bugs to ptxdist@pengutronix.de
```

If any tool that is necessary for PTXdist is missing, script will abort with an error. In this case you need to install the missing tool from your linux distribution. After that, please restart the configure script.

In any case, tool *quilt* needs to be installed additionally, in order to avoid errors while compiling patches!

- After successful completion of configure script, please type:
make
- Now you can install PTXdist. Because target directory is */usr/local*, this must be done as user *root*:
sudo make install
[enter root password]
[...]
- Finally you can delete the temporary directory:
cd
rm -rf local

When using PTXdist for the first time, setting some configurations is needed. The most important are:

- Where to put the source packages to.
- Proxy settings for accesses to the internet.

Please start PTXdist setup:

ptxdist setup

PTXdist will download all necessary sources from the internet. For every source package, that cannot be found on your host, PTXdist will do this by executing command *wget*. If a proxy server should be used for doing internet accesses, please select *Proxies*, then *FTP Proxy* and then enter address and port of the proxy in the following form:

<protocol>://<address>:<port>

PTXdist will store downloaded source packages locally. Thus, if you work with more than one project, every project would download its own sources, even if they are the same that have already been downloaded. In order to avoid this, we suggest to use one single directory for all source packages of all projects. Some source packages are already included within this BSP, so we suggest to use its *src*-directory as source directory for all projects. Please select *Source Directories* and enter:

`${PTXDIST_WORKSPACE}/src`

In order to be able to build an BSP for the MPC5121e, you need a toolchain with a suitable crosscompiler for this processor. So please decompress the corresponding archive with root rights. It will be stored in directory */opt* automatically.

`tar xPvjf powerpc-603e-linux-gnu.tar.bz2`

Now you're ready to build the BSP. Please decompress archive `OSELAS.BSP-Phytec-phyCARD-i.MX27-PD09.1.0R1.tar.gz` into a directory and then move into it. Select the platform for which the BSP should be build:

**ptxdist platform \
configs/phyCARD-i.MX27-1.99.12-trunk/platformconfig**

You should get the messages:

```
info: selected platformconfig:  
      'configs/phyCARD-i.MX27-1.99.12-trunk/platformconfig'
```

Furthermore PTXdist should have recognized the necessary toolchain and confirm this by giving you the messages:

```
found and using toolchain:  
'/opt/OSELAS.Toolchain-1.99.3/powerpc-603e-linux-gnu/gcc-4.3.2-glibc-2.8-binutils-  
2.18-kernel-2.6.27-sanitized/bin'
```

Otherwise you need to choose the toolchain manually:

**ptxdist toolchain /opt/OSELAS.Toolchain-1.99.3/powerpc-603e-linux-
gnu/gcc-4.3.2-glibc-2.8-binutils-2.18-kernel-2.6.27-sanitized/bin**

For building the BSP you could call ptxdist go now, but then you would get so much stuff for the root-filesystem of your target, that it won't fit into the flash-memory any more. This rootfs could only be used when mounting it via NFS. Thus, PHYTEC provides the BSP together with a file called a 'collection', that excludes not necessary stuff for you. Please tell ptxdist to use this collection by calling

ptxdist collection configs/collectionconfig-kit

Now, for building the BSP call

ptxdist go

Now it will take some hours. Sometimes it can happen that an attempt to download a source package failes due to dead links. PHYTEC cannot guarantee the accessibility of all the webpages that are needed to build a BSP. You can check our ftp-server via

ftp://ftp.phytec.de/pub/BSP_PACKAGES/EXTERNALS

If you find a missing source package there. Otherwise you need to search the web for missing packages. If you have found them, you need to download them into your src-directory manually. After that, please restart build process by calling **ptxdist go** again. If you cannot find a needed source package, please tell us via support@phytec.de.

Sometimes it even might happen, that ptxdist can download a file that is claimed to be an archive, but then cannot decompress it. Reason is, that the link didn't deliver the requested archive, but an HTML containing a message like: "Error - This file has been moved to ...". Please treat this case in the same way as if nothing had been delivered by the link.

Finally you can create the rootfs by calling

ptxdist images

You will find the kernel-image linuximage and the flashfilesystem-image root.jffs2 in directory platform-phyCORE-MPC5121e/images. The NFS-mountable rootfs will be located in platform-phyCORE-MPC5121e/root.

- Close the terminal window.

In this section you learned how to configure and compile a new kernel. Now you can add new features to your kernel, or remove features you do not need.

3.2 Writing the Kernel into the Target's Flash

In this passage you will find a description how to write the newly created images into the phyCORE'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 five 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, the fourth contains the root file system, and the fifth contains the oftree.

The four partitions have the following address ranges:

```
0xffff00000 - 0xffff5ffff (U-Boot)
0xffff60000 - 0xffff9ffff (U-Boot Environment)
0xfe060000 - 0xfe23ffff (Linux Kernel)
0xfe240000 - 0xffefffff (Linux Root File System)
0xffffa0000 - 0xffffffffff (Linux Ofspace)
```

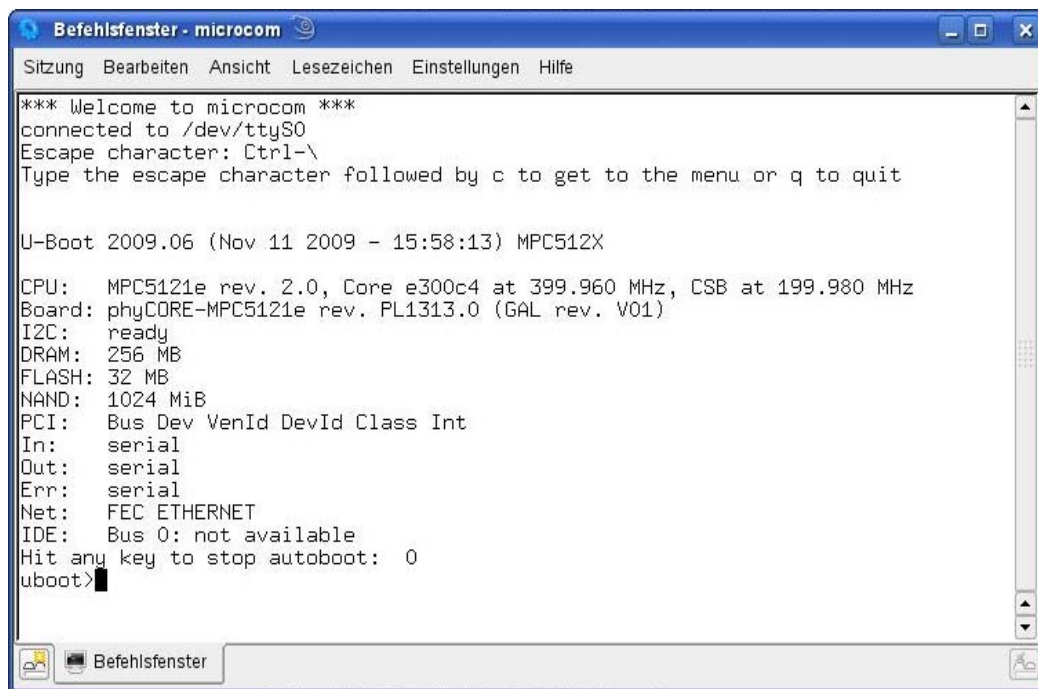


You should never erase the U-Boot partition. If this partition is erased, you won't be able to start your target anymore. Refer to the chapter “*Installing Linux on the phyCORE-MPC5121e-tiny*” for detailed information on how to restore your U-Boot partition in such a case.



- First open a new terminal window if it is not opened yet.
- Copy the new images to the */tftpboot* directory and exit:

```
cd platform-MPC5121e/images
cp linuximage /tftpboot
cp root.jffs2 /tftpboot
cp oftree /tftpboot
exit
```

```
Befehlsfenster - microcom
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
*** Welcome to microcom ***
connected to /dev/ttyS0
Escape character: Ctrl-\
Type the escape character followed by c to get to the menu or q to quit

U-Boot 2009.06 (Nov 11 2009 - 15:58:13) MPC512X

CPU: MPC5121e rev. 2.0, Core e300c4 at 399.960 MHz, CSB at 199.980 MHz
Board: phyCORE-MPC5121e rev. PL1313.0 (GAL rev. V01)
I2C: ready
DRAM: 256 MB
FLASH: 32 MB
NAND: 1024 MiB
PCI: Bus Dev VenId DevId Class Int
In: serial
Out: serial
Err: serial
Net: FEC ETHERNET
IDE: Bus 0: not available
Hit any key to stop autoboot: 0
uboot>
```

- 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.

You can download an image from the TFTP – server, erasing the required Flash area and writing the image from the RAM into the Flash with one simple command.

- Before you can execute this command you have to set the names of your images to the corresponding environment variables:

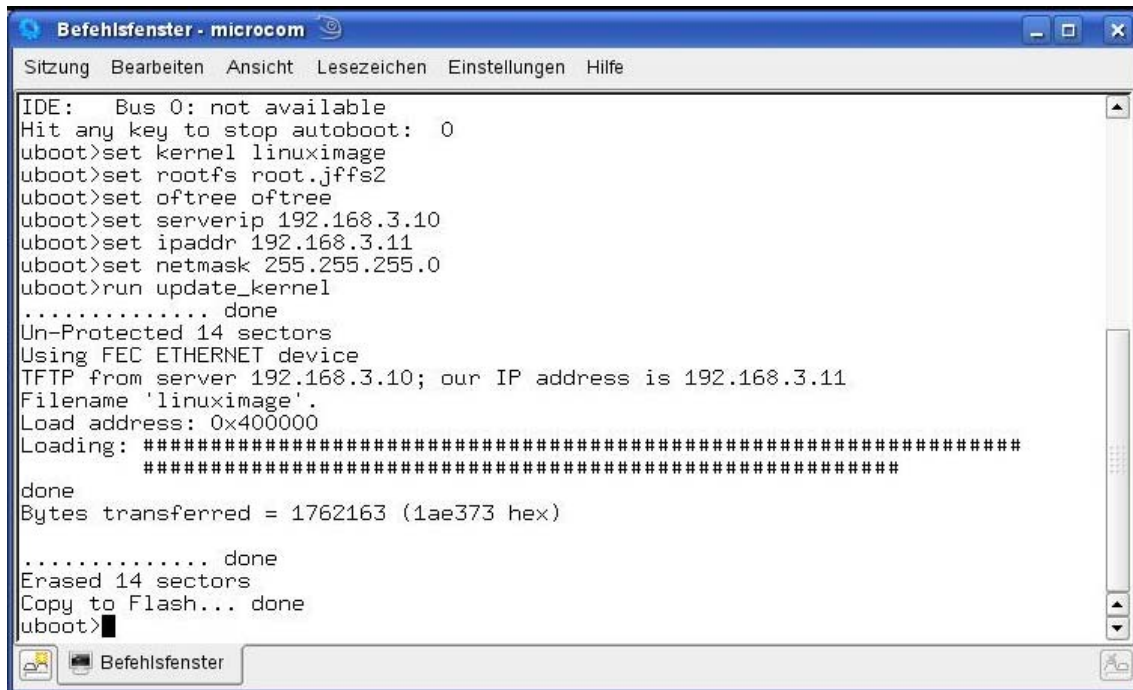
```
set kernel linuximage
set rootfs root.jffs2
set oftree oftree
```

- Make sure that you’ve got the correct IP-setting:

```
set serverip 192.168.3.10
set ipaddr 192.168.3.11
set netmask 255.255.255.0
```

- Now you can start downloading and writing the three images into the Flash using the commands:

```
run update_kernel  
run update_rootfs  
run update_oftree
```



```
Befehlsfenster - microcom  
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe  
IDE: Bus 0: not available  
Hit any key to stop autoboot: 0  
u-boot>set kernel linuximage  
u-boot>set rootfs root.jffs2  
u-boot>set oftree oftree  
u-boot>set serverip 192.168.3.10  
u-boot>set ipaddr 192.168.3.11  
u-boot>set netmask 255.255.255.0  
u-boot>run update_kernel  
..... done  
Un-Protected 14 sectors  
Using FEC ETHERNET device  
TFTP from server 192.168.3.10; our IP address is 192.168.3.11  
Filename 'linuximage'.  
Load address: 0x400000  
Loading: #####  
#####  
done  
Bytes transferred = 1762163 (1ae373 hex)  
..... done  
Erased 14 sectors  
Copy to Flash... done  
u-boot>
```

The copy processes can take some minute, depending on the speed of your system.

- Press the RESET button on the target to restart with the new images.

The target will restart booting with the newly created images.

- Close Microcom when the target successfully finished with booting the kernel and mounting the root filesystem.



Troubleshooting:

If any problem occurs after writing the images into flash, you can restore the original images system from your setup CD-ROM.

You will find the kernel and root file system in the directory *PHYTEC/PCM046 phyCORE-MPC5121e/Linux-Kit/BSP/Images*.

- Copy the files *uImage-pcm046*, *root-pcm046.jffs2* and *oftree-pcm046* into the directory */tftpboot*
- Type the following commands in the u-boot command line:
set kernel uImage-pcm046
set rootfs root-pcm046.jffs2
set oftree oftree-pcm046
run update_kernel
run update_rootfs
run update_oftree



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

Opening an Existing Project

In this section 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.

Copying the HelloWorld project:

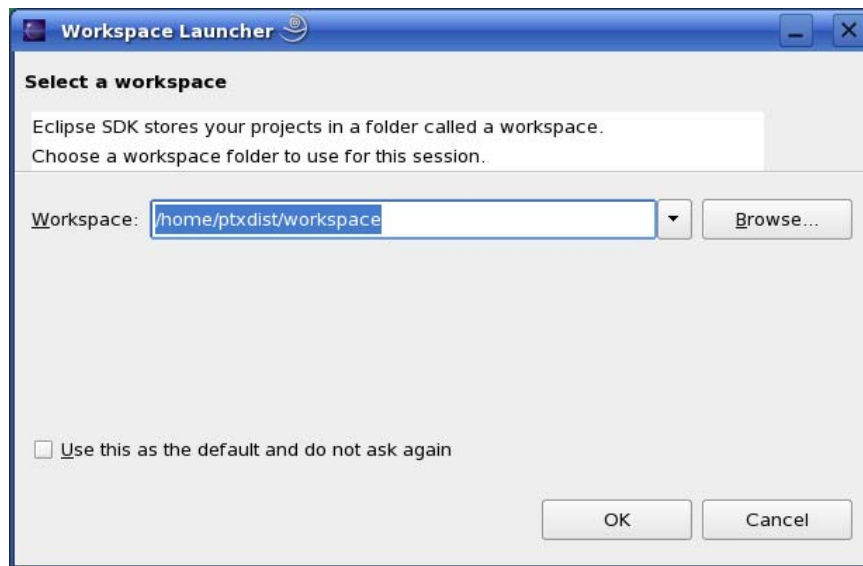


- Click on the *phyCORE-MPC5121e-Kit* directory icon.
- Right-click the *HelloWorld* directory and select *Copy*.
- Browse to your home directory.
- If the *workspace* directory doesn't exist, create a directory *workspace* in your home directory.
- Enter the *workspace* directory.
- Right-click in the *workspace* directory and select *Paste*.
- Close the *Konqueror* file browser.

Starting Eclipse and importing the example project:

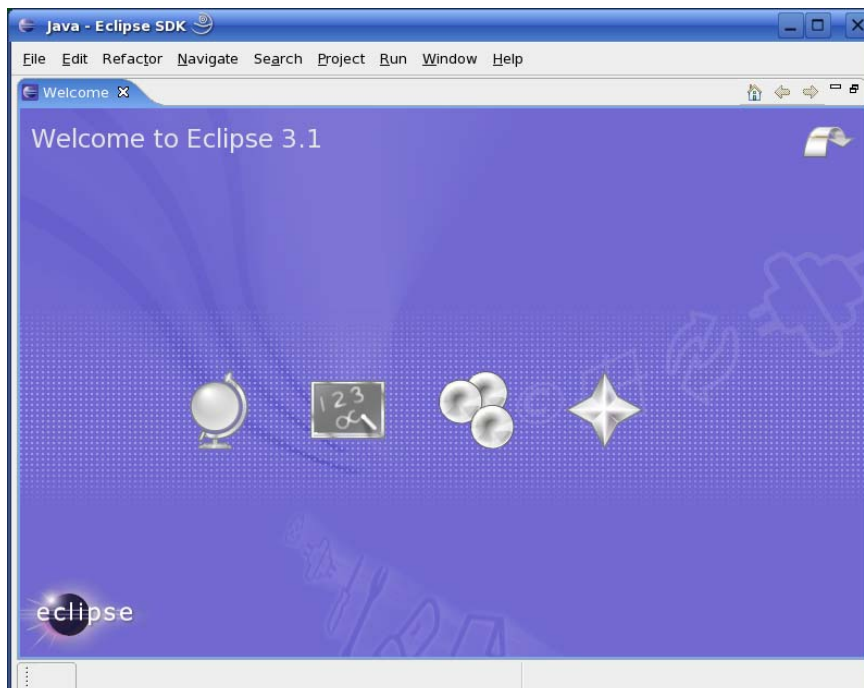


- Click on the *Eclipse* icon to start the application. You can find the icon on your desktop.

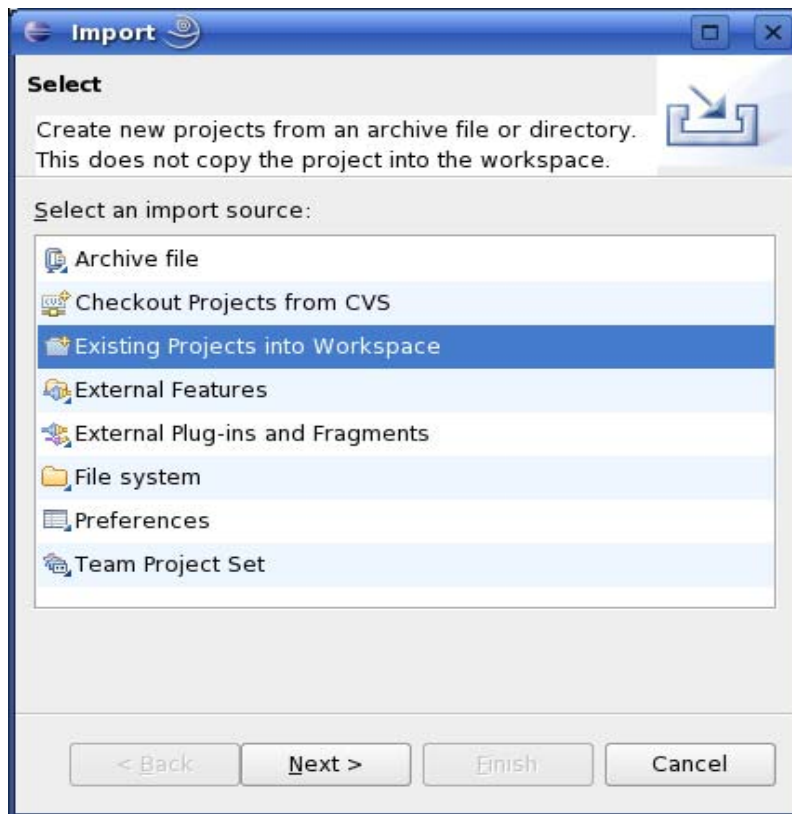


- Confirm the workspace directory with *OK*.

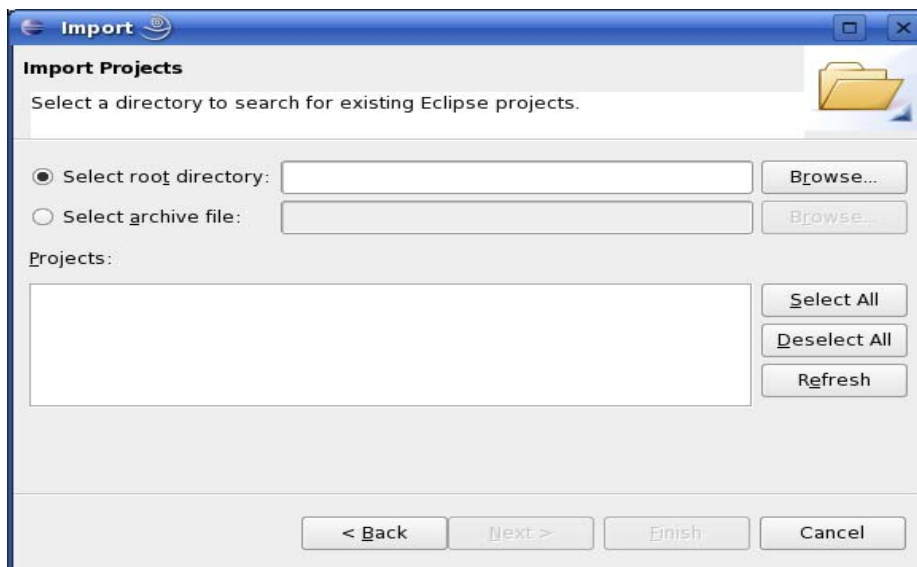
The *Welcome* screen will appear.



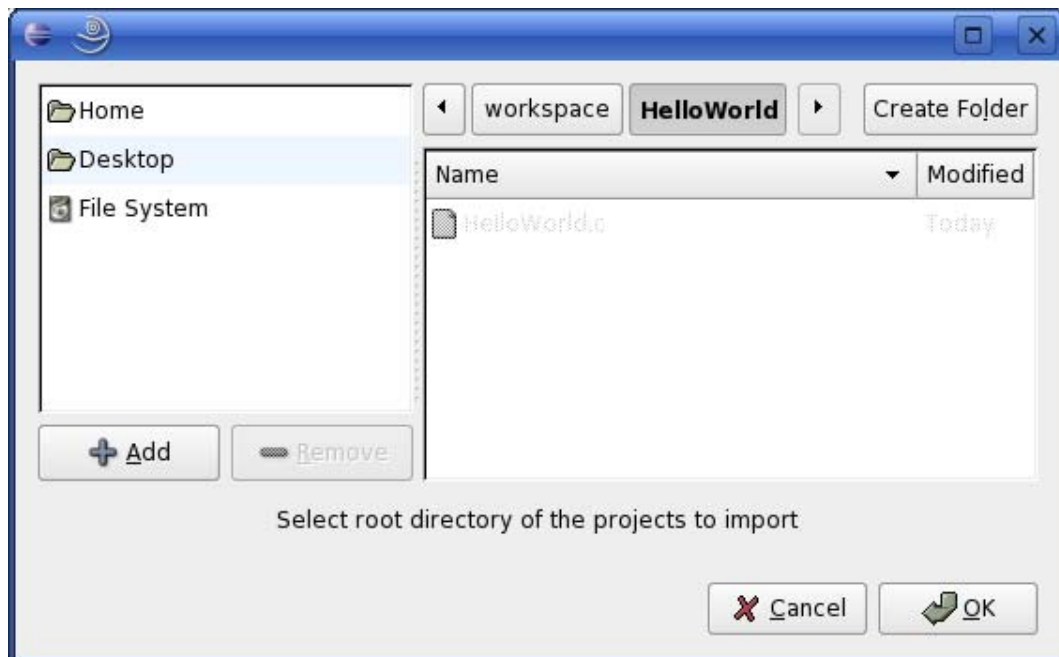
- Select *File -> Import* in the menu bar.



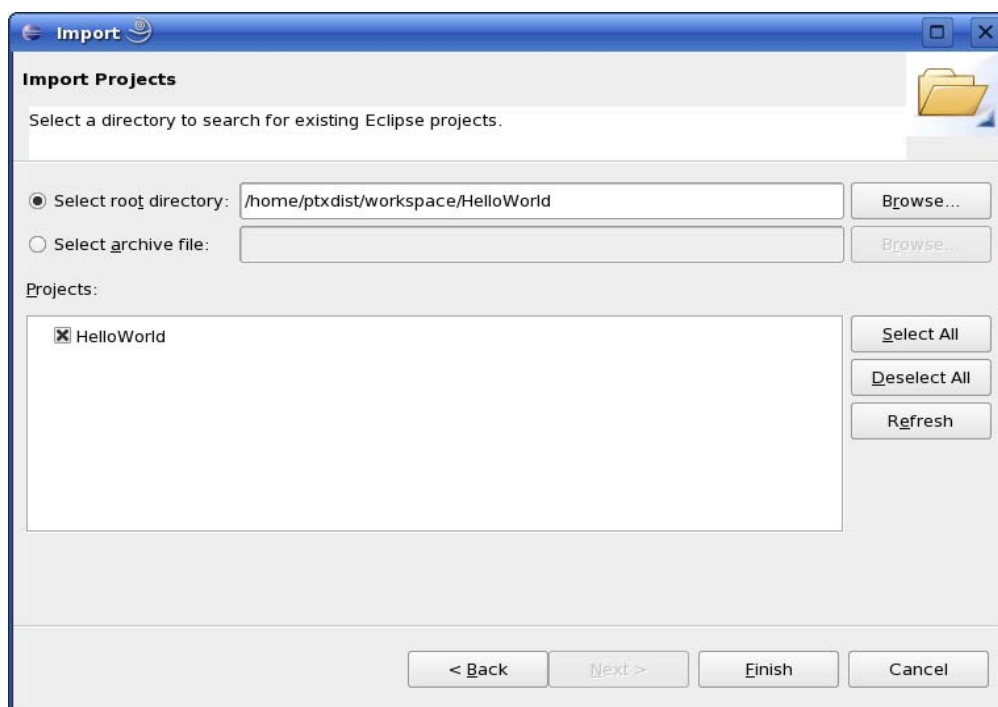
- Select *Existing Projects into Workspace*.
- Click *Next*.



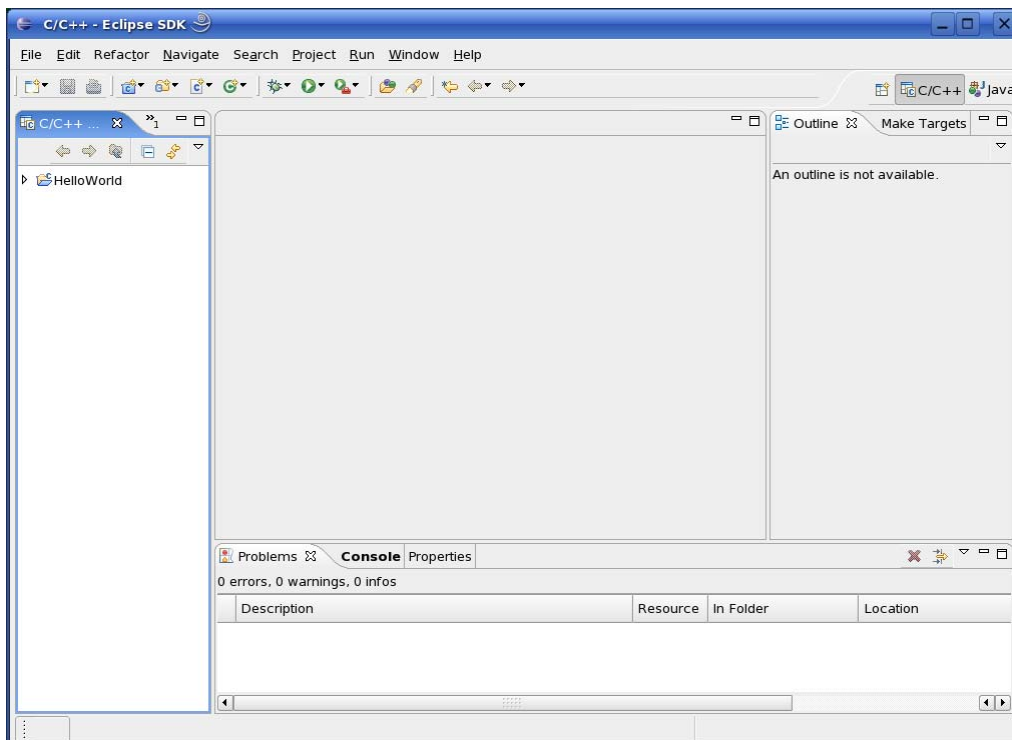
- Select *Browse*.



- *Double-Click* on the *HelloWorld* directory.
- Click *OK*.



- Select *Finish* to import the project.

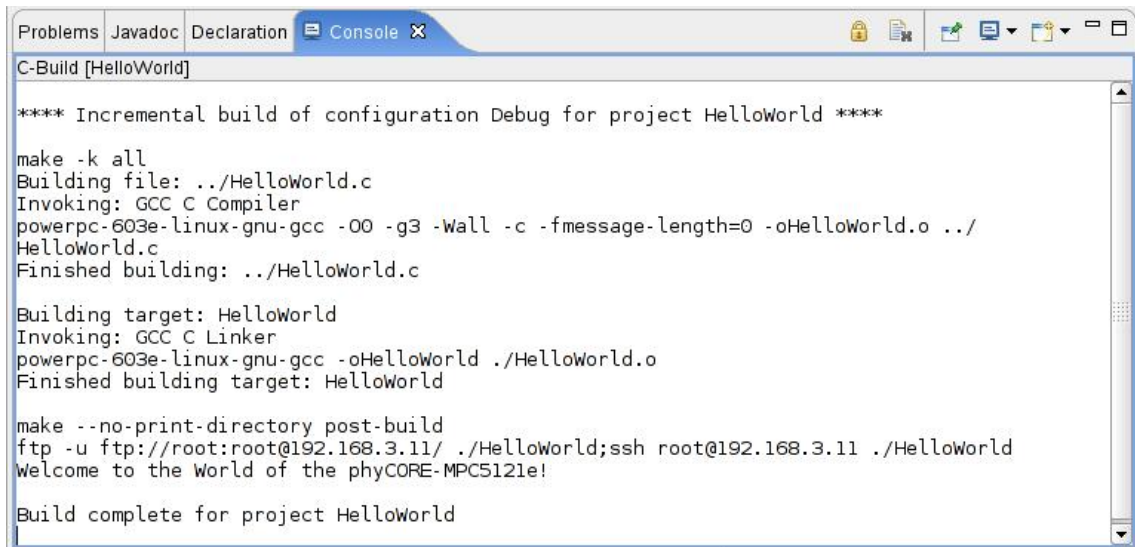


- Close the *Welcome Screen*.

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 the FTP. After the file has been copied to target, the program is executed on the target using SSH. You should now see the “*Welcome to the World of the phyCORE-MPC5121e!*” output in the console window.

- Select the *Console* tab.

You will see the following content in the console window:



```
C-Build [HelloWorld]

**** Incremental build of configuration Debug for project HelloWorld ****

make -k all
Building file: ../HelloWorld.c
Invoking: GCC C Compiler
powerpc-603e-linux-gnu-gcc -O0 -g3 -Wall -c -fmessage-length=0 -oHelloWorld.o ../
HelloWorld.c
Finished building: ../HelloWorld.c

Building target: HelloWorld
Invoking: GCC C Linker
powerpc-603e-linux-gnu-gcc -oHelloWorld ./HelloWorld.o
Finished building target: HelloWorld

make --no-print-directory post-build
ftp -u ftp://root:root@192.168.3.11/ ./HelloWorld;ssh root@192.168.3.11 ./HelloWorld
Welcome to the World of the phyCORE-MPC5121e!

Build complete for project HelloWorld
```



If the project is not built automatically, you will have to check *Project ► Build automatically* from the menu bar.



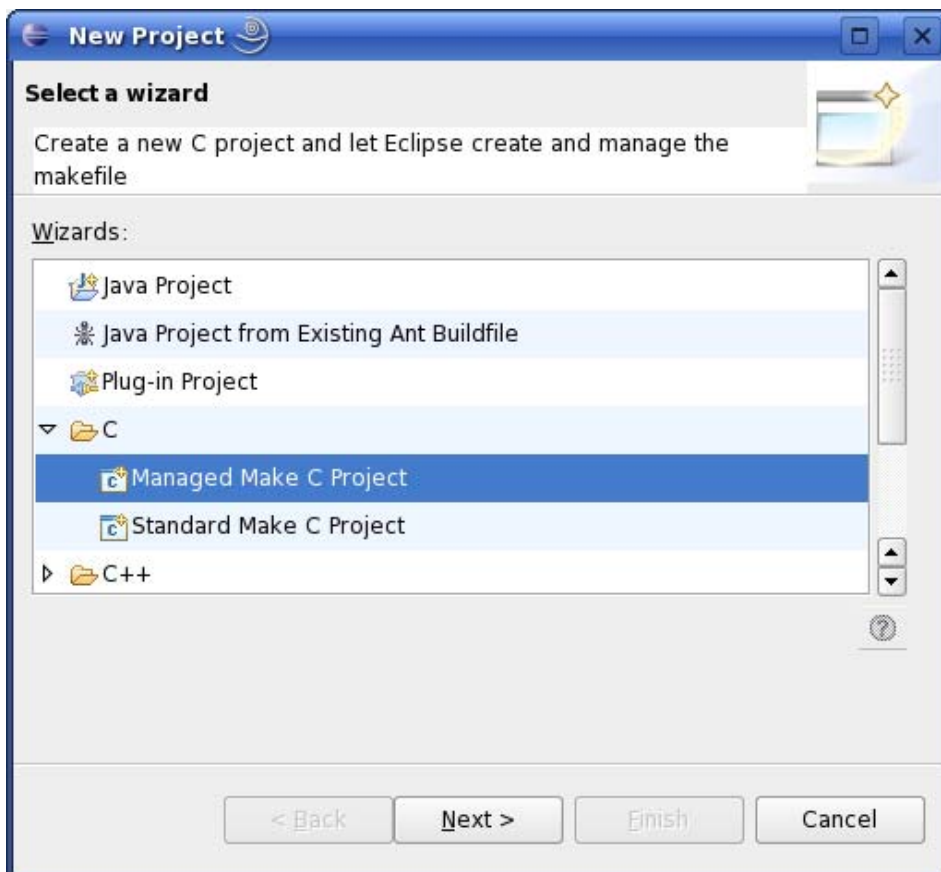
You have successfully passed the first steps with the Eclipse IDE. You are now able to import existing projects into the Eclipse Workspace. You can compile an existing project and execute the program on the target.

3.3 Creating a New Project

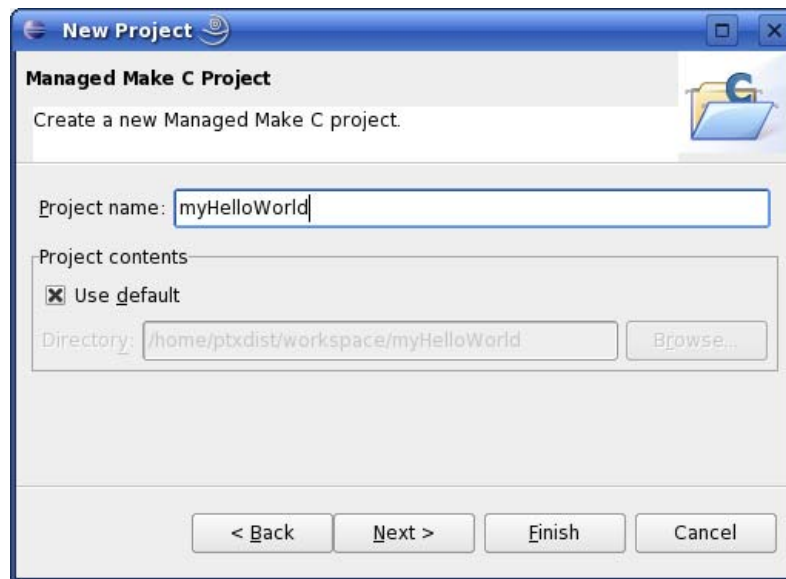
In this section you will learn how to create a new project with Eclipse and how to configure the project for use with the GNU C/C++ cross development toolchain.

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

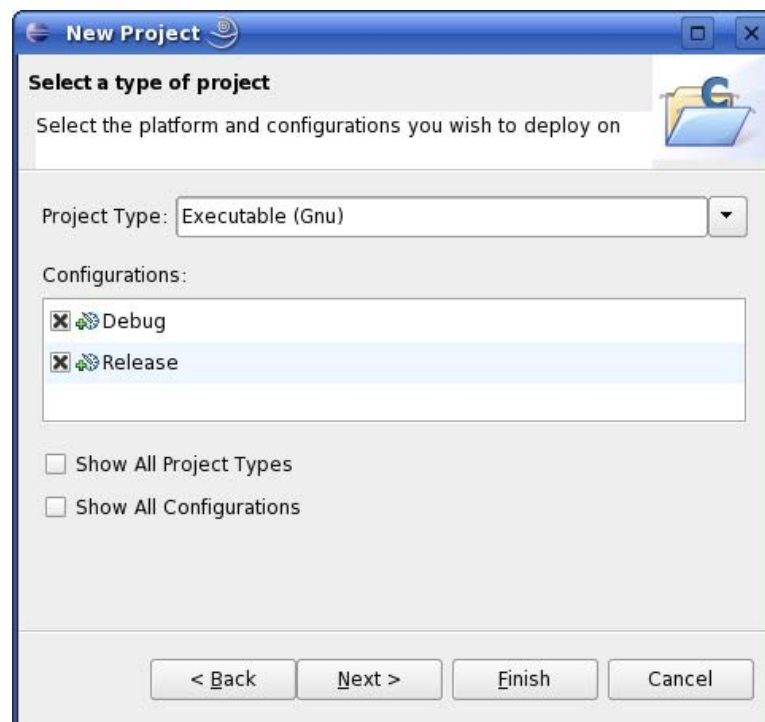
A new dialog opens.



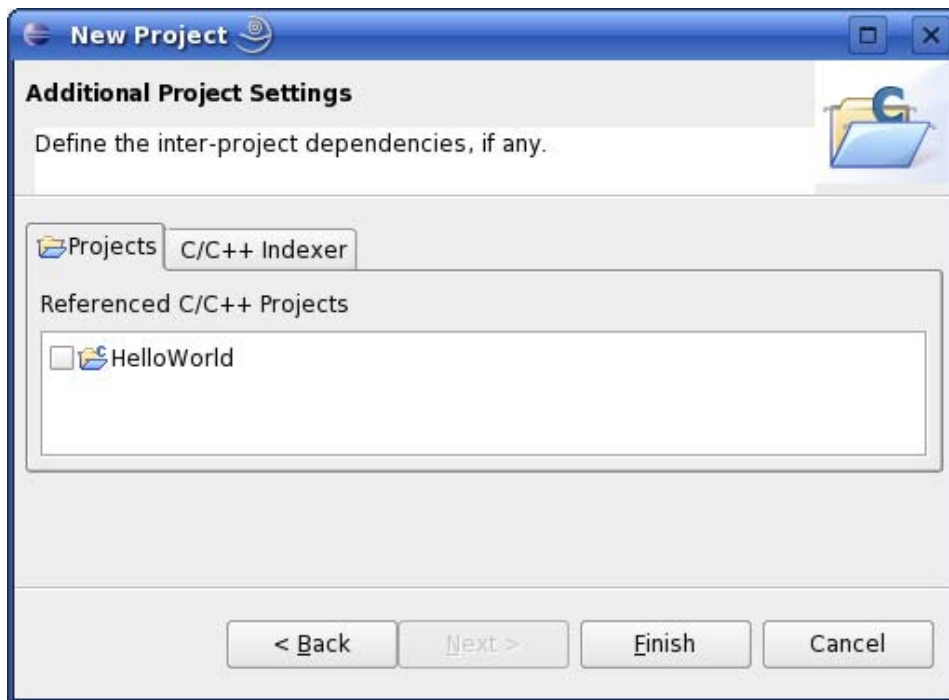
- Select *Managed Make C Project* and click *Next*.



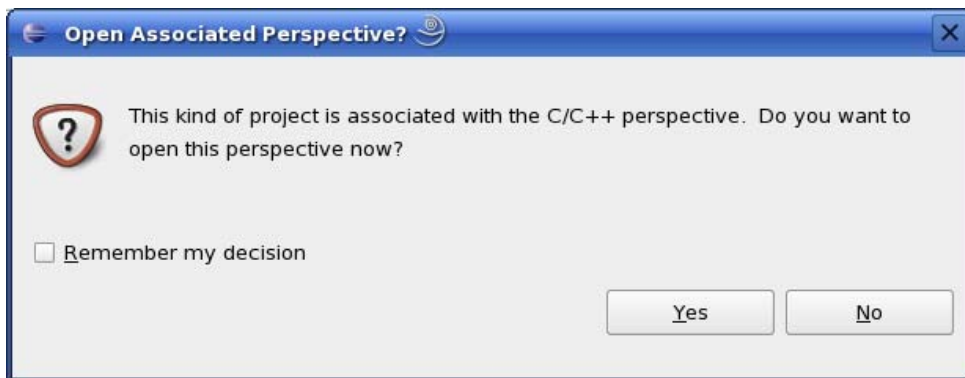
- Enter the project name *myHelloWorld* and click *Next*.



- Click *Next*.

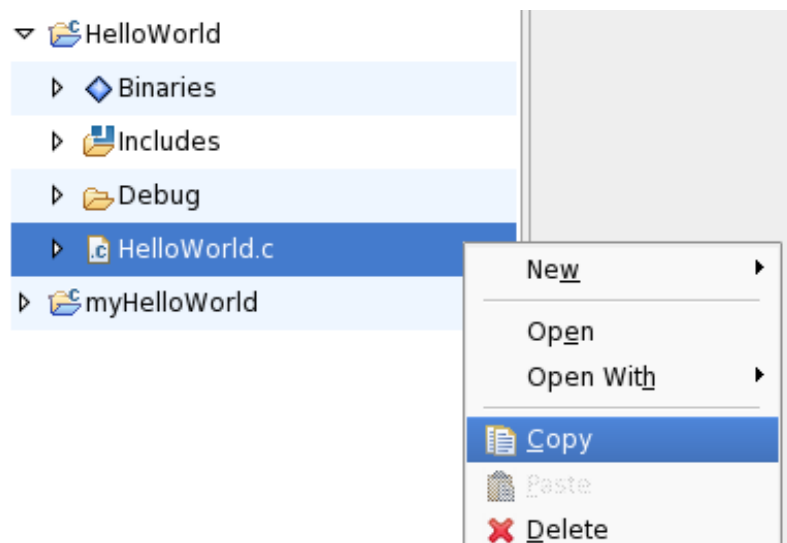
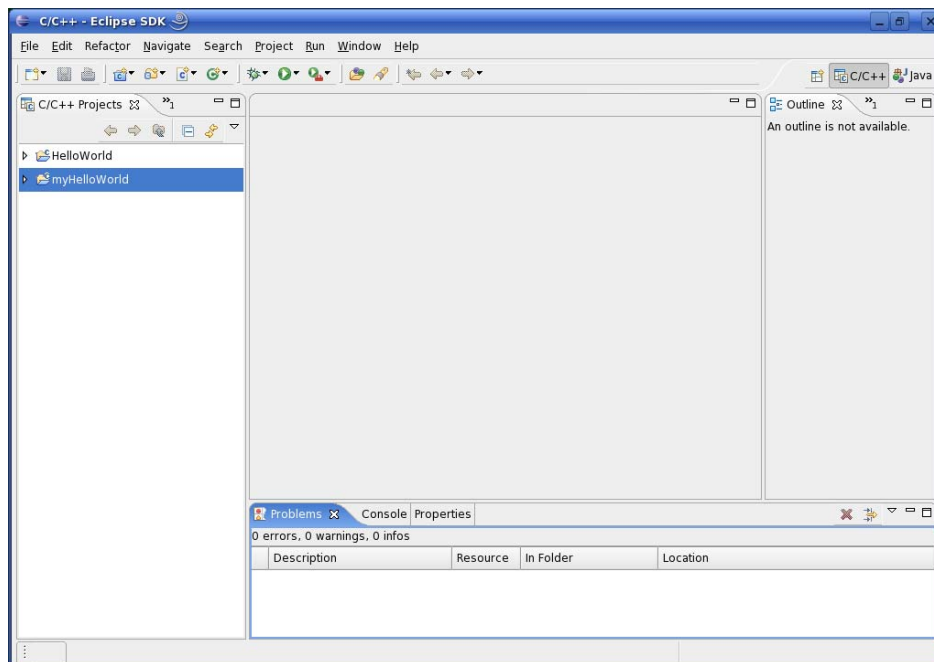


- Click *Finish*.

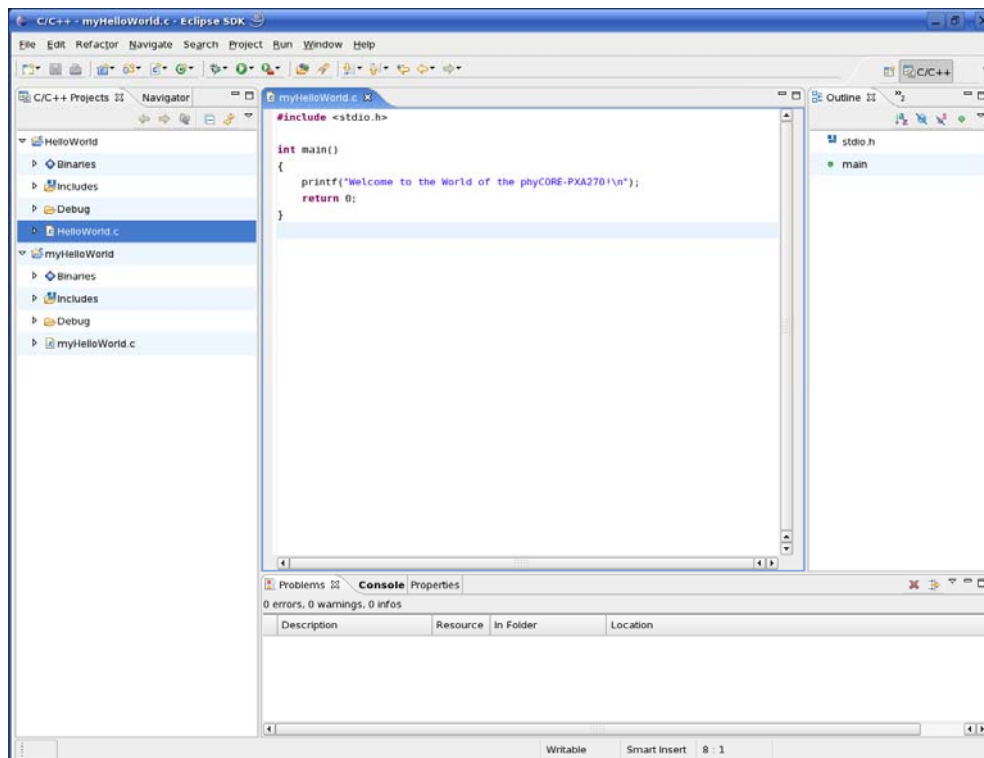


- Select *Yes* to open the C/C++ perspective.

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



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



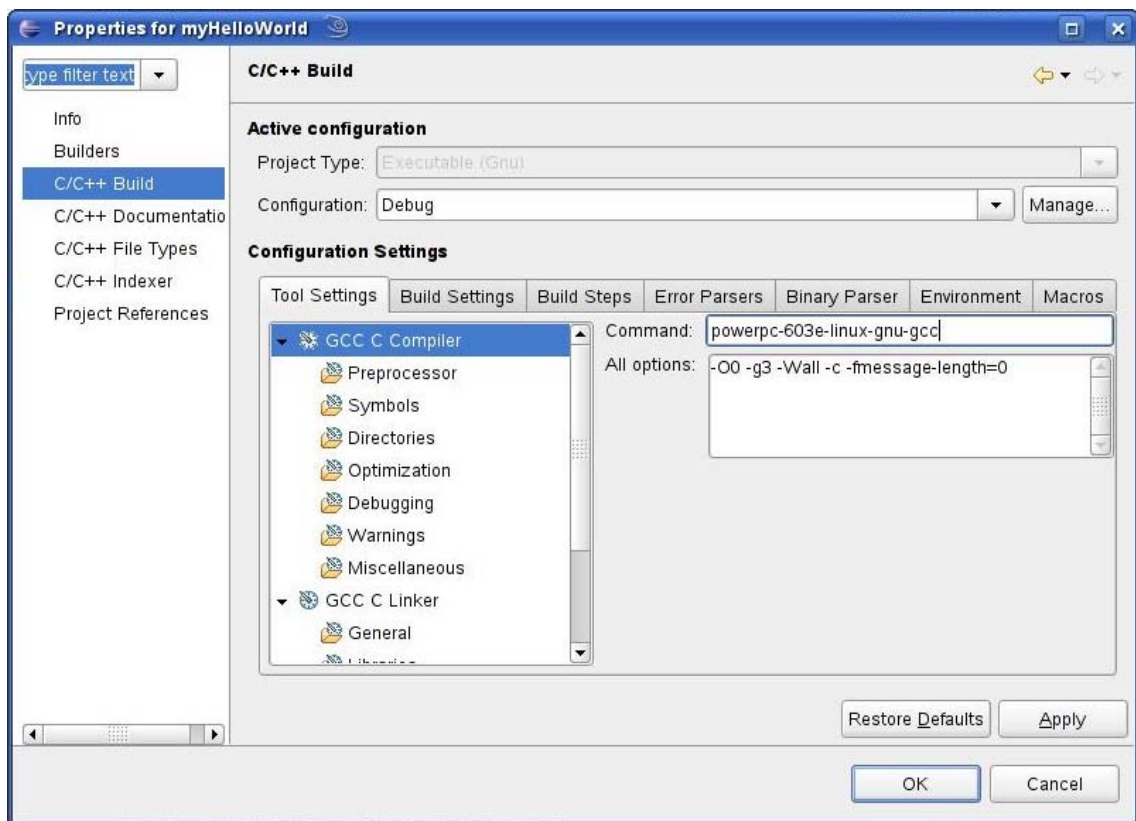
- Select the *myHelloWorld* project.
- Right-click and select *Paste*.
- Double-click on *HelloWorld.c* in the *myHelloWorld* project.

If *Build Automatically* from the *Project* menu is selected, the *HelloWorld* application will now be compiled and created with the standard GNU 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.

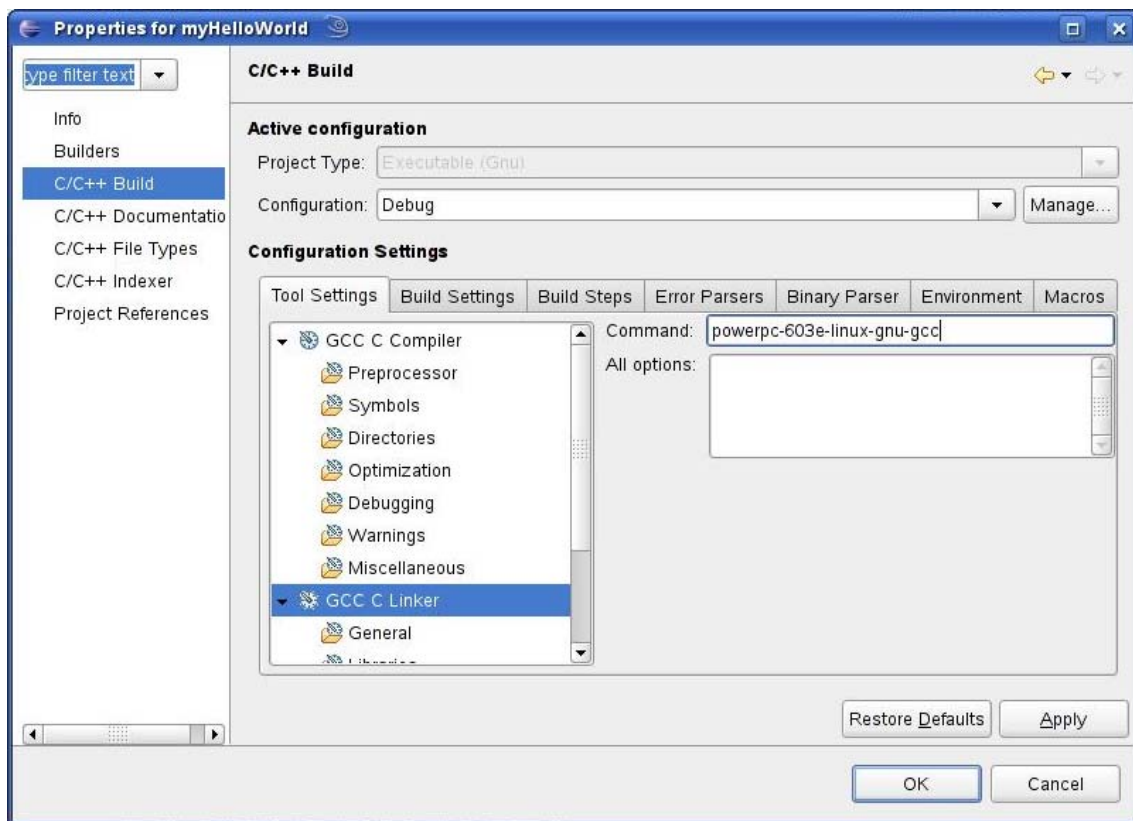
To compile your project for the phyCORE-MPC5121e-tiny instead, you will have to use the GNU C/C++ cross compiler.

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

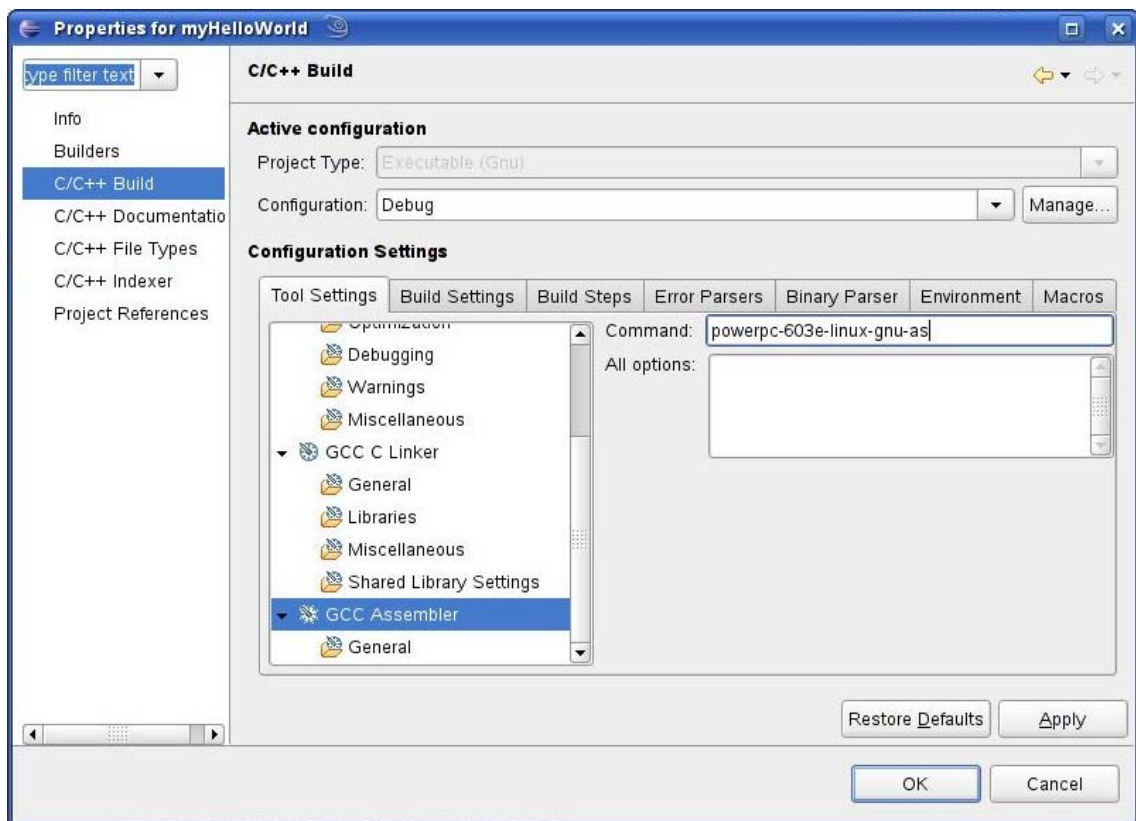
The *Properties* dialog appears.



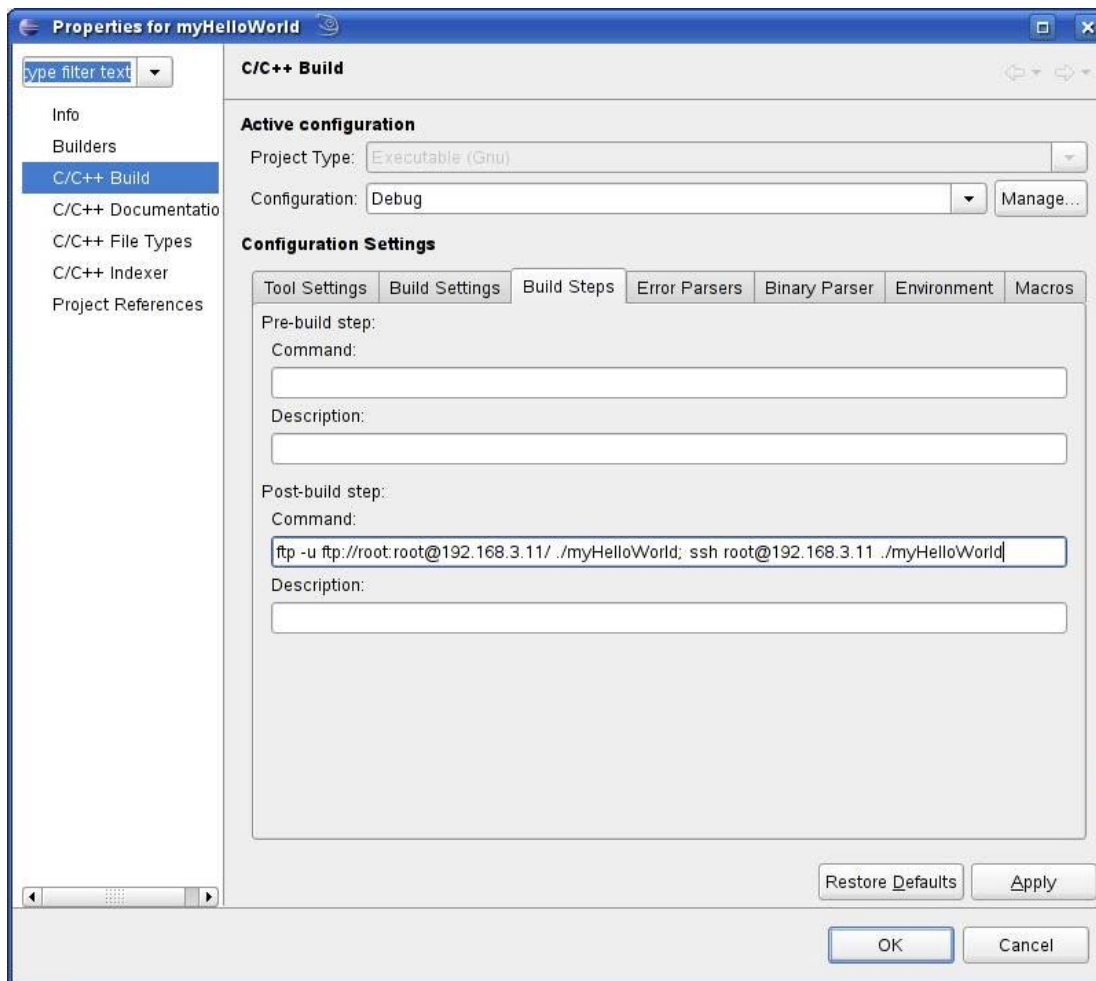
- Select *C/C++ Build*.
- Enter **powerpc-603e-linux-gnu-gcc** into the *Command* input field.



- Select *GCC C Linker*.
- Enter **powerpc-603e-linux-gnu-gcc** into the *Command* input field.



- Select *GCC Assembler*.
- In the *Command* input field, change the default **as** to **powerpc-603e-linux-gnu-as**.
- Click *Apply*.



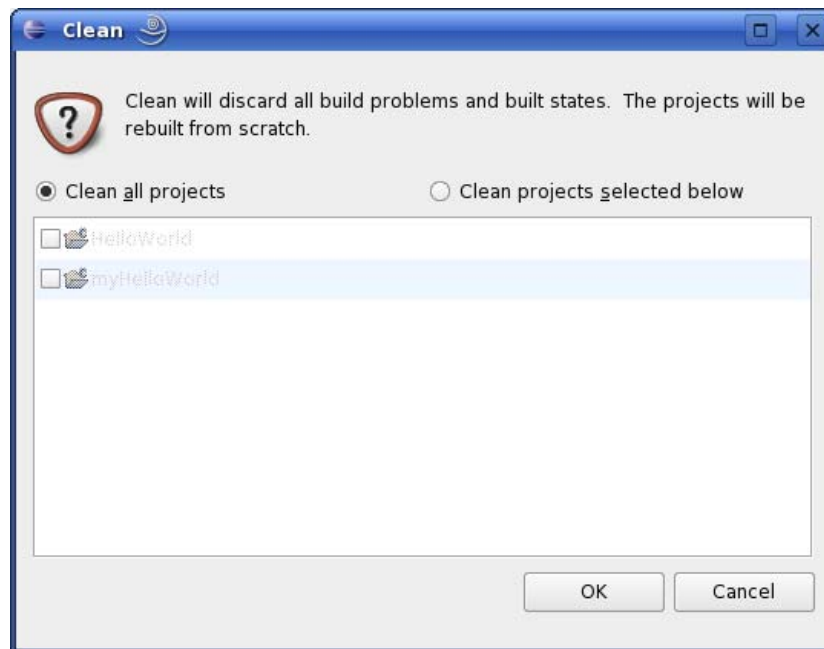
- Select the *Build Steps* tab:
- Enter following command in the *Command* input field:

```
ftp -u ftp://root:root@192.168.3.11/ ./myHelloWorld;  
ssh root@192.168.3.11 ./myHelloWorld
```



Be sure to enter the semicolon between **./myHelloWorld** and **ssh**.

- Click Apply.
- Click OK.

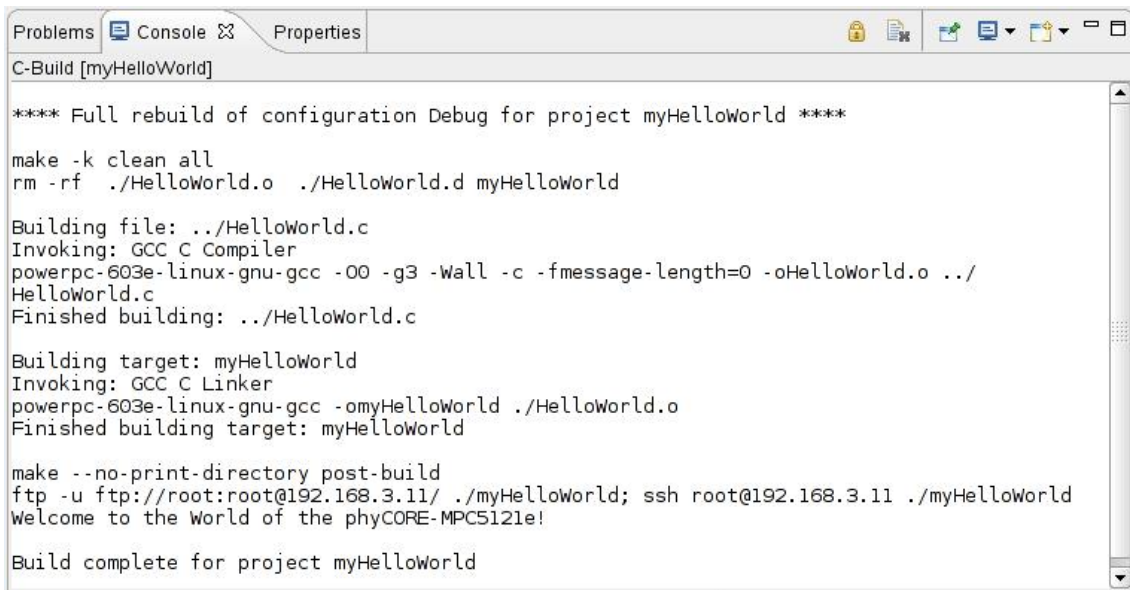


- Select *Project* ► *Clean* from the menu bar.
- Confirm with *OK*.

The project will be rebuilt.

- Select the *Console* tab.

If no errors occur while building, you will see the following output:



```
C-Build [myHelloWorld]

**** Full rebuild of configuration Debug for project myHelloWorld ****

make -k clean all
rm -rf ./HelloWorld.o ./HelloWorld.d myHelloWorld

Building file: ../HelloWorld.c
Invoking: GCC C Compiler
powerpc-603e-linux-gnu-gcc -O0 -g3 -Wall -c -fmessage-length=0 -oHelloWorld.o ../
HelloWorld.c
Finished building: ../HelloWorld.c

Building target: myHelloWorld
Invoking: GCC C Linker
powerpc-603e-linux-gnu-gcc -omyHelloWorld ./HelloWorld.o
Finished building target: myHelloWorld

make --no-print-directory post-build
ftp -u ftp://root:root@192.168.3.11/ ./myHelloWorld; ssh root@192.168.3.11 ./myHelloWorld
Welcome to the World of the phyCORE-MPC5121e!

Build complete for project myHelloWorld
```



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.

3.4 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-MPC5121e-tiny, is connected to the system console */dev/console*):

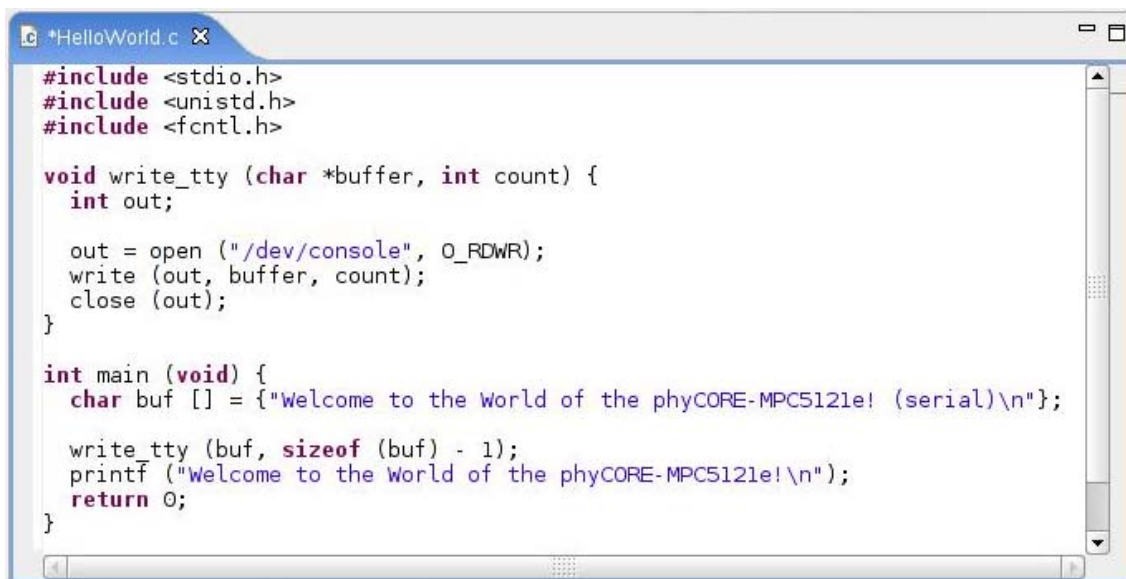
```
void write_tty (char *buffer, int count) {
    int out;

    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
               phyCORE-MPC5121e! (serial)\n"};
write_tty (buf, sizeof (buf) - 1);
```

In the next screenshot you can see the complete program.

A screenshot of a code editor window titled '*HelloWorld.c'. The code is as follows:

```
#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) {
    char buf [] = {"Welcome to the World of the phyCORE-MPC5121e! (serial)\n"};

    write_tty (buf, sizeof (buf) - 1);
    printf ("Welcome to the World of the phyCORE-MPC5121e!\n");
    return 0;
}
```

- Save your program after changing the code.

The application will be compiled, built, copied to the target and executed.

Executing the program on the target using Microcom:

- 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-MPC5121e! (serial)
Welcome to the World of the phyCORE-MPC5121e!
```

- Close Microcom.

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



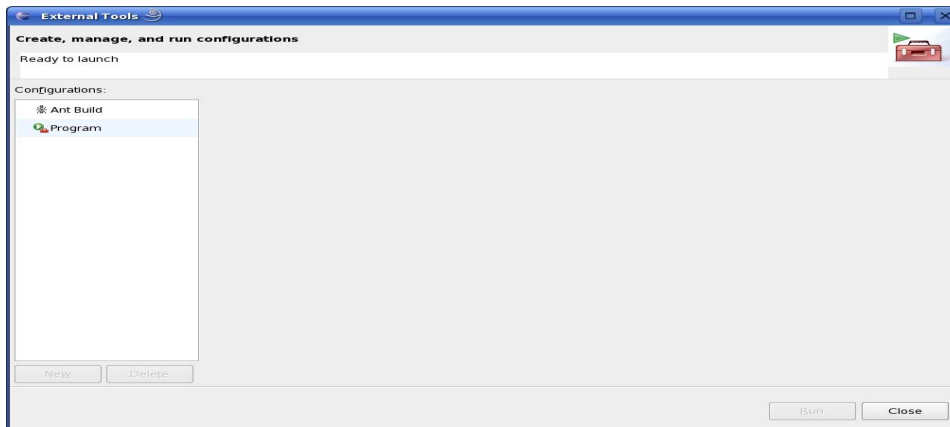
The first line is a direct output on the serial interface. You can't see this line in an SSH session, because you are connected over a TCP/IP connection to the target. With Microcom, however, you have direct access to serial interface, so you can also see the line that is written to the serial console.

In this passage 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()*.

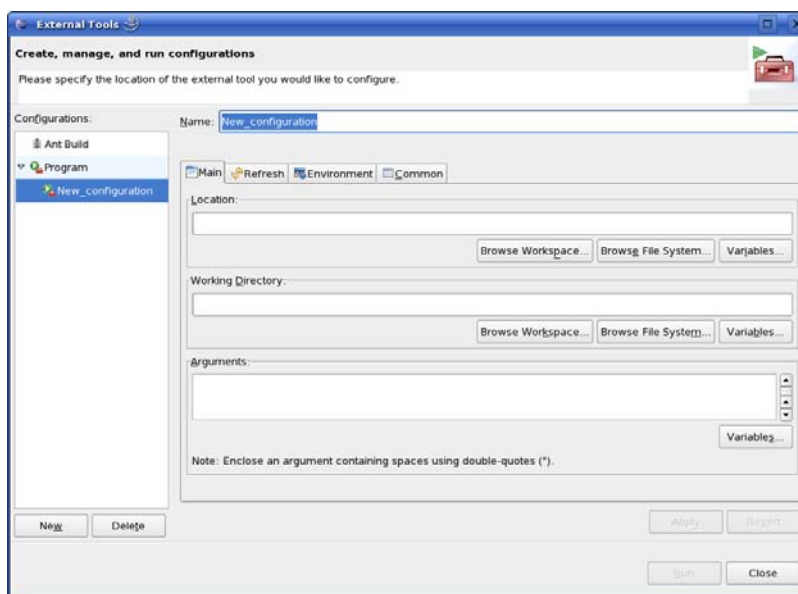
This procedure is in principle quite typical for Linux, because Linux treats everything like a file.

3.5 Starting a Program out of Eclipse on the Target

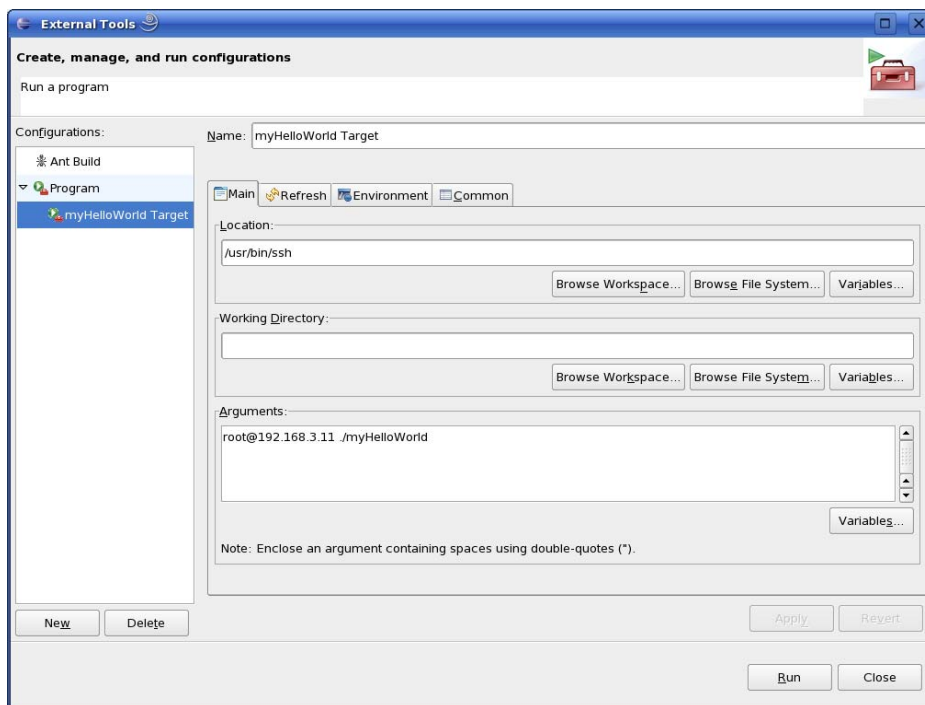
After compiling a project in Eclipse, the program is copied to the target and directly executed. A program can also be executed on the target without compiling a project. In the following section you will learn how to start a program on the target as an external tool.



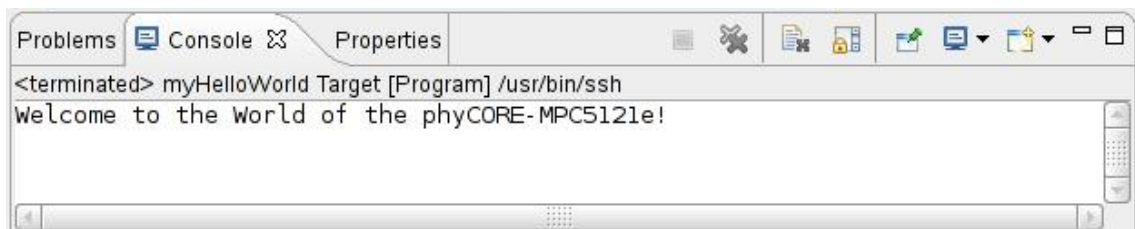
- Select *Run* ► *External Tools* ► *External Tools* from the menu bar.



- Select Program.
- Select New.

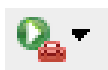


- In the *Name* input field, enter: **myHelloWorld Target**
- Enter **/usr/bin/ssh** in the *Location* input field.
- Enter **root@192.168.3.11 ./myHelloWorld** into the *Arguments* field.
- Select *Apply*.



- Select *Run*.

If you want to execute the program the next time, you can use the *Run External Programs* button from the menu bar.



3.6 Automatically starting the Program when booting the Target

In this passage you will integrate the *myHelloWord* 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.

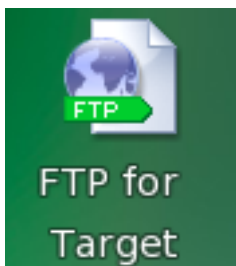


TIP

The scripts for controlling the system will be found in */etc/init.d*. These are executed directly or indirectly by */sbin/init*, the father of all processes. The configuration of the */sbin/init* is placed in */etc/inittab*.

After system startup, */sbin/init* will switch on 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 start script in */etc/init.d*.



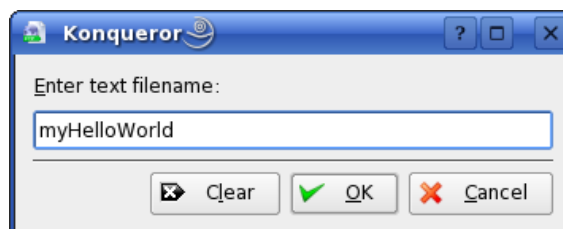
- Click the *FTP for Target* icon on your KDE desktop.



- Browse to the target's */etc/init.d*. If an authorization dialog should appear, just click *OK* (no password is required).

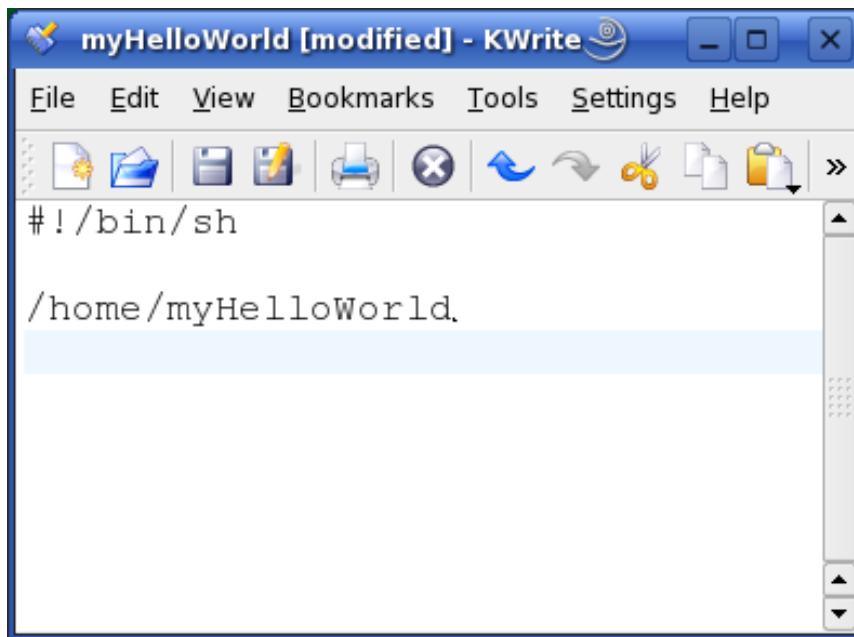
In the directory */etc/init.d* you can see existing scripts.

- Right-click in the opened window and select *Create New* ► *Text File*.



- Enter *myHelloWorld*.
- Click *OK*.
- Right-click on *myHelloWorld* and select *Open with*.
- Enter **kwrite** and click *OK*.

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



- Enter the following two lines:

```
#!/bin/sh
/home/myHelloWorld
```
- Select *File* ► *Save*.
- Close the *KWrite* window.
- Close the FTP window.



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

```

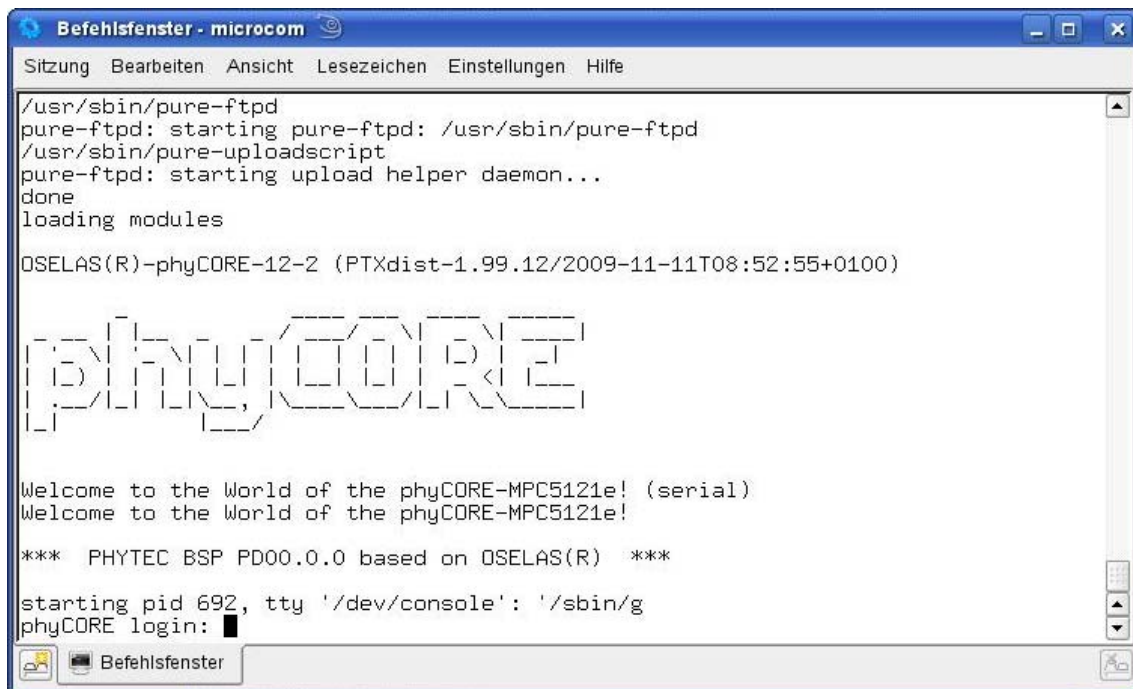
telnet - Konsole
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
root@phyCORE:/etc/rc.d ls -l
lrwxrwxrwx 1 root root 14 Nov 11 2009 S00udev -> ../init.d/udev
lrwxrwxrwx 1 root root 17 Nov 11 2009 S08syslog -> ../init.d/syslogd
lrwxrwxrwx 1 root root 15 Nov 11 2009 S10acpid -> ../init.d/acpid
lrwxrwxrwx 1 root root 14 Nov 11 2009 S12dbus -> ../init.d/dbus
lrwxrwxrwx 1 root root 17 Nov 11 2009 S16openssh -> ../init.d/openssh
lrwxrwxrwx 1 root root 17 Nov 11 2009 S16telnetd -> ../init.d/telnetd
lrwxrwxrwx 1 root root 20 Nov 11 2009 S21alsa-utils -> ../init.d/alsa-utils
lrwxrwxrwx 1 root root 20 Nov 11 2009 S26networking -> ../init.d/networking
lrwxrwxrwx 1 root root 18 Nov 11 2009 S91lighttpd -> ../init.d/lighttpd
lrwxrwxrwx 1 root root 18 Nov 11 2009 S91pureftpd -> ../init.d/pureftpd
lrwxrwxrwx 1 root root 17 Nov 11 2009 S98modules -> ../init.d/modules
lrwxrwxrwx 1 root root 16 Nov 11 2009 S99banner -> ../init.d/banner
-rwxr-xr-x 1 root root 77 Nov 11 2009 S99zzz_PHYTEC_BSP_version_startup_script
root@phyCORE:/etc/rc.d █

```

- Enter **root** and press *Enter* to login.
- The startup script we have just created with KWrite 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 show 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...*, whereas *S99zzz_PHYTEC_BSP_version_startup_script* will be started last. 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.



```
Befehlsfenster - microcom
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
/usr/sbin/pure-ftpd
pure-ftpd: starting pure-ftpd: /usr/sbin/pure-ftpd
/usr/sbin/pure-uploadsript
pure-ftpd: starting upload helper daemon...
done
loading modules
OSELAS(R)-phyCORE-12-2 (PTXdist-1.99.12/2009-11-11T08:52:55+0100)

phyCORE

Welcome to the World of the phyCORE-MPC5121e! (serial)
Welcome to the World of the phyCORE-MPC5121e!

*** PHYTEC BSP PD00.0.0 based on OSELAS(R) ***

starting pid 692, tty '/dev/console': '/sbin/g
phyCORE login: █
```

The program *myHelloWorld* now starts automatically on startup. Because its link in */etc/rc.d* starts with *S99...*, you should see *myHelloWorld*'s output near the output of the two other scripts that start with *S99...* (which print all sorts of version information).

- Close Microcom.

Now you can add your own programs to the root file system and start these programs automatically when your phyCORE-MPC5121e-tiny boots.



You have successfully passed the “Getting More Involved” part of this QuickStart.

4 Debugging an Example Project

**20 min**

TIME

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 or 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.

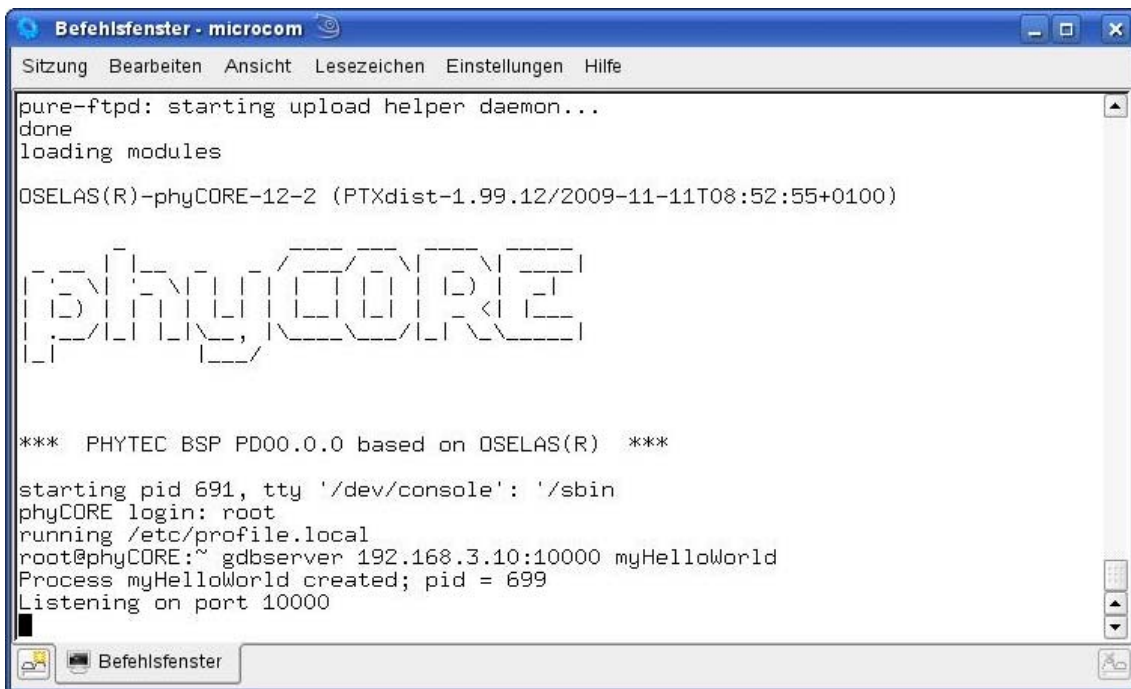
4.1 Starting the GDB Server on the target

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 extending debugging symbols. This has already been added while building the program.



- Open Microcom



```
Befehlsfenster - microcom
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
pure-ftpd: starting upload helper daemon...
done
loading modules
OSELAS(R)-phyCORE-12-2 (PTXdist-1.99.12/2009-11-11T08:52:55+0100)
phyCORE
*** PHYTEC BSP PD00.0.0 based on OSELAS(R) ***
starting pid 691, tty '/dev/console': '/sbin
phyCORE login: root
running /etc/profile.local
root@phyCORE:~ gdbserver 192.168.3.10:10000 myHelloWorld
Process myHelloWorld created; pid = 699
Listening on port 10000
```

- Type **root** and press *Enter*.
- Start the gdbserver:
`gdbserver 192.168.3.10:10000 myHelloWorld`

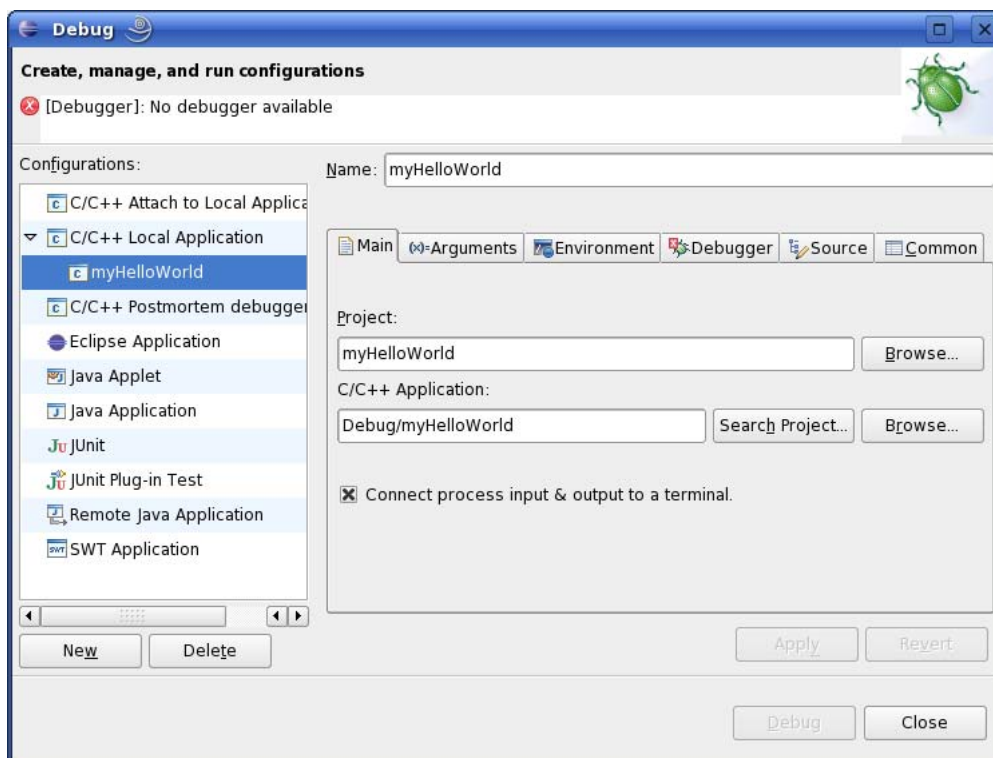
You have started the GDB server on the target. The GDB server is now waiting for connections on TCP port 10000.

4.2 Configuring and Starting the Debugger in Eclipse

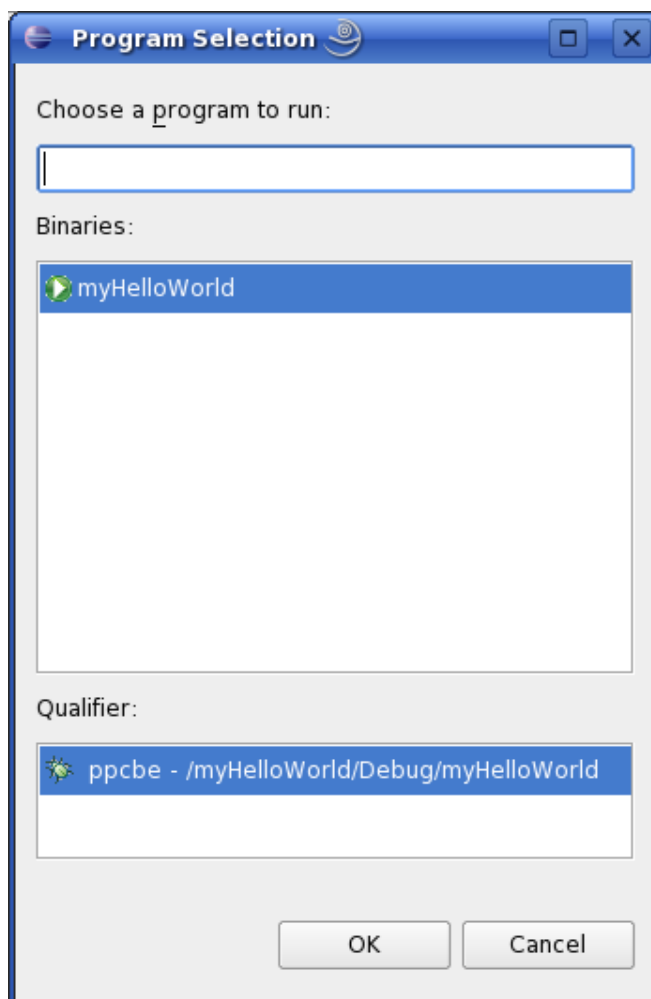
In this passage you will learn how to configure your project settings to use Eclipse with the GNU debugger. After the configuration of your project settings, the GNU debugger will start and connect to the GDB server on the target.

- Start Eclipse if the application is not started yet.
- Select *myHelloWorld* in the *Navigator* window.
- Select *Run* ► *Debug* from the menu bar.

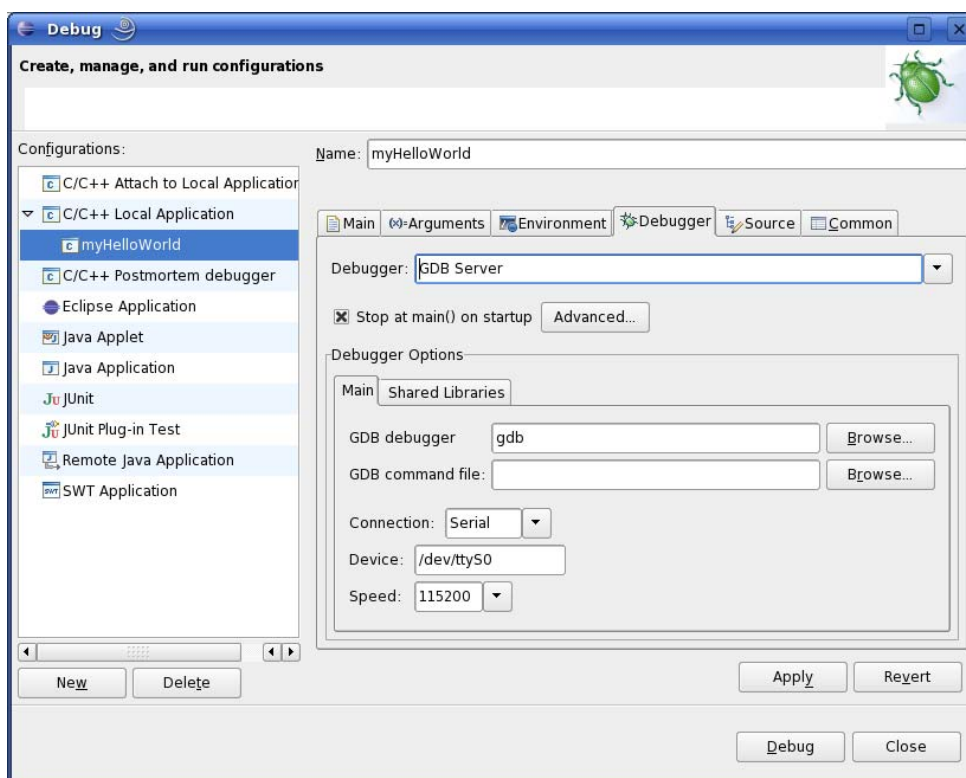
A dialog to create, manage and run applications will appear.



- Select *C/C++ Local Application*.
- Click *New*.



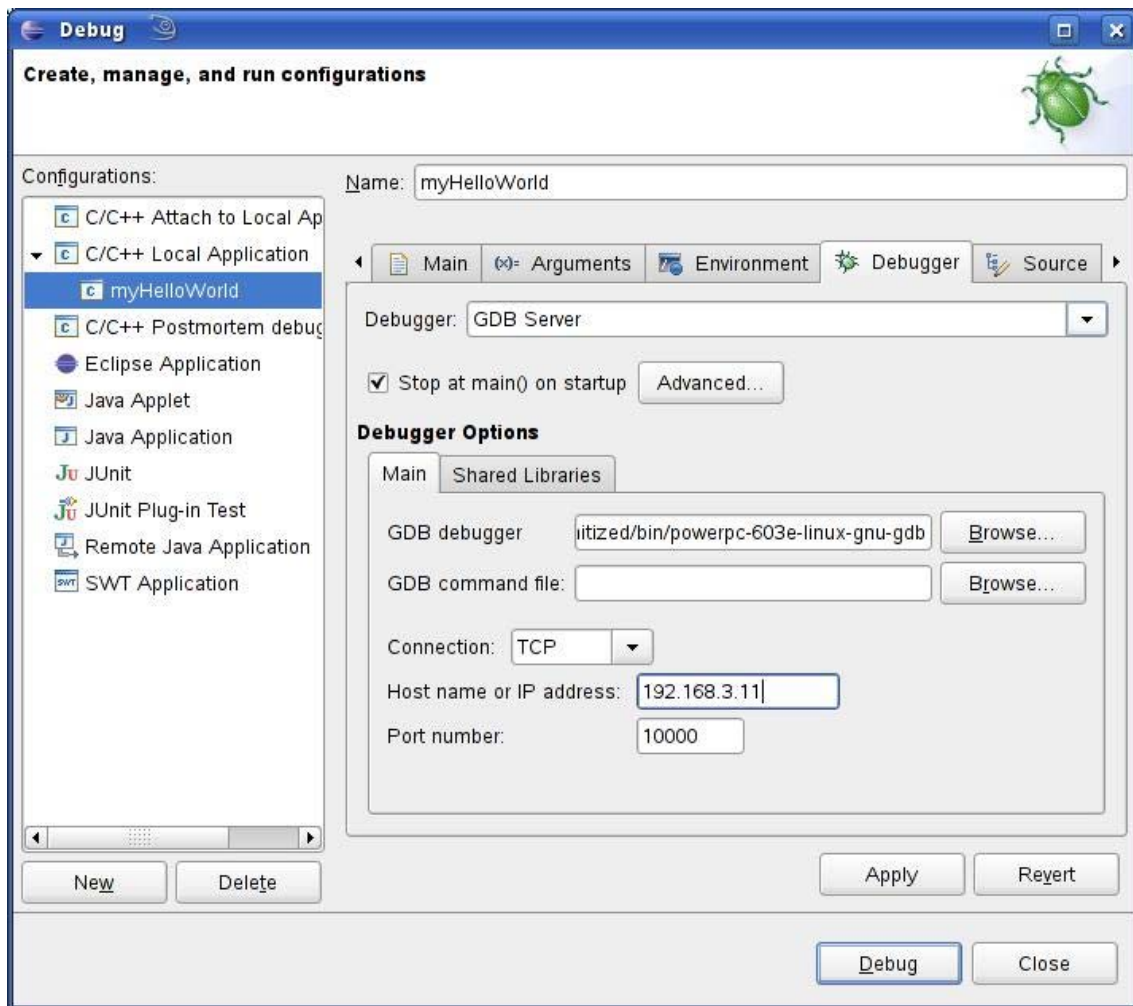
- Select the *Search Project* button.
- Click *OK* .



- Select the *Debugger* tab.
- Select *GDB Server* 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-1.99.3/powerpc-603e-linux-gnu/gcc-4.3.2-glibc-2.8-binutils-2.18-kernel-2.6.27-sanitized/bin*.
- Select the file *powerpc-603e-linux-gnu-gdb*.
- Click *OK*.



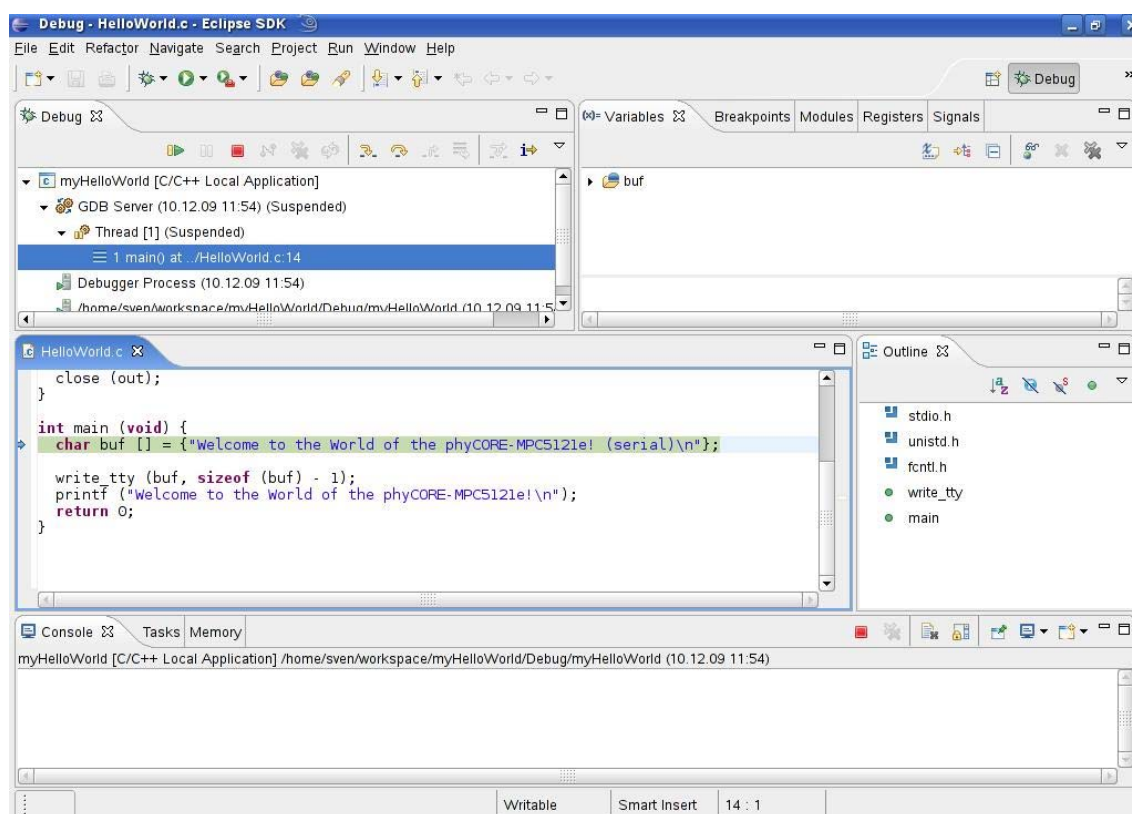
- From the *Connection* drop-down box, select *TCP*.
- Enter **192.168.3.11** (the target's IP address) in the *Host name* input field. The host's GDB will connect to this IP address to communicate with the target's GDB server.
- Click *Apply*.
- Click *Debug*.

A new dialog appears.



- Select *Yes* to switch to the *Debug* perspective.

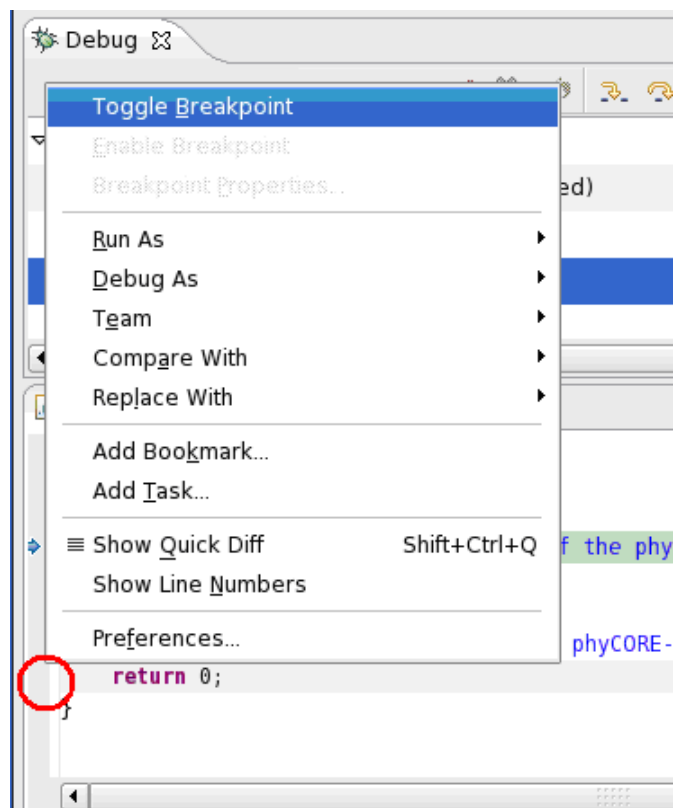
The *Debug* perspective opens and the debugger stops at the first line automatically. The host's GDB is now connected to the GDB server on the target.



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.

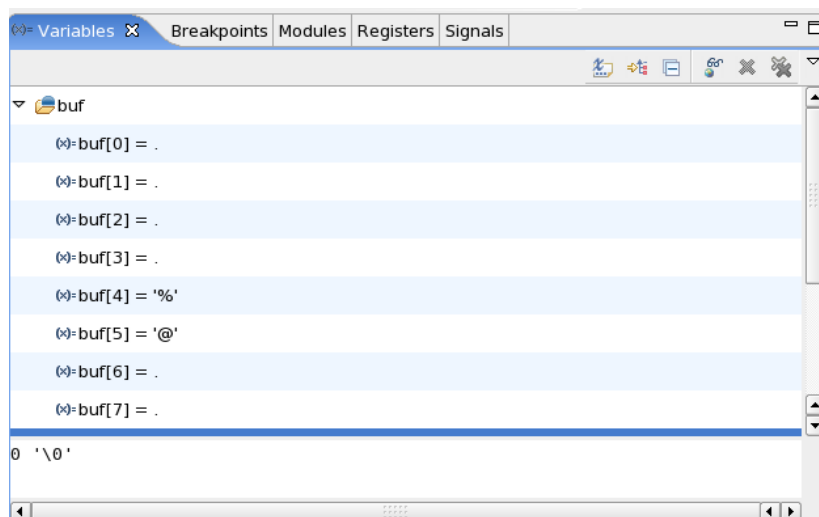



- Select the last line in *main()*.
- *Right-click* into the small grey border on the left-hand side and select *Toggle Breakpoint* to set a new breakpoint.

4.4 Stepping and Watching Variables Contents

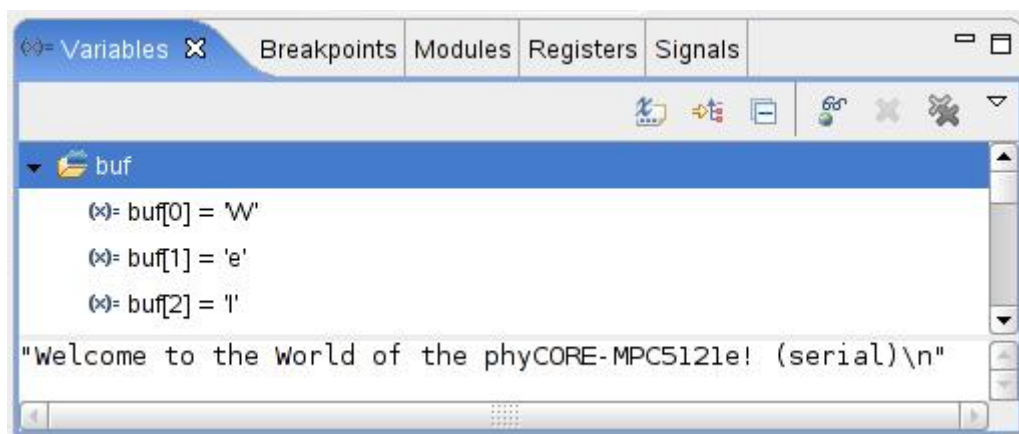
In this part you will step through the example project with the debugger. You will also learn how to watch the content of a variable.

- Expand *buf* in the *Variables* window

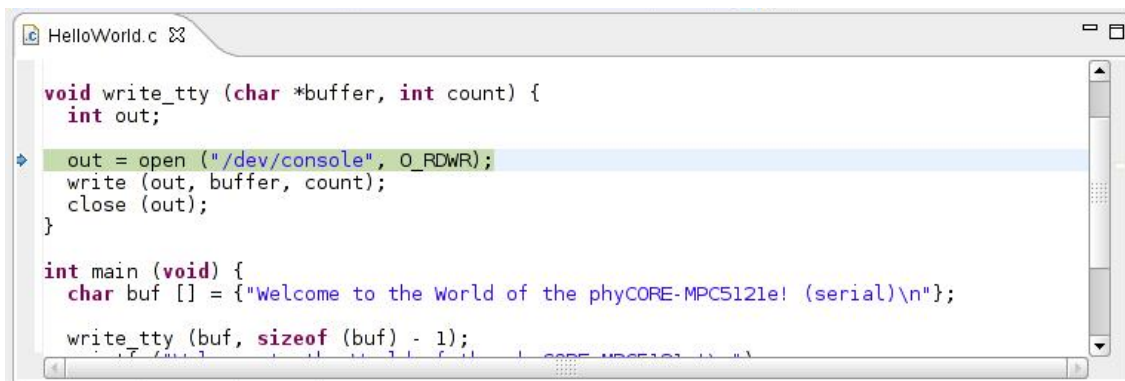


- Click the *Step Over*  button in the *Debug* Window to step to the next line.

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



- Click on the variable *buf*.



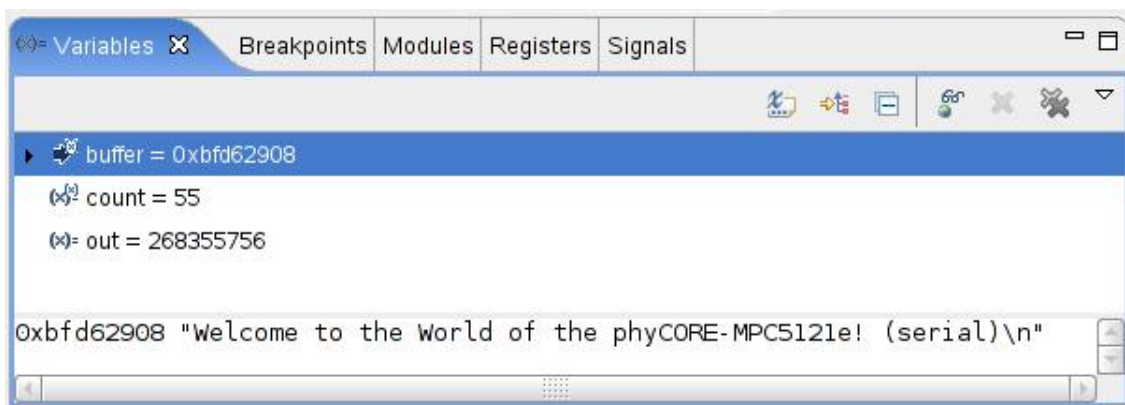
```
void write_tty (char *buffer, int count) {
    int out;
    out = open ("/dev/console", O_RDWR);
    write (out, buffer, count);
    close (out);
}

int main (void) {
    char buf [] = {"Welcome to the World of the phyCORE-MPC5121e! (serial)\n"};
    write_tty (buf, sizeof (buf) - 1);
}
```

- 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:



Variables Breakpoints Modules Registers Signals

buffer = 0xbf62908
count = 55
out = 268355756

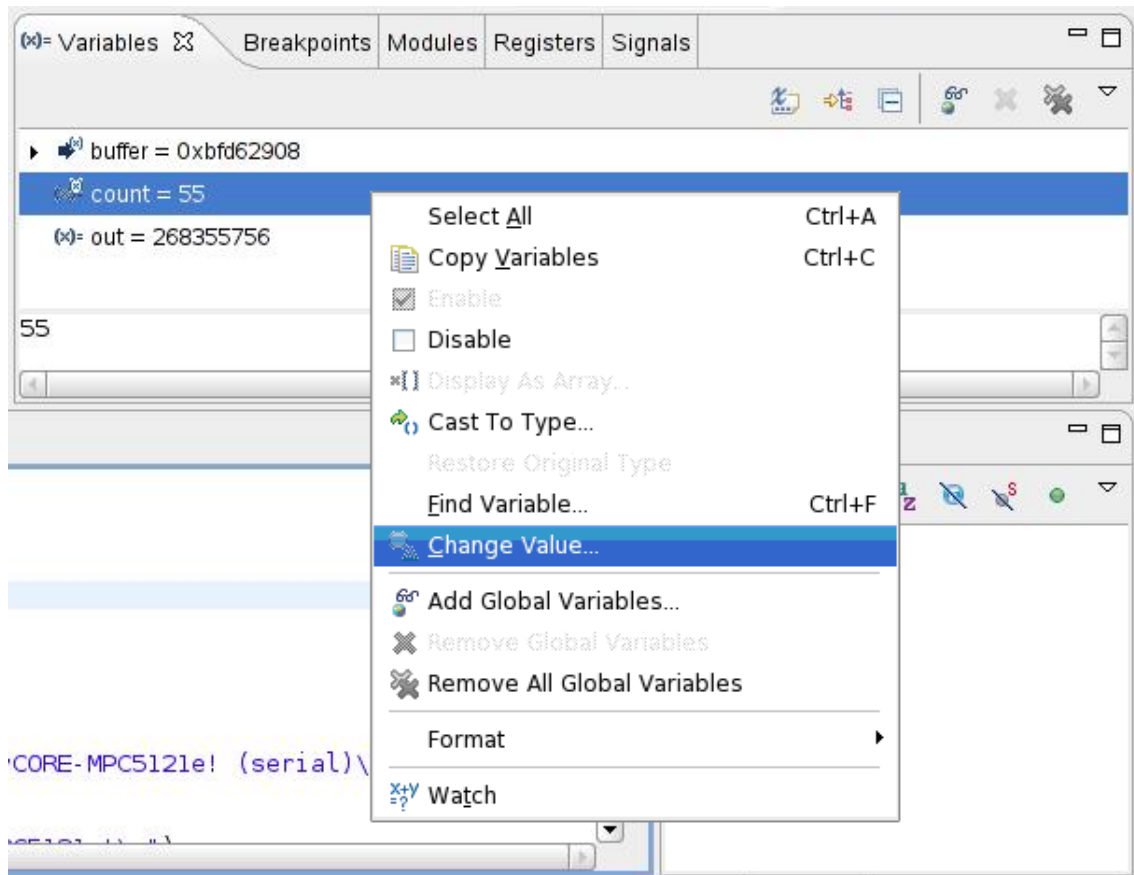
0xbf62908 "welcome to the world of the phyCORE-MPC5121e! (serial)\n"

- Click on the variable *buffer*.

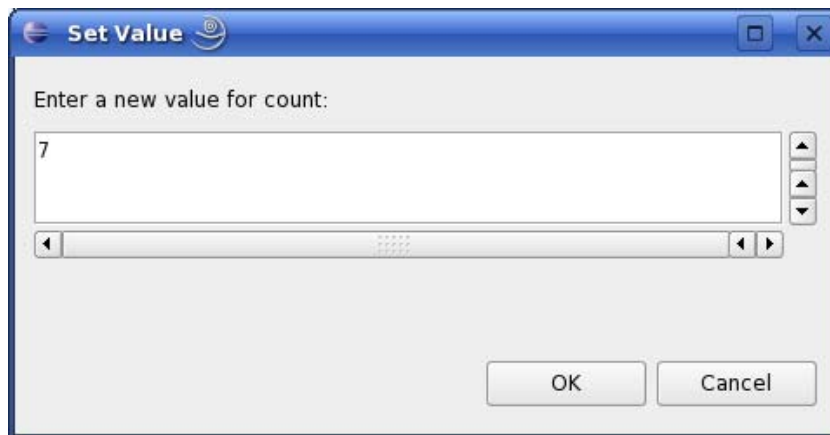
You will probably see a different address at the *buffer* pointer. Remember what address is shown in your case; you will need this address later.


4.5 Changing Variables Values

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*.



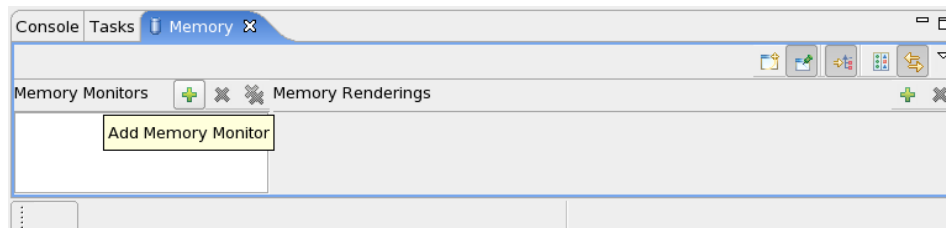
- Change the value of count to **7** and select *OK*.
- Open Microcom if the application is not already opened.
- Go back to Eclipse.
- Click the *Step Over*  button two times.
- Change to Microcom.

```
root@phyCORE:~# gdbserver 192.168.3.10:10000 myHelloWorld
Process myHelloWorld created; pid = 710
Listening on port 10000
Remote debugging from host 192.168.3.10
Welcome
```

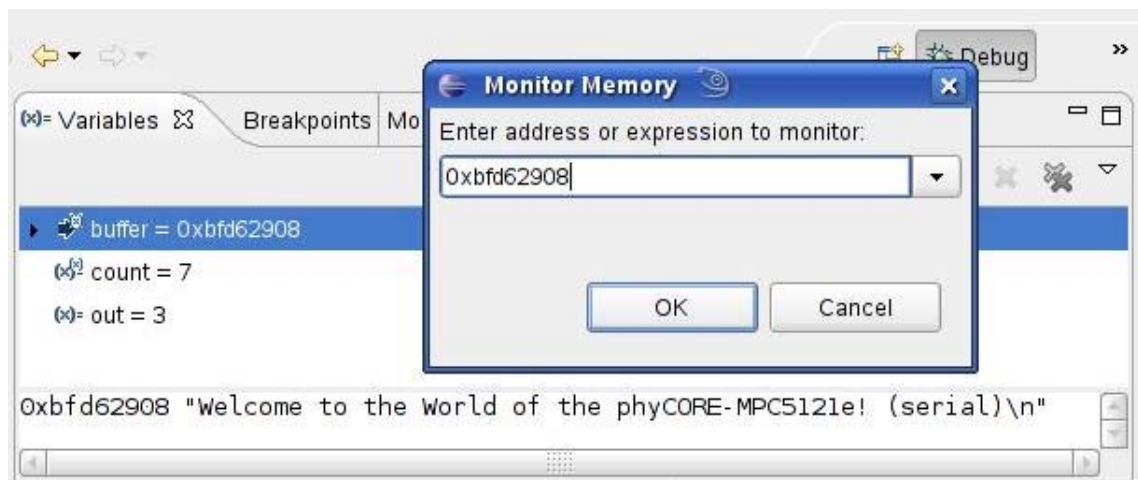
You will see the output *Welcome* in the Microcom window. This means that due to changing the *counter* variable's value, instead of printing the full "Welcome to the World of the phyCORE-MPC5121e!" string, only the first seven characters of the buffer were written to the screen.

4.6 Using the Memory Monitor

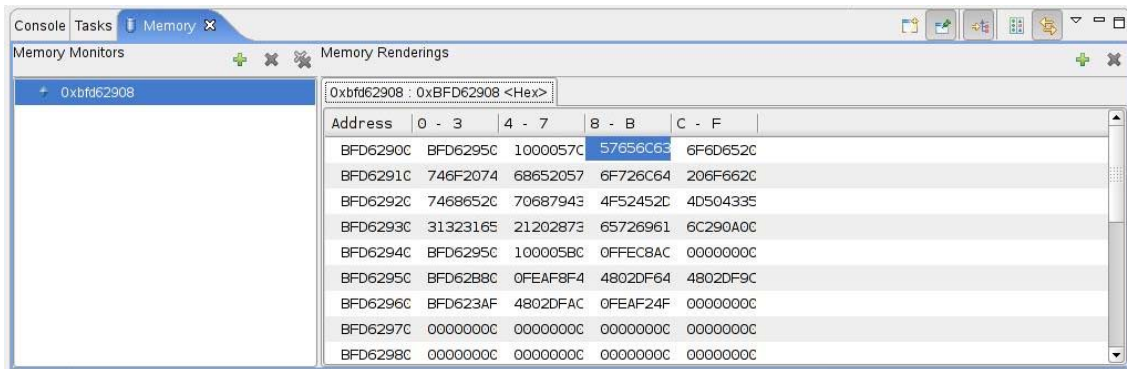
At the last part in this chapter you will use the memory monitor to watch the content at a memory address.



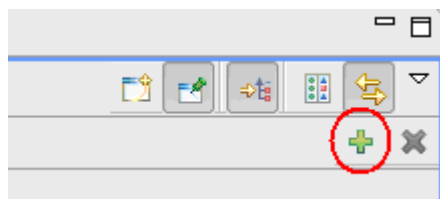
- Select the *Memory* tab in the window below.
- Click *Add Memory Monitor*.



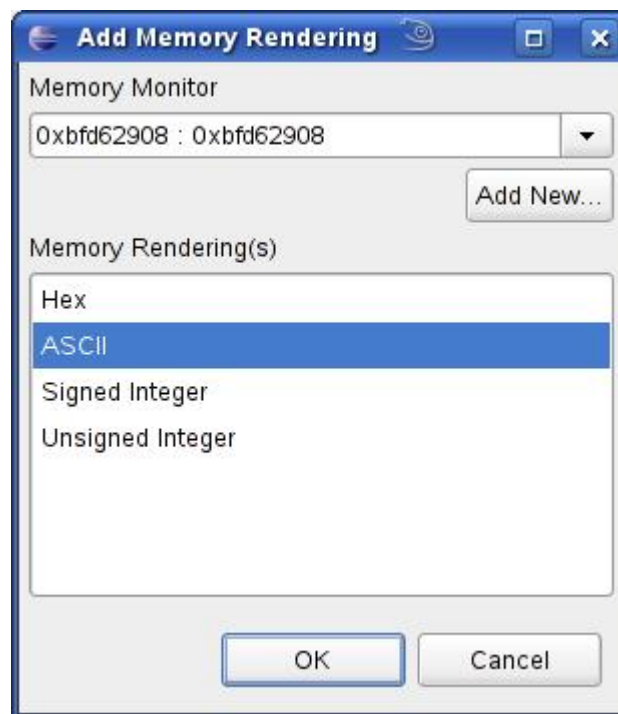
- Enter the address of *buffer* and click *OK*. Remember that the variable's address might differ on your system.



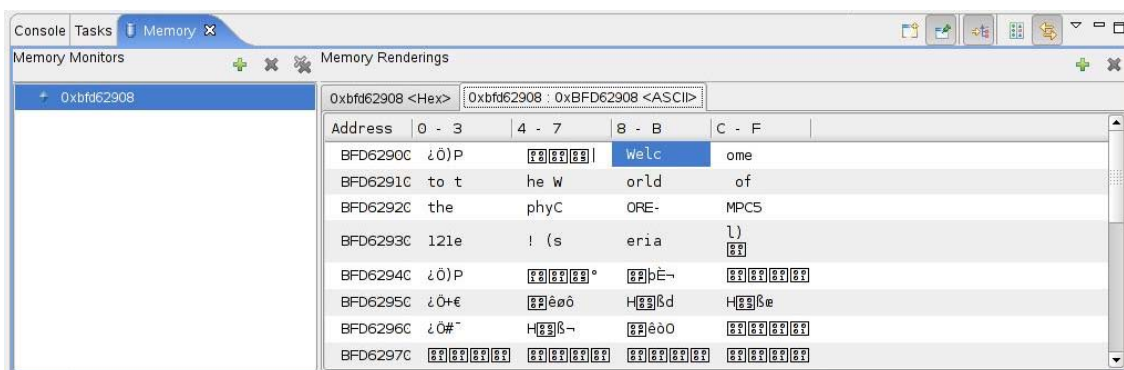
- Change the window size.



- Click *Add Rendering*.

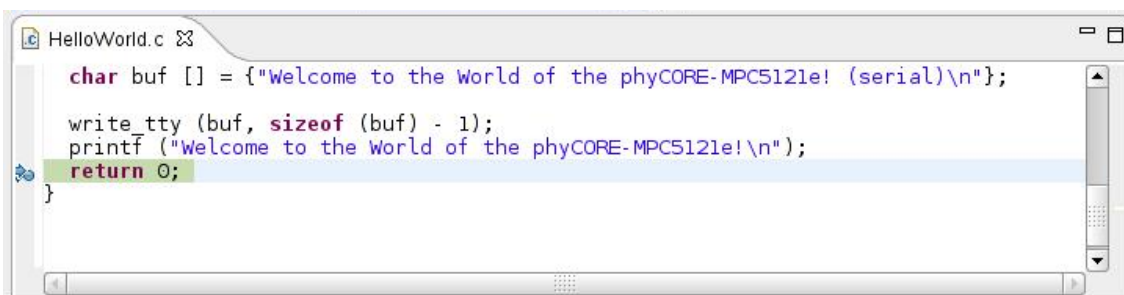


- Select ASCII and click *OK*.



You can see the content of the variable `buf` at the address `0xbfd62908` (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.

5 Further Information

In the PTXdist User Manual you can find further information. You will find the manual in the directory *OSELAS.BSP-Phytec-phyCORE-MPC512x-tiny-PD09.1.0/documentation* of the archive *OSELAS.BSP-Phytec-phyCORE-MPC512x-tiny-PD09.1.0.tar.gz* on your setup cdrom.

The PTXdist User Manual includes information about the following topics:

- Installation and Configuration of PTXdist
- Using and Building a Toolchain
- Create and activate a project
- Running phyCORE-MPC5121e-tiny from network only
- Running phyCORE-MPC5121e-tiny stand alone
- U-Boot and phyCORE-MPC5121e-tiny
- phyCORE-MPC5121e-tiny's BSP

6 Summary

This QuickStart manual provided 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[®]-MPC5121e-tiny in conjunction with the Eclipse IDE and GNU C/C++ software tools.

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

In the *Getting More Involved* section you got a step-by-step instruction on how to configure and build a new kernel, modify the example application, create and build new projects and copy output files to the phyCORE – MPC5121e-tiny using Eclipse.

The *Debugging* part of this QuickStart 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.

Document: phyCORE[®]-MPC5121e-tiny with Linux
QuickStart Instructions
Document number: L-747e_0, December 2009

How would you improve this manual?

Did you find any mistakes in this manual? page

Submitted by:

Customer number: _____

Name: _____

Company: _____

Address: _____

Return to:

PHYTEC Technologie Holding AG
Robert-Koch-Str. 39
55129 Mainz
Fax: +49 (6131) 9221-26

Published by

PHYTEC

© PHYTEC Messtechnik GmbH

Ordering No. L-747e_0
Printed in Germany